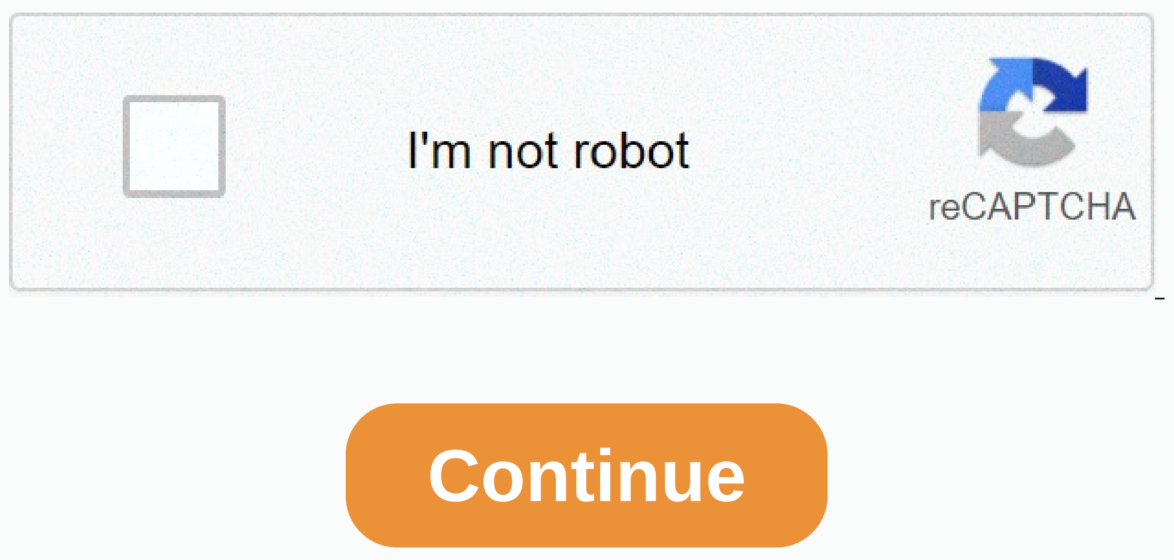

□ I'm not robot
reCAPTCHA
Continue

C primer plus answers

Chapter 1 Chapter 2 Chapter 3 Chapter 4 Chapter 5 Chapter 6 Chapter 7 Chapter 8 Chapter 9 Chapter 10 Chapter 11 Chapter 12 Chapter 13 Chapter 14 Chapter 15 Chapter 16 Chapter 18 Chapter 19 Chapter 20 Chapter 21 Chapter 22 Chapter 23 Page 2 1:Give a reason why computers permead our society today. A:Computers are versatile and can be programmed to solve a wide range of problems. 2:What are the differences between a computer programmer and an end user? A:A computer programmer uses programming language to write the exact instructions that make up the computer program that the end user will apply. 3:Give a brief description of how the world is viewed from the point of view of an object-oriented developer. A: An object-oriented world consists of objects that interact to solve different tasks. 4:What are the two basic parts of the computer? A:Hardware and software. 5:What is software? A:The software consists of instructions that the computer performs. 6:How does a computer user interact with a computer? A:The user provides data (called input) to the computer (via keyboard or something like that) that the computer processes. The computer corresponds to the user's output (for example, through the screen). 7:What is a processor? A:The processor is an important part of the computer hardware that executes the instructions of the computer program. 8:Specify two categories of computer memory. Briefly describe each of them. A:Auxiliary memory and main memory. Auxiliary memory permanently stores data, while main memory represents temporary storage that contains parts of the startup program and related mid-calculation results. 9:What data would be included in the calculation of average monthly rainfall in New York City during 2000? A: Total rainfall during 2000 10:What are bits and bytes? A: Few can have one of two values, often referred to as 1 and 0. Byte consists of eight bits. 11:How many different numbers can be represented by a single byte? A: Each of the eight bits in the byte can represent two values. Thus, the byte can represent 2 x 2 x 2 x 2 x 2 x 2 x 2 x 2 = 256 different numbers. 12:How does main memory organize the data it stores? A:Through a list of numeric locations called bytes. Each location has an address for a simple reference. 13:What are files used for? A:Permanently store different types of data. 14:What is the original program? A:The original program consists of text with high-level instructions that are translated into machine-level instructions executed by the computer. The original program is kept in the original program. 15:How can you identify the C# original program file? A:A C# original file program has an extension .cs (for c sharp) as in YourProgram.cs. 16:Describe what the compiler is doing? A:The compactor translates the instructions on the high level of the original program file to a low level language that a computer processor can execute. 17:List several services provided by .NET A:Among many other services, .NET: Contains an execution engine that can execute MSIL instructions that represent instructions at the high level of C# program Holds a large selection of pre-written program parts that you can use in your programs Manages the computer memory used by your computer program 18:What is MSIL? How does it improve .NET architecture over previous architectures? A:Microsoft Intermediate Language (MSIL) provides a middle layer between a high-level source program and a low-level machine language, thereby decoding the two parts. Page 3 1: What are the four basic activities in the software development process? Describe each activity briefly. A: The four basic activities of the software development process are software specification Determining software design requirements Determining how the software specification can be converted into a fully functional program Writing the text of the source code program, which in our case consists of C# language constructs software validation and correction by verifying that the program is doing what is specified in its software specification 2: What is the algorithm and how can it be expressed? A:The algorithm is a set of precise instructions that the computer follows to perform the task. For example, algorithms can be expressed using a pseudocode or high-level language such as C#. 3:Why do computers need a very precise set of troubleshooting instructions? A:Computers are very fast, but extremely stupid when trying to understand a computer program. They don't assume and should be told exactly what to do. A:Execution begins with the highest line and moves through lines one by one as they appear in pseudocode or C# code. The execution ends on the back line. 5:S what are the three different errors that developers encounter? Describe each of them briefly. A: Syntax errors occur when the source code program does not follow the (grammatical) rules (syntax) outlined in the C# language for writing instructions. Errors in logic represent flaws in the logic of the program. Work time errors range from errors directly caused by the developer to hard-to-prevent issues, such as trying to access corrupted or nonexistent files. 6:Why is the computer testing a cunning endeavor? A:Most computer programs are so complex that it is impossible to test all the different scenarios that the program will endure. Logic errors can still occur in programs without syntax errors. 7:Briefly describe process-oriented programming. A:Process-oriented programming focuses on actions, not related data. 8:Briefly describe the object-oriented program. What are the benefits compared to process-oriented programming? A:Object-oriented programming combines actions and their related data to form standalone units called classes, with an equal emphasis on actions and data. The facilities are currently in class. Objects collaborate to solve tasks by sending messages to each other. One of the main advantages of object-oriented programming is that it allows developers to divide the program into smaller, simpler, standalone units with the resulting reduction in complexity. 9:What is an object? What does it consist of? A:The object is a fairly self-deprecating module that contains data and actions that affect this data. 10:What is the relationship between the class and its subjects? The A:A class is a blueprint for generating objects. The class can be seen in the source code, while the object is created based on class while running the program. 11:How do objects of a particular program problem often be identified? A:Describing the problem of programming in spoken language such as English and then identifying the nouns used in this description. 12:What is a reusable software part? A:Software component. 13:What is the basic unit of re-need in . Net? Describe this unit. A: Assembly is the basic unit of re-need in . net. This is a self-describing element consisting of MSIL, metadata, and resources. 14:What part. net offers software components for re-creation? A:.NET Frameworks Class Library. 15:What are some of the design goals of C# and .NET? A:The designers wanted C# to be an evolutionary step up from previous powerful languages (C and C++, in particular) upgrading the strengths of these languages and allowing developers to migrate to C#. At the same time, they wanted C# to be simpler, safer and more productive than their predecessors. C# is the first real component-oriented language, it provides full support to RAD, and it is a real object-oriented language where everything can be viewed as an object. 16:What are the main steps needed to run a simple C# program? A:C# source code was written with the text editor. The source code is compiled into MSIL using the C# Language Builder. The .NET performance engine executes MSIL to start the program. 17:Can the C# program accepted by the compiler still contain errors? A:Yes, the compiler checks for syntax errors, but cannot know what you want your program to do, so it is unable to verify that there are any logical errors. Responses to Chapter 2 2 program exercises: Write a pseudocode for an algorithm that specifies the maximum number of two default numbers. Tip: Use the following logic: If number 1 is greater than 2, number one is the maximum number; otherwise, number2 is greater than or equal to 1. A:Exercise 1: Take a piece of paper and a pencil. Write number1 on the left side of the paper and number2 on the right paper and the largest number below in the middle of the page. Write the number that number1 represents below 1. Write the number that number2 represents below number2. If the number associated with number1 is greater than the number associated with number2, write this former number below Maximum, otherwise write the number associated with the number2 under the maximum number. 2:Write a pseudocode for an algorithm that specifies the maximum number in a specific list of numbers. A list can contain any amount of numbers. Tip: Keep a variable called maxSoFar (written on paper, with zero as the starting value). Walk through the list one by one and compare each number with maxSoFar. Any number thought to be greater than maxSoFar becomes equal to maxSoFar. At the end of the list, maxSoFar represents the largest number in the list. A:Exercise 2: 1. Take a piece of paper and a pencil. 2. Write maxSoFar with zero below. 3. Repeat the following action (3a) for each value in the list: 3a. If the current value we are focusing on in the list is greater than the value written under maxSoFar then put a line through this previous number and write a higher value under the old crossed number. 4. The value written furthest down that is not crossed is the largest number in the list of numbers. 3:Change the average pseudocode in the Algorithms and Pseudocode section so that it does not divide by zero, even when no numbers have been provided. A:Exercise 3: Change point 4 of the pseudocode to: 4. If ListSize is greater than 0, divide Sum with the ListSize and write down the number another way if ListSize is 0 then simply write down 0. 4:Write, assemble, and run a C# program that says the following on the console: Our alien friend lives on the planet Blipos away A:Exercise 4: using the system; storyteller public class { Public Static Void Main() { Console.WriteLine(Our Alien Friend); Console.WriteLine(lives on the planet Blipos); Console.WriteLine(away); } Page 4 1:How does abstraction help the developer cope with complexity? A:Abstraction allows developers to simplify the problematic domain they are trying to turn into a computer program and allows them to focus only on their relevant parts. 2:Is the idea behind encapsulation limited to software design? Give an example from everyday life. A:No, the idea behind encapsulation is not limited to software design. Example: Although the car's engine can be adjusted in a number of ways, the driver can only regulate it in a few simple areas; the rest are encapsuled by the driver and preset from the factory or controlled by the computers in the vehicle. If the driver could make all possible adjustments, it would lead to two problems. It would be complicated to drive a car, and the likelihood of the driver hurting would increase for several months. 3:What are the benefits of using encapsulation in software construction? A:Encapsulation allows an object to exhibit only parts that are relevant to the outside world. This prevents external users from tampering with the internal parts of the property. At the same time, it becomes easier to manage the object because irrelevant complicating parts are not part of the object interface. 4:What are the two C# keywords important for the implementation of encapsulation? A:private and public. 5:What are the differences between a class and its objects? The A:A class is a passive blueprint (which exists in source code) for its active object counter parts. 6:What is the significance of the class interface? The A:Class interface is what the outside world uses to communicate with objects of this class. 7:How do I determine the beginning of a comment? A:S two double forward cutting signs //. 8:Why use comments if the compiler ignores them? A:People who read source code look at comments to better understand the source code. 9:What are keywords and identifiers? A:The keyword has a special predefined meaning in the C# language. It's part of C# vocabulary, just like it's part of English vocabulary and has a special predefined meaning. An identifier is another word for name. Identifiers are used to identify C# elements, such as classes, methods, and variables, that must have a related identifier. 10:How is the block listed in C#? What are blocks used for? A:The block is enclosed by curly braces {}. Represents the logical unit in C#. For example, a block can belong to a class or method. 11:Can you write a program without the main method? Why or why not? A:No, because the work time always calls the main method of the program to begin execution. 12:What are the essential parts of the variable? A: His identifier, his type and his value. 13:What is called simple C# instruction? How do you break up? A: It is called a statement and must be abolished by the semicolon. 14:What class and method can you invite to the .NET Framework class library to print text on the console? Write a statement that says my dog is brown. A:System.Console class writing (or writing) method can be used to print text on the screen, as in the following: System.Console.WriteLine; 15:What is the name of the method? What happens when the method is called? A:By writing the method identifier and providing the arguments that the method requires. If the method is named outside the object it is in, the name of the object must also be specified. When a method is invoked, the flow of execution enters the body of the method and the statements contained herein are executed. Then the execution is returned to the caller. 16:What is the task? What symbol is used to perform the task? A:Mechanism of giving a new value to variables invited the task. The equal sign (=) is used to mark a task. 17:What is the advantage of declaring variables? A:Ensures that the program contains only correctly declared variables. This prevents stray variables, misguided due to misspelled and the like, from appearing in the C# program. 18:How can you get the program to decide between executions twice? A: Using a statement if. 19:What is white space? Does the computer care about white space? A:Whitespace consists of blank lines, space signs, card signs and transport returns. The compiler in most cases ignores white space, but the chips can not break the white space. 20:Do all developers have to follow the same style to write valid C# programs? A:No, many different valid styles are applied to developer communities. Answers to Chapter 3 Program Exercises 1: Instead of printing Bye Bye! As the last text on command consoles before the program is finished, change the source code so that the program writes Bye Bye. Make a good day! 2:Instead of typing y to make the program printing Hello World!, have user type Yes. Notify the user by changing line 10. 3:Instead of using a variable name response to hold entries from users, change the name to userInput. 4:Have the program print an additional line under Bye Bye. Have a good day! saying: The programme is being terminated. 5:Declare another string type variable named userName. Before the press program Do you want me to write familiar words?, ask the user to type their name. After the user entered their name, did the program read this name and store it in the username variable. Then to programmically print Hello followed by the content of the username. Tip: The last print can be achieved by typing the system. Console.WriteLine(Hello + userName); Typical program execution should result in the following output: Type your name Deborah&lt;enter&gt; Hello Deborah Do you want me to write familiar words? Type Yes for Yes; n for No. Then &lt;enter&gt;. Da&lt;enter&gt; Hello World! bye bye. have a good day! The program is being interrupted. A:The following list provides answers to exercises 15. This is a simple C # program class Hello { // The program begins with a call to Main() public static void Main() { string userInput; string userName; System.Console.WriteLine; username = System.Console.ReadLine(); System.Console.WriteLine(Hello + UserName); System.Console.WriteLine;Do you want me to write two familiar words?); System.Console.WriteLine. Then &lt;enter&gt;); userInput = System.Console.ReadLine(); if (userInput == Yes) System.Console.WriteLine(Hello World!); System.Console.WriteLine (Bye Bye. Have a good day); System.Console.WriteLine(The program is terminated); } } &lt;/enter&gt;&lt;/enter&gt;&lt;/enter&gt;&lt;/enter&gt;5 1:What are namespaces used for in C# and .NET? A:Namespaces act as containers for classes. They help us organize the classes of our programs and provide easy access to them (when they use them again) externally. 2:What is the advantage of including a keyword followed by the name of the namespace at the beginning of the program? A: This allows us to write short names instead of long, fully qualified names when we invite ourselves to classes in a namespace. This makes typing the name easier and improves the readability of the source code. 3:Which name space contains classes associated with mathematical calculations and console input/output. A: System namespace. 4:How should comments be applied in source code? A: Like the salt in your food, excessive amounts of comments are just as damaging as too few comments. 5:Describe the int type variable. A:An int type variable takes up 32 bits of memory and can represent integers between 2147483648 and 2147483647. 6:Why are x and y often considered unacceptable variable identifiers? Why are they eligible in the 4.1 list? A:X and y identifiers do not transmit any information about what value they represent, unlike names such as accountBalance and height. However, x and y in list 4.1 represent different values that cannot be specified in a specific name, so generic names x and y are acceptable. 7:What are the fundamental parts of the method? A:The header of the method and its methodical body, which is enclosed by curly braces. 8:Why is MoveLeft a bad name for class? Which C# construct would you be better suited for? A:Classes generally represent subjects (tangible or conceptual) so nouns make better names for classes than verbs. Verbs, such as MoveLeft, are more suitable for methods taken by actions. 9:How do I specify that the method does not return a value? A:Write a keyword blank before the method identifier in the method header. 10:How do I specify that the method returns an int? A:Write a keyword before the method identifier in the method header. 11:What are arguments (by call method)? A:The value of the method argument is transferred to the called method. 12:What are the formal parameters? A:The formal parameter is part of the method header and has a certain value as the method begins to run. 13:How do arguments and formal parameters relate? A:Each method dial argument is assigned to the appropriate formal parameter during the method. 14:Does + operator only perform arithmetic additives? A:It has the potential (operator overload) to perform many different operations, depending on the types involved. We have seen examples of arithmetic addition and concatedranation of the string. 15:How can you break down the inner complexities of class? A:Separating several complex actions in a few simpler actions (method). 16:What is cohesive A:A method that accomplishes one clear task. Responses to Chapter 4 1 program exercises: Change comments in multiple lines in rows 3 6 to two single-line comments. 2:In addition to adding and multiplying, allow the user to perform a subtraction. Among other changes, you must add a subtraction method. (The symbol is used to perform subtractions in C#). A:The following program contains responses to exercises 1 and 2. Parts associated with Exercise 1 and Exercise 2 are marked ex.1 and ex.2, respectively. using the system; Ex. 1 //This class finds the sum, product, difference, //min and a maximum of two public class numbers SimpleCalculator { public static gap Main() { int x; int y; Console.Write(Enter the first number:); x = Convert.ToInt32 (Console.ReadLine()); Console.Write(Enter another number:); y = Convert.ToInt32 (Console.ReadLine()); Console.WriteLine(Sum is: + Sum(x, y)); Console.WriteLine(Product is: + Product(x, y)); Console.WriteLine(Maximum number is: + Math.Max(x, y)); Console.WriteLine(Minimum number is: + Math.Min(x, y)); Console.WriteLine(The difference is: + Subtraction(x, y)); Ex. 2 } // Sum calculates the sum of two int public static int Sum(int a, int b) { int sumTotal; sumTotal = a + b; return sumTotal; } // The product calculates the product of two int-int public static int products (int a, int b) { int productTotal; productTotal = a *b; return productTotal; } // Ex.2 // Subtract calculates the difference between two int public static int Subtract (int a, int b) { int difference; difference = a - b; return difference; } 3:Instead of calculating the amount and product of two numbers, let the program perform calculations on three numbers. (Here you can ignore the functions of Max and Min.) Tip: You need to declare another int variable in Maine. Sum, product, and subtraction methods must accept three arguments instead of two. You must allow the user to enter a third number, and you must include the third argument when these methods are called. 4:Create two methods called MyMax and MyMin that both take three arguments and find the maximum and minimum value of these arguments. Tip: Math.Max(Math.Max(a, b),c) returns the maximum a, b, and c. O:The following program presents responses to exercises 3 and 4. Enables calculations on three values and implements MyMax and MyMin methods: system usage; This class finds the sum, product, difference, //min and maximum of three public class numbers SimpleCalculator { Public Static Void Main() { int x; int y; int z; // Ex.3 Console.Write(Enter first number:); x = Convert.ToInt32 (Console.ReadLine()); Console.Write(Enter another number:); y = Convert.ToInt32 (Console.ReadLine()); Ex. 3 Console.Write(Enter a third number:); z = Convert.ToInt32(Console.ReadLine()); Console.WriteLine(Sum is: + Sum(x, y, z)); Console.WriteLine(The product is: Product(x, y, z)); Ex. 4 Console.WriteLine(Maximum number is: + MyMax(x, y, z)); Console.WriteLine(Minimum number is: + MyMin(x, y, z)); Ex.3 Console.WriteLine(The difference is: + Subtraction(x, y, z)); } Ex. 3 // Sum calculates the sum of two int public static int Sum (int a, int b, int c) { int sumTotal; sumTotal = a +b + c; return sumTotal; } // Ex. 3 // The product calculates the product of two int public static int products (int a, int b, int c) { int productTotal; productTotal = a * b * c; return productTotal; } // Ex. 3 // Subtract calculates the difference between two int public static int Subtract(int a, int b, int c) { difference in intu; difference = a - b - c; feedback difference; } //Ex. 4 // Find the maximum value of three parameters public static int MyMax(int a, int b, int c) { int max; max = Math.Max(a, Math.Max(b, c)); return maximum; } // Ex. 4 // Find min value three parameters public static int MyMin(int a , int b, int c) { int min; min = Math.Min(a, Math.Min(b, c)); return min; } Page 6 1: What is lexical analysis? A:Lexical analysis is a process that a compiler performs to distinguish atomic parts (in the source code of a computer language) from each other. 2:What are the atomic parts of the C# program? A:Atomic C# parts make up unscruspecting parts, such as keywords, identifiers, curly braces, parentheses, semicolons, and so on. 3:What is Pascal housing and camel housing? For which parts of the C# program should be used? A: If the identifier is written in Pascal enclosure, all words within this name begin with a large character, as in CalculateMaxDistance. The camel housing is almost identical to pascal housing, and the only difference is that the identifiers written with the camel housing do not contain the upper letter in the first word. numberLeactors are written with camel casing. Pascal housing should be used for class and methods of identifiers and camel housing for variables. 4:What is the main difference between variables and literal ones? A:Literal has a fixed value that is written in the source code. For example, a literal 10 always has a value of 10. In contrast, a variable called myNumber can represent different values. 5:What are operators and operands? How do they relate? A:Operators operate on operands. 6:Is 50 an expression? Is it (50+ x)? Is it public? A:50 and (50+ x) are both expressions, but the keyword public is not an expression. 7:Write examples of typical keywords in C#. A:if, publicly, privately. 8:Why isn't the pseudocode well suited to express the overall design of the object-oriented program? A:Pseudocode is very solving to express a set of detailed actions (algorithms) that must be performed sequentially to solve the task. However, classes are released from this sequential execution pattern. This makes the pseudocode incapable of describing the design of the overall class. 9:What kind of relationship it makes property to have with a BankTeller facility? How is this expressed in UML? A:Associational relationship, which is illustrated in UML with a simple line. 10:What kind of relationship hearts have with HumanBody. How is this expressed in UML? A:Compositional relationship, which is illustrated in UML by a line that has a black diamond attached. 11:What kind of relationship does the Lamp with the object of the lamp have. How is this expressed in UML? A:Aggregation bond, which is illustrated in UML by a line that has a white diamond attached. 12:How is the relationship of the association carried out in C#? A:Through instances of variables related to other objects. 13:How can I initialize instances of a variable when I create an object? A:Instances of variables can be initialized as part of their declaration of declaration or in the constructor. 14:Describe the passenger when declared as in the following line: private passenger; A: A random variable traveler may have a reference to a person's object. Responses to programming exercises in chapter 5 1: Print Simulation began on the command console immediately after the program began. 2:Print Simulation ended up being the last thing just before the program stopped. 3:Instead of just choosing floors between 1 and 30, the Person class selects floors between 0 and 50. 4:On its first ride the lift starts on floor number 0 instead of floor number 1. 5:Elevator facility makes 10 trips instead of the 5 it now does. 6:The elevator currently counts the total access floors with the total VariableFloorsTraveled. Declare the instance variable within the Elevator class that tracks the number of rides this elevator completes. You can call this variable totalTripsTraveled. The elevator should update this variable by adding one to the totalTripsTraveled after each trip. Update the ReportStatistics method of elevator class print not only totalFloorsTraveled, but also totalTripsTraveled, with an explanation for what is printed. A:The following program contains responses to all six program exercises presented in Chapter 5. Lines that are added or changed compared to chapter 5.1 because of the exercise are marked with a comment. For example, the line affected by Exercise 1 is marked //Ex.1. Easy simulation of elevators using the system; Elevator class { private int currentFloor = 0; //Ex.4 private int requestedFloor = 0; private int totalFloorsTraveled = 0; private int totalTripsTraveled = 0; //Ex.6 private Person passenger; public void LoadPassenger() { passenger = new Person(); } } public void InitiateNewFloorRequest() { requestedFloor = passenger. NewFloorRequest(); Console.WriteLine(Departing floor: + currentFloor + Trip to the floor: + requestedFloor); totalFloorsTraveled = totalFloorsTraveled + Math.Abs (currentFloor - requestedfloor); = totalTripsTraveled + 1; Ex.6 currentFloor = requestedfloor; } Public blank ReportStatistic() { Console.WriteLine(Total floors traveled: + totalFloorsTraveled); Console.WriteLine(Total trips traveled: + totalTripsTraveled); Ex.6 } } } class Person { private System.Random RandomNumberGenerator; public Person() { randomNumberGenerator = new System.Random(); } public int NewFloorRequest() { // Return of randomly generated randomNumberGenerator.Next(0.50) return number; //Ex.3 } } class Building { private static ElevatorA; public static void Main() { Console.WriteLine (Simulation started); //Ex.1 elevatorA = new elevator(); elevatorA.LoadPassenger(); elevatorA.InitiateNewFloorRequest(;); elevatorA.InitiateNewFloorRequest(); elevatorA.InitiateNewFloorRequest(); elevatorA.InitiateNewFloorRequest(); elevatorA.InitiateNewFloorReque (); elevatorA.InitiateNewFloorRequest();//Ex.5 elevatorA.InitiateNewFloorRequest();//Ex.5 elevatorA.InitiateNewFloorRequest()/);/Ex.5 elevatorA.InitiateNewFloorRequest();//Ex.5 elevatorA.InitiateNewFloorRequest();//Ex.5 elevatorA.ReportStatistic(); Console.WriteLine(Simulation completed); Ex.2 } Page 7 1: Why are there two ways to categorize types in C#? A:The difference of simple species in relation to derived species is used for historical reasons and because it is appropriate, but strictly speaking is not correct. Unlike Value Type relative to reference types it is accurate, but perhaps less intuitive. 2:Why does C# have other simple types besides int? A:Different types of data are required to represent different types of data. 3:What types of attributes distinguish different types from each other? A:Types of data that the type can represent, memory usage, the type of operations they will participate in, and in case it can hold simple numbers: range and precision. 4:What does it mean to be a strongly typed language like C#? A:Strongly typed language tries to prohibit types from being involved in processes in which they are not intended to participate. 5:Can I assign a short type value to an int variable without explicit conversion? Why or why not? A:No, because assigning type values to an int variable type of short potentially results in a loss of size. Any to-do with a potential risk of data loss (size or precision) requires explicit conversion. 6:How to determine literally be the type to float? A:So you add f (or F) to the number. So 6.89f is a literal cork type. 7:Write a code line that declars a private instance variable called distance, which is of the float type and is initialized to 100.5. A:private floating distance = 100.5f;. 8:The byte type variable holds the value 255. The program is now trying to improve the value by one. What happens: If the compilation switch is set to unverified? If it's set to be checked? A:If the Builder is set to unverified, variable overflow to make its value equal to 0. If it is set to check for an error in the time of operation. 9:What simple type would you use for an arbitrary person's weight storage variable (in pounds)? (1 pound is about 0.45 kilograms)? A:Simple guy floats. 10:What simple type would you use to store account balances that would be included in interest rate calculations that require extremely high precision? A:Simple decimal type. 11:Let's assume myNumber and YourNumber are both adrift. Write a statement that adds these values together as ints and assigns them to an int variable. A:myIntValue = (int)myNumber + (int)yourNumber;. 12:If number1 and number2 type int and text1 is string type, what is the type of next expression? number1 + text1 + number2 A:Expression is string type. 13:Is the following phrase true or false when evaluated in C# and why? ((10 * 0.2f) == 2.0) A:0.2f isn't exactly 0.2. This inaccuracy is exposed when (10 * 0.2f) is compared to the value of a double type, such as 2.0, resulting in a false expression. 14:When should you use constants in your source code? What are their advantages? A:Use constants when the source code contains values that are constant for the duration of the program. The benefits of using constants are easier to understand the meaning of a name like Pi instead of a literal one like 3:14. Using constants allows you to change the value in one place (where it is declared) and not in every place where it is used in the source code. 15:Write a statement that prints 3000000.326m on screen with this format: $30,000,000.33 A:Console.WriteLine (300000000.326m. ToString(C)). Responses to chapter 6 of Program Exercise 1: Allow the clock to adjust by plus/minus 100 seconds by giving commands H and M. 2:A per year on Blipos has 256 days. The first day of the year has the No. Civilization is very old on Blipos; their current year stands at 4,294,967,296. Allow Blipos watch to show the date on Blipos as ddd/y,y, yyyy,yyyy. What types would you choose for ddd and yyyyyyyy? 3:Allow the user to set and adjust the day and year of the Blipos watch so that it becomes possible to move back and forth in time for one day and one year. 4:The Emperor's birthday is marking the 100th anniversary of the 1994-1995 world. Allow Blipos Watch to write Happy Birthday to the Emperor! on the screen whenever the date appears to be 100. A:The following source code provides responses to program exercises 1 4. The code parts associated with a particular exercise are marked with a comment, such as // Ex.1 or // Exercise 1. using the system; class BliposClock { private byte seconds; private short minutes; //Exercise 2 private byte days; private ulong years; public BliposClock() { seconds = 0; minutes = 0; // Exercise 2 days = 0; year = 0; } public emptiness { byte originalSeconds = sekunde; kratki originalMinutes = minute; //Ex. 2 byte originalDays = days; //Ex. 2 seconds++; if (originalSeconds &gt; seconds) // Overflow of seconds variable minutes++; // Exercise 2 if (originalMinutes &gt; minutes) // Overflow of minutes variable days++; if (originalDays &gt; days) // Overflow of days variable years+; } public void OneBackward() { byte originalSeconds = seconds; short originalMinutes = minutes; //Ex. 2 byte originalDays = days; //Ex. 2 seconds--; if (originalSeconds &lt; seconds) // Underflow of seconds variable minutes--; // Exercise 2 if (originalMinutes &lt; minutes) // Underflow of minutes variable days--; if (originalDays &lt; days) // Underflow of days variable years--; } public void FastForward() { byte originalSeconds = seconds; short originalMinutes = minutes; //Ex. 2

byte originalDays = days; //Ex. 2 seconds = (byte)(seconds + 50); if (originalSeconds &gt; seconds) // Overflow of seconds variable minutes++; // Exercise 2 if (originalMinutes &gt; minutes) // Overflow of minutes variable days++; if (originalDays &gt; days) // Overflow of days variable years++; } public void FastBackward() { byte originalSeconds = seconds; short originalMinutes = minutes; //Ex. 2 byte originalDays = days; //Ex. 2 seconds = (byte)(seconds - 50); if (originalSeconds &lt; seconds) // Underflow of seconds variable minutes--; // Exercise 2 if (originalMinutes &lt; minutes) // Underflow of minutes variable days--; if (originalDays &lt; days) // Underflow of days variable years--; } // Exercise 1 public void HundredForward() { byte originalSeconds = seconds; short originalMinutes = minutes; //Ex. 2 byte originalDays = days; //Ex. 2 seconds = (byte)(seconds + 100); if (originalSeconds &gt; seconds) // Prelijevanje sekundi varijabilnih minuta++; Vježba 2 ako (originalMinutes &gt; minuta) // Prelijevanje minuta promjenjivih dana++; ako (originalDays &gt; dana) // Prelijevanje dana promjenjivih godina++; } // Vježba 1 javna praznina HundredBackward() { byte originalSeconds = seconds; short originalMinutes = minutes; //Ex. 2 byte originalDays = days; //Ex. 2 seconds = (byte)(seconds - 100); if (originalSeconds &lt; seconds) // Underflow of seconds variable minutes--; // Vježba 2 ako (originalMinutes &lt; minuta) // Underflow minuta promjenjivih dana--; ako (originalDays &lt; days) // Underflow of days variable years--; } // Exercise 3 public void AddOneDay() { byte originalDays = days; days+; ako (originalDays &gt; days) // Overflow of days variable years++; } // Exercise 3 public void DeductOneDay() { byte originalDays = days; days--; if (originalDays &lt; days) // Underflow of days variable years--; } // Exercise 3 public void AddOneYear() { years++) { ; } // Vježba 3 javna praznina DeductOneYear() { years--; } javna praznina SetSeconds(byte sec) { seconds = sec; } javna praznina SetMinutes (kratka min) { minute = min; } Vježba 2 javna praznina SetDays(byte newDays) { days = newDays; } // Exercise Exercise public void SetYears(ulong newYears) { years = newYears; } public void ShowTime() { Console.WriteLine(Sec: + seconds + Min: + minutes); Exercise 2 Console.WriteLine(Day/year: + days + / + years); Exercise 4 if (days == 100) Console.WriteLine (Happy Birthday Emperor!); } RunBliposClock class { Main public static gap() { string command; Console.WriteLine(Welcome to Blipos watch. + 256 seconds per minute + 65536 minutes per day); BliposClock myClock = new BliposClock(); Console.WriteLine(Please set the clock); Console.Write(Enter seconds:); myClock.SetSeconds (Convert.ToByte(Console.ReadLine())); Console.Write(Enter minutes:); myClock.SetMinutes (Convert.ToInt16 (Console.ReadLine())); Console.Write(Enter Days:); Ex.2 myClock.SetDays (Convert.ToByte(Console.ReadLine())); Ex.2 Console.Write(Enter Years:); Ex.2 myClock.SetYears (Convert.ToUInt64(Console.ReadLine()); Ex.2 Console.WriteLine (Enter command: F) Forward B) Backward + A) Add fifty); Console.WriteLine(D) Deduct fifty H) Add hundred + //Ex.1 M) Deduct hundred); Ex.1 Console.WriteLine (Q) Add One Day Y) Add One Year + //Ex.3 U) Back Off One Day W) Back Off One Year T) Stop); Ex.3 to { command = Console.ReadLine(). ToUpper(); if (command == F) myClock.OneForward(); if (command == B) myClock.OneBackward(); if(command == A) myClock.FastForward(); if(command == D) myClock.FastBackward(); Exercise 1 if(command == H) myClock.HundredForward(); if(command == M) myClock.HundredBackward(); Exercise 3 if(command == Q) myClock.AddOneDay(); if(command == Y) myClock.AddOneYear(); if(command == U) myClock.DeductOneDay(); if(command == W) myClock.DeductOneYear(); myClock.ShowTime(); } while (command != T); Console.WriteLine(Thank you for using the Blipos watch); } } 5:Write a program that contains a method called AverageAge that calculates the average of the three ages provided to it as arguments. Allow the user to enter three ages (as an integer) and use AverageAge to calculate the average age and print the results with a precision of three decimal places. What type of return would you use for AverageAge? How can you format the output? A:The following source code is a response to exercise 5. using the system; AgeCalculator class { public double AverageAge (age by number of years, age bytes) { double average; average = (double)(age1 + age2 + age3) / 3; return average; } } Tester class { public static gap Main() { byte ageA; byte ageB; byte ageC; AgeCalculator myCalculator = new AgeCalculator(); Console.Write(Enter age 1:); agea = Convert.ToByte (Console.ReadLine()); Console.Write(Enter age 2:); ageb = Convert.ToByte(Console.ReadLine()); Console.Write(Enter age 3:); agec = Convert.ToByte(Console.ReadLine()); Console.WriteLine(Average age: + myCalculator.AverageAge (ageA, ageB, ageC). ToString (N3)); } Page 8 are the following terms evaluated in C#: A: 5 + (10 * 2) = 5 + 20 = 25 (5 * 6) / 3 = 30 / 3 = 10 (12 / 4) * 6 = 3 * 6 = 18 MyIntVariable value is incrementative for one. If myIntVariable++ is part of a longer expression, these steps will take place after other operations in the expression. The value of myIntVariable is decontamentary by one. If -myIntVariable is part of a longer expression, this absophrase will occur before other operations in the expression. 2:Suppose you need to sum number1 and number2 and multiply this result by number3. First attempt: number one + number2 * number3 gave the wrong result. Solve the problem. A:Add a pair of parentheses, as in the following row: (number one + number2) * number3 3:The following expression is correct, but not clear for the code reader. Let's make it clearer. num1 + num2 / num3 * num4 num5 * num6 / num7 A: (num1 + ((num2 / num3) * num4)) - ((num5 * num6) / num7) 4:Write if a statement that checks whether the number (call it myNumber) type int is even or strange. Only if myNumber should have written the number at all is even on the screen. A: if ((myNumber % 2) == 0) Console.WriteLine 5:What is the type of next expression if the weight is of a short type? - Weight A:It's type int. 6:Improve the clarity of the following statement by separating it into three statements without altering the effects of the code: totalBacteria = ++bacteriaInBody1 + bacteriaInBody2++; A: BacteriaInBody1++; totalBacteria = bacteriaInBody1 + bacteriaInBody2; bacteriaInBody2++; 7:Draw a flowchart that illustrates the following logic. A person wants to go for a walk. If it rains, he asks his wife if they have an umbrella. If they have an umbrella he'll come out; otherwise it will stay inside. If it doesn't rain, they'll come out whether they have an umbrella or not. A: 8:What construct would be suitable for presenting the following values in the C# program: { Red, Green, Blue, Yellow, Purple} A:The enumerator construct. 9:Write a statement that prints the following text on the console (including quotation marks)? And then he said, This is the big moment A: Console.WriteLine. This is a big moment\ ); 10:Let's say myString contains 20 characters. Write an expression that returns a sub-string from myString starting with 10 characters and including 5 characters. A: myString.Substring (9.5) 11:Suppose distance1 is 100, distance2 is 200, and distance3 is 400. Write a statement that uses built-in formatted numbers to include three variables in a string so that the final text reads First distance is 100 meters, the second distance is 200 meters, and the third is 400 meters. A: Console.WriteLine(The first distance is {0} meters, the second distance is {1} meters, and the third distance is {2} meters, distance1, distance2, distance3); Reply to Chapter Program exercises 1: In addition to reporting every Sunday, they should also occur every Saturday. Tip: At this time, the program detects every 7, 14, 21, 28 and so on. What numbers should be revealed to find all Saturdays? That would be six, 13, 20, 27, and so on. How do they relate to Sunday numbers? How can this be implemented? 2:Let's say there are 365 days in the simulation every year. Change the program to allow it to report running time in days, weeks, and years, instead of just the days and weeks currently logged in. A:The following source code contains responses to exercises 1 and 2. The relevant parts are marked // Exercise 1 or // Ex. 1 or so. using the system; klasa DayCounter { javna statička praznina Main() { uint dayCounter = 0; uint maxSimulationDays; uint weeks; uint years; // Exercise 2 byte remainderDays; byte remainderDaysSaturday; //Exercise 1 Console.Write(Please enter the number of days + simulacija bi trebala raditi za); maxSimulationDays = Convert.ToUInt32 (Konzola.ReadLine()); dok (danCounter &lt; maxSimulationDays) { dayCounter++; weeks = dayCounter / 7; years = weeks / 52; //Vježba 2 remainderDays = (byte)(dayCounter % 7); //Exercise 1 remainderDaysSaturday = (byte)(dayCounter + 1) % 7); //Exercise 2 Console.WriteLine(Weeks: {0} Days: {1} Years: {2}, weeks - (years * 52), remainderDays, years); if(remainderDays == 0) // TODO send it's Sunday poruka kontroloru Console.WriteLine(\t\tHey Hey It's Sunday!); //Vježba 1 ako(remainderDaysSaturday == 0) Console.WriteLine(\t\tHey Subota je); // TODO simulacija početka u trajanju od jednog dana. // Neka program pauzira za 200 milisekundi system. Threading.Thread.Sleep(200); Console.WriteLine(Simulation completed); } } 3:Remember the Blipos Clock case study in Chapter 6, Type I: Simple Types. Instead of relying on the overflow/lower flow mechanism of integer types, rewrite the BliposClock class to use the operator module to perform exactly the same tasks. A:The following source is a response to exercise 3: using the system; Class BliposClock { //A time in BliposClock is represented by only one value: totalSeconds. //All times shown are derived from this baseline value. private ulong totalSeconds; public BliposClock() { totalSeconds = 0; } public void AddSeconds (ulong secondsToAdd) { totalSeconds = totalSeconds + secondsToAdd; } Public Gap Deductions (ulong secondsToDeduct) { totalSeconds = totalSeconds - secondsToDeduct; } //initialMinutes can be between -32768 and 32767 so that to convert this value //to totalSeconds we must add 32768 and multiply by 256 before you add this //value to the initialsSeconds as shown in the following method statement. public blank SetTime (byte initialSeconds, short initialMinutes) { totalSeconds = (ulong)initialSeconds + (ulong)((initialMinutes + 32768) * } javna praznina ShowTime() { byte byte int secondsPassedBy; The total number of seconds per day is a short minuteItShow; secondsToShow = (byte)(totalSeconds % 256); secondsPassedBy = (int)(totalSeconds % (256 * 65536)); By dividing thepassedBy seconds by 256 in the next row, we find that //number of full minutes is a day old. This number can be //between 0 and 65535. However, we need the number // between -32768 and 32767, which we can get //subtract 32768 as shown in the following reto. minuteToShow = (short) ((secondsPassedBy / 256) - 32768); Console.WriteLine(Sec: {0} Min: {1}, secondsToShow, minutesToShow); } RunBliposClock class { Main public static gap() { string command; byte tempSeconds; short tempMinuti; Console.WriteLine(Welcome to Blipos watch. + 256 seconds per minute + 65536 minutes per day); BliposClock myClock = new BliposClock(); Console.WriteLine(Please set the clock); Console.Write(Enter seconds:); tempSeconds = Convert.ToByte(Console.ReadLine()); Console.Write(Enter minutes:); tempMinutes = Convert.ToInt16 (Console.ReadLine()); myClock.SetTime (tempSeconds, tempMinutes); Console.WriteLine(Enter (F)orward ackward + (A)dd fifty educt fifty (T)erminate); up to { command = Console.ReadLine(). ToUpper(); if (command == F) myClock.AddSeconds(1); if (command == B) myClock.DeductSeconds(1); if(command == A) myClock.AddSeconds(50); if(command == D) myClock.DeductSeconds(50); myClock.ShowTime(); } while (command != T); Console.WriteLine(Thank you for using the Blipos watch); } Page 9 1: What general type of statement would you use to implement each of the following logical descriptions: If the number is greater than 100, write more than 100 on the screen; otherwise, write less or equal to 100. Check each letter of the string repeatedly until there are no more letters left in the string. A: Statement of branched if-other. 2:Suppose your program contains three variables called precipitation, wind and temperature, all types of double. For each of the following points, write a few lines of code that implement the described logic (you do not need to write the entire program): If the rainfall is greater than 100, write Heavy Rainfall on the screen. If precipitation is greater than 100 or if the wind is more than 120, write Bad Weather on the screen. If the amount of precipitation is equal to 0 and the wind is less than 10, and the temperature is between 23 and 27 (Celsius), write Good weather. If precipitation is equal to 0, write Do not rain; otherwise, write Raining. A: if(rainfall &gt; 100) Console.WriteLine(Heavy rainfall); if((precipitation &gt; 100) || (wind &gt; 120)) Console.WriteLine(Bad weather); if(rainfall == 0) &amp;&amp; (wind &lt; 10) &amp;&amp; (temperature &gt;= 23) &amp;&amp; (temperature &lt;= 27)) Console.WriteLine(Good weather); if(precipitation == 0) Console.WriteLine(It's not raining); other Console.WriteLine(It's raining); 3:Regardless of rainfall value, the following lines will always say rain on the screen. Why? if (precipitation &gt; 10); Console.WriteLine(It's raining); A: A half-gallon after the first row represents a blank statement, so if the statement can also be written as follows. If precipitation is greater than 0, an empty statement is executed; otherwise not. The last sentence is always executed. if (precipitation &gt; 0) ; Console.WriteLine(It's raining); 4:What is the point of nesting if statements? A:This allows the program to choose from more than two alternate branches. 5:Write logical expressions (using C#'s comparison operators and logical operators) representing the following conditions: the amount of precipitation is between 100 and 150. the number is odd and not equal to 23. the number is even less than 100, or the weight is less than 100. A: ((precipitation &gt; 100) &amp;&amp; (precipitation &lt; 150)) ((number % 2 != 0) &amp;&amp; (number != 23)) ((number % 2 == 0) || (weight &lt; 100)) 6:The method with the bool IsEven header (int number) returns true if the argument is passed to the number of parameters anew; otherwise, it returns falsely. Write a Boolean phrase that contains a call to IsEven that is true if the number was passed along with the call strange. A:! (Iseven(someNumber)). 7:What does the term scope refer to? A:Scope refers to the source code section where a specific variable identifier can be used. The scope of the variable begins where it is declared and ends where the block in which it was declared stops. The scope includes blocks that could nest within this block. 8:What is one of the few correct uses of goto statements? A:Direct the control to the second part of the switch in the switch statement. 9:Rewrite the following multibranch if-other statement using the switch statement. if(timeOfDay == morning) Console.WriteLine(Good morning); secondly if (timeOfDay == midday) Console.WriteLine(Good afternoon); secondly if(timeOfDay == evening) Console.WriteLine(Good evening); other Console.WriteLine(Invalid Time); A: switch(timeOfDay) { case morning: Console.WriteLine(Good morning); pause; case noon: Console.WriteLine(Good day); pause; case evening: Console.WriteLine(Good evening); pause; default: Console.WriteLine(Invalid time); pause; } 10:What does the drop refer to in relation to the switch statement? How can it be prevented from going to waste? A: Falling is when the execution flow moves from one part of the switch directly to the next part of the switch. This can be prevented by a pause or goto statement. 11:Your program contains variables called cost1, cost2 and minimumCost. Write a statement that includes a conditional operator that assigns lower values in cost1 and cost2 to a minimumCost. A:minimumCost = (cost1 &lt; cost2) ? cost1 : cost2; Responses to Chapter 8 program exercises 1:The currency on the planet Blipos is called sobbing. Foreigners on Blipos pay the following amount of tax depending on income: 0% less than or equal to 10000 sopilbs 5% of income that is greater than 10000 and less or equal to 25000 sopilbs 10% of income that is greater than 25000 and less than or equal to 50000 sopilbs 15% of income that is greater than 50000 and less than or equal to 100000 sopilbs 20% of income that is greater than 100000 sopilbs So if a stranger earns 60000 sobs, will have to pay the following amount of tax: 0 * 10000 + 0.05 * 15000 + 0.1 * 25000 + 0.15 * 10000 = 4750. Write a program that accepts a specific income as input and calculates and exits the tax to be paid from that amount. A:Exercise 1: using the system; TaxCalculator class { Main public static gap() { double income, double tax; Console.Write(Enter revenue:); revenue = Convert.ToDouble(Console.ReadLine()); if (tax &lt;= 10000) = 0; secondly if((tax &gt; 10000) &amp;&amp; (&lt;= 25000)) tax = (income - 10000) * 0.05; secondly if((income &gt; 25000) &amp;&amp; (income &lt;= 50000)) tax = (15000 * 0.05) + ((income - 25000) * 0.1); secondly if((income &gt; 50000) &amp;&amp; (income tax &lt;= 100000)) = (15000 * 0.05) + (25000 * 0.1) + ((income - 50000) * 0.15); second tax = (15000 * 0.05) + (25000 * 0.1) + (50000 * 0.15)+ ((income - 100000) * 0.2); Console.WriteLine(Tax is: {0}, tax); } } 2:Write a program (using the switch statement) that accepts one of the exam results A, B, C, D and E. In response, the program will receive an appropriate percentage rating, namely A: 90 100; B: 80 89; C: 70 79; D: 60 69; E: 0 59. A:Exercise 2: using the system; Class ExamScoreKonverter { public static gap Main() { string result; Console.WriteLine(Enter the result you want to convert); score = Console.ReadLine(). ToUpper(); switch(score) { case A: Console.WriteLine(90-100 percent score); break; case B: Console.WriteLine(80-89 percent score); break; case C: Console.WriteLine (70-79 percent results); pause; case D: Console.WriteLine(60-69 percent results); pause; case E: Console.WriteLine(0-59 percent results); pause; default: Console.WriteLine(Invalid exam result); pause; } 3:Write a program that can find a number between 0 and 100. The user decides on the number and the program repeatedly guesses and constantly narrows the number until the number is found. Tip: Have your computer hit a number in the middle of a possible maximum number and possible minimum number. For example, the first assumption should be 50. If the user indicates the number to be higher, the next hit should be 75, and so on. A:Exercise 3: using the system; NumberFinder class { Main public static gap() { int maxNumber = 100; int minNumber = 0; bool found = false; intCount guesser = 0; string reply; do { guessCounter++; Console.WriteLine(My opinion is: {0}, ((maxNumber + minNumber) / 2)); Console.WriteLine(Enter H for H)igher or L for L)ower C for C)orrect?); answer = Console.ReadLine(). ToUpper(); if(answer == H) minNumber = (maxNumber + / 2; second if (answer == L) maxNumber = (maxNumber + minNumber) / 2; other found = truth; } until (!found); Console.WriteLine(I found a number in {0} hits, guessCounter); } Page 10 1: What is the main difference between time and time statements? Why is the statement the most common preferred type of statement? A:Input Status Statement is a status input loop statement; do-while is a statement output loop state. The input condition loop may never be executed, while the status will be done until-while executed at least once. The input state loop is better suited for generating and moving data series because the expected data set may be empty or may not need to generate data in a specific program flow. 2:What is the output from the next part of the source code? (Mentally monitor code.) inta counter = 0; while (counter &lt; 4) { counter++; Console.Write(counter +); } A: 1 2 3 4 3:What is the output from the next do-while construct? Trace the code. int counter = 4; up to { Console.Write(counter +); counter--; } while(counter &gt; 0); Console.WriteLine(Counter value: {0}, counter); A: 4 3 2 1 Counter value: 0 4:What is the output from the following to build a loop? (Track the code.) for(int i = 0, and &lt; 0, and += 2) { Console.Write(i +); } A: 0 2 4 6 5:Write the loop generating the following output: 12 9 6 3 0 -3 A: za(int i = 12; i &gt;= -3; and -= 3) { Console.Write(i +); 6:Is a valid following for the loop (neglecting not to be displayed as part of the complete program). If so, what exit will it generate? for(int i = 1, j = 1, i &lt; 5, i++, j += 2) { Console.Write({0} , i * j); } A: It's worth it. The output is as follows: 1 6 15 28 7:What is the output from the next nested loop pair? (Track source code.) for(int i = 1, i &lt; 4, i++) { for(int j = 1, j &lt; 5, j++) { Console.Write({0} , i + j); } A: 2 3 4 5 3 4 5 6 4 5 6 8:Consider the following for loop: za(int counter = 1; ; counter++) { Console.WriteLine(Counter value: {0}, counter); Console.WriteLine(This counter value * 10: {0} , counter * 10); } Syntactic, this applies to the loop. However, it is an infinite loop as it stands now, because it has no state of the loop. Change the loop so that on your fourth repeat loop it ends between the two writeline statements. So, on its fourth repeated loop, one only needs to write the value of the counter: 4, and then break. two writeline statements. Executes whenever the counter is greater than 3. for(int counter = 1; ; ; ; counter++) { Console.WriteLine (Counter Value: {0}, Counter); if(counter &gt; 3) break; Console.WriteLine(Counter value * 10: {0}, counter); } 9:Consider the following for the loop shown above in list 9.8: for(int = 0; and &lt;= 20; i++) { if(i % 4 == 0) continue; Console.Write(i + ,); } Rewrite the loop and eliminate the continuation statement, keeping the loop semantics intact. A: for(int i = 0, and &lt;= 20, i++) { if(!) and % 4 == 0)) Console.Write(i + ,); } 10:Specify one advantage of pseudocoding over flowcharts over structured program design. A:The arrow in the flowchart represents the movement of the control from one part of the program to another. It's easy to draw arrows on a chart that goes anywhere. However, sometimes the only way this can be done is through an error-prone goto statement. On the other hand, the pseudocode format is very similar to C# source code (you write one line at a time and the lines are mostly executed sequentially) and is very compatible with compact structured C# constructions, such as loops, if statements and so on. Answers to Chapter 9 program exercises 1:Factorial is an important mathematical function. Factorial n is written n!. N! is calculated as a product of whole wholegers from 1 to n. For example, 4! = 4 x 3 x 2 x 1. Onwards 1! = 1 and 0! = 1. Write a program that receives an integer from a user, calculates its factor value, and returns that value as output. Allow the user (through the loop construct) to perform as many calculations as they want during one program. After each calculation, the program must ask the user if he wants to perform another calculation. A:Exercise 1: using the system; Class Calculator { Public Static Blank Main() { String Response; Long Factor; Int Number, up to { Console.Write(Enter number:); number = Convert.toint32(Console.ReadLine()); factor = 1; if(number &gt;= 0) { for(int i = number; and &gt;= 1, i--) { factorial *= and } other { factorial = -1; Console.WriteLine({0} factorial = {1}, number, factor); Console.WriteLine(Run another calculation of Y)es N)o); answer = Console.ReadLine(). ToUpper(); } until(reply != N); Console.WriteLine(Thank you for using the factor calculator); } 2:Write a program that generates the following output: * ** *** **** *** ** ** A:Exercise 2: using the system; StarTriangle class { Main public static gap() { for(int i = 1; i &lt;= 8, i++) { for(int j = 1, j &lt;= i; j++) { Console.Write(*); } Console.WriteLine(); } } } 3:Write a program that generates the following output: * ## ** ### ** ##*** ##*** ##* ## #A:Exercise 3: using System; MixedTriangles class { Main public static gap() { for(int and = 1; i &lt;= 8, i++) { for(int j = 1, j &lt;= i; j++) { Console.Write(*); } Console.WriteLine(); for(int k = 1, k &lt;= 9 - and; k++) { Console.Write(#); Console.WriteLine(); } } Page 11 1: Declare an array variable called distances with double-type elements. Assign a string object with 5 elements Assign a field object to distances 20.1, 30.7, 45.8, 19.1, 12.4, and 34.5. A: private double distances; distance = new double[5]; private double distances = {20.1, 30.7, 45.8, 19.1, 12.4, 34.5} ; 2:Consider the following string: private int[] numbers = { 10,5,3,15} ; What is the value of the next element of the field? numbers[1] What is the output from the following loop construct: for(int i = 3; i &gt;= 0; i--) { Console.Write({ 0} , numbers[i]) } Write a loop using the pre-ear construction to move and print on the screen of each element in numbers. This should result in the following output: 10 5 3 15. What is wrong with the following code trying to cross a series of numbers? What happens when it's done? for(int i = 0, and &lt;= 4, i++) { Console.WriteLine (numbers); A: Number value[1] is 5 foreach (int temp in numbers) { Console.Write({0}, temp); The loop repeats once too much, so I become larger than 3 in the last loop and causes IndexOutOfBoundsException. The loop state should also be &lt; 4 instead of &lt;= 4. 3:Class contains the following string declaration: private int[] = new int[3]; What is the value of each field element immediately after this string is implicitly initialized? The line in your class looks like the following heights [4] = 10; Is this a valid statement? What happens when this line is executed? A: Int field elements are implicitly initialized to 0. No, that's not a valid statement. Index 4 is out of range. When this line is executed, IndexOutOfRangeException will be generated. 4:Why is it not a good idea to use the series to represent the accounts of the real world bank? A:The bank has a constant need to delete and add new accounts. This requires a string-like structure that can dynamically add and delete new elements. Fields are not suitable for this task because they have a fixed length after they are created. 5:Consider the following string statement (same as in question 2): private numbers = { 10,5,3,15} ; A fellow programmer has written the following method to retrieve values from the field: int GetNumber(int index) { return numbers[index]; } Adjust to the zero-level field index so that GetNumber (1) returns the value of the first element of numbers instead of the value of the second element, as is the case with the method currently displayed. A: int GetNumber(int index) { return numbers[index - 1]; } 6:What's wrong with the following string statement? byte [] age = new byte [4] { 10, 34, 12, 19, 21,56} ; A:[4] (positioned after byte) states that field length 4. However, the same line tries to assign six values at a time; this difference is invalid. 7:Write a method called DisplayArray that accepts a series of base-type int references. The method must be able to print the values of the field object referenced on the screen, regardless of the length of the field. A: void DisplayArray(int[] tempArray) { for(int i = 0; i &lt; tempArray.Length; i++) { Console.WriteLine(tempArray[i]); } } 8:Write a method with the following header: int [] AddNumber(int [] tempArray, int num) that will add a number to each tempArray element and return this string back to the caller. A: int [] AddNumber(int [] tempArray, int num) { for(int i = 0; i &lt; tempArray.Length; i++) { tempArray[i] += num; } return tempArray; } 9:Consider the following two statements: int [] myNumbers = { 2,4,6,8} ; int [] yourNumbers; Suppose we assign myNumbers to yourNumbers and add 10 to the first element of yourNumbers, as in the following lines yourNumbers = myNumbers; yourNumbers[0] += 10; What is the value of myNumbers[0] after these statements have been made? Explain what's going on. A:myNumbers and yourNumbers refer to the same field object, so adding 10 to yourNumbers[0] is the same as adding 10 myNumbers[0]. As a result, myNumbers[0] is equal to 12. 10:What is the fundamental difference between cloning a field by the method of cloning a field and simply assigning a variable field value to another array variable? A:The Clone method creates a whole new field object with copies of field element values that are separated from those in the string by which the Clone method is called. In contrast, a simple assignment from one array variable to another passes only a reference, causing the two array variables to refer to the same array object. 11:Consider two series from Question 9, myNumbers and yourNumbers. Suppose each array variable refers to another field object. Both of these array objects contain exactly the same number of array elements and each pair of corresponding array elements has the same value. Will the following comparison be true or false? Explain why. (MyNumbers == YourNumbers) A:This is false because the comparison operator (==) tests whether comparative array variables refer to the same object (reference-based comparison). 12:Your program contains a class called Planet. You write a different class called SolarSystem, which in this case must consist of 10 planets. You want to represent 10 planets in a series of 10 planets. Write a declaration that you must insert into the SolarSystem class to enable 10 planets to be displayed. A: Planet [] Planets = New Planet[10]; Responses to Chapter 10 Program Exercises 1: Implement a class called ArrayMath, which contains the following methods for performing calculations on strings: A method called ArrayAverage that, as an argument, accepts a series of double base types (of any length) and returns the average of this field. A method called ArraySum that accepts two strings of base type int. Two fields must have an identical length. The method will sum each corresponding pair of field elements from the two strings. Each amount is assigned to the corresponding element of the third field, which is finally returned to the caller. A method called ArrayMax that finds and returns the maximum value in a string of basic int types. The array argument can be of any length. Write a test code for this class and its methods. A:Exercise 1: using the system; ArrayMath class { public static double ArrayAverage (double [] tempArray) { double sum = 0; foreach(double temp in temparray) { sum += temp; } return total / tempArray.Length; } public static int [] ArraySum(int [] tempArray1, int [] tempArray2) { int [] sumArray = new int [tempArray1.Length]; for(int i = 0; and &lt; tempArray1.Length; i++) { sumArray[i] = tempArray1[i] + tempArray2[i]; } return sumArray; } public static int ArrayMax(int [] tempArray) { int maxValue = -2147483648; foreach(int temp in temp in tempArray) { if(temp &gt; maxValue) maxValue = temp; } return maxValue; } Class Tester { Public Static Gap Main() { double [] distances = {100, 200, 300}; int [] agesTeam1 = {10, 20, 30}; int [] agesTeam2 = {34, 38, 31}; int [] sumArray; Console.WriteLine(Average distance: {0}, ArrayMath.ArrayAverage(distances)); Console.WriteLine;.Age In AgeTeam1 field: {0}, ArrayMath.ArrayMax(agesTeam1)); sumArray = ArrayMath.ArraySum (dobTeam1, agesTeam2); Console.WriteLine (sumArray's element values: {0} {1} {2}, sumArray[0], sumArray[1], sumArray[2]); } 2:Write the basic parts of the car game program. The program must include a car class with the following members: an instance variable called an int A type position with a public gap in the MoveForward (int distance) header that adds distance to the position of the A variable instance with the Reverse(int distance) public void header that takes away the distance from the position of the A method called GetPosition that simply returns the position value to the caller Onwards, the program must contain a class called CarGame that (using fields) contains 5 Objects of type Car. It must be possible to move each car (back and forth) and get the position of each of the cars by providing an index of fields of the corresponding car. Write a small test program to ensure that the two classes work properly. A:Exercise 2: using the system; Car class { private int position = 0; public void MoveForward (distance from inta) { position += distance; } public void Reverse(int distance) { position -= distance; } public int GetPosition() { return position; } } CarGame class { private car [] cars; public CarGame() { cars = new car [5]; for(int and = 0; and &lt; cars. Length; i++) { cars [i] = new car(); } public int GetCarPosition (int carIndex) { return cars[carIndex]. GetPosition(); } public emptiness MoveCarForward (int carIndex, int distance) { cars[carIndex]. (distance); } ReverseCar public gap (int carIndex, int distance) { cars[carIndex]. Reverse(distance); } CarGameTester class { public public void Main() { CarGame testGame = new CarGame(); testGame.MoveCarForward (0, 100); testGame.MoveCarForward (1, 40); Console.WriteLine(Car Position 0: {0}, TestGame.GetCarPosition(0)); Console.WriteLine (Car Position 1: {0}, testGame.GetCarPosition(1)); } Page 12 1: How many dimensions does the warpedWorld field have if access to one of its elements requires a line of code such as the following: warpedWorld[2,4,1,6,4] How many nested loops would it take to access all elements in the WarpedWorld series? A: 2:You are considering writing a chess program. How many dimensions should a string representing a chessboard have? A:2 dimensions. 3: Declare a rectangular three-dimensional network called observations with a basic type of uint. Assign it a reference to a new field object with the following element numbers in each dimension 5, 10, 20. Write a statement that assigns a value of 100 to the observation element with indices 3, 2, 10. What happens if you try to access the element with indices 1, 12, 14? Write nested for access and print loops on the screen the value of each element in a series of observations. Use the foreay loop to perform the same operation as d. O: uint [, ,] observations = new uint [5,10,20];. observations[3,2,10] = 100;. The second index (12) is out of range and will run running time to generate IndexOutOfRangeException. for(int i = 0, i &lt; 5, i++) { for(int j = 0, j &lt; 10, j++) { for(int k = 0; k &lt; 20; k++) { Console.WriteLine(i, j, k)); } foreach(uint temp in observations) { Console.WriteLine(temp); } 4: Declare a jagged two-dimensional string called base type int numbers, in which the first dimension contains 7 elements. Declare the third element of the first dimension containing a one-dimensional string of 20 lengths. Assign a value of 100 to the fourth element in the series contained in the third element of the first dimension. A: int [][] numbers = new int [7][]; numbers [2] = new int [20]; 5:Suppose the number name string is declared as follows: int [] numbers = {2, 4, 6, 8, 10, 12, 6} ; What values do the following method calls return? System.Array.IndexOf(numbers, 6) System.Array.IndexOf(numbers, 7) System.Array.IndexOf(numbers, 2, 3) System.Array.LastIndexOf(numbers, 6) System.Array.Reverse(numbers) followed by System.Array.IndexOf(numbers, 12) A: 6:Suppose your program searches a string containing one million data items using a binary search algorithm. How many data items does it discard (and don't have to re-view) from search during the first search loop? A:Half of a million: 500000 7:Your program should look for a value in a series of lengths in 2048. What maximum number of loops is required to find values using A: If sequential search starts at the beginning of the field and the key value is at the end of the field. the maximum number of loops required in 2048. After each binary search loop, half of the previous segment of the field is eliminated. Consequently, a maximum of 11 searches are required, as 2048 is divided by 2 11 times equal to 1. In other words, 211 = 2048. Answers to Chapter 11 1 program exercises: Sometimes the number of cars passing a specific point on the highway should be counted. Suppose a person should count the cars that pass every hour during one full week. Write a program that can help that person store, retrieve, and analyze this information. In particular, the program must be able to store car number entries per hour 24 hours a day and 7 days. Have the user enter a specific number specifying the day, hour, and count. Have the user reach a specific number by specifying the day and hour. Calculate and show the total number for the entire week. Specify the number of hours required before passing a certain number of cars. For example, if the first five hours contain the following figures of 30, 40, 10, 50 and 100, it takes 3 hours to reach 75 or more cars and 5 hours to 226 or more cars. Note that the user expects the day indices to be 1 7 (this requires adjusting access to the zero base string), and hourly indices 0 23. A:Exercise 1: using the system; From the user's point of view, the first day has the number 1 because //its index is adjusted for the zero value index and //first hour is number 0. Thus, the days have indices of 1-7, //hours has indices of 0-23 classes CarCounter { private int [,] hourCarCounts = new int [7.24]; public CarCounter() { //Initialize all hourCarCounts elements to zero for(int and = 0; i &lt; 7; i++) { for(int j = 0; j &lt; 24; j++) { hourCarCounts[i,j] = 0; } //Display a menu, //have the user enter a command and respond accordingly to the public void run() { string answer; Console.WriteLine(I enter a different value); Console.WriteLine(G Get count for specific hour); Console.WriteLine (S Calculate the total number of cars); Console.WriteLine(H Calculate the number of hours to reach a certain number of cars); Console.WriteLine(T Break); until { Console.Write(Select option:); answer = Console.ReadLine(). ToUpper(); switch(answer) { case I: InputCarCount(); pause, case G: GetHourCount(); break; case S: Console.WriteLine(Total number of cars counts {0}, TotalCarsCounted()); break; case H: HoursToReachCount(); break; case T: Console.WriteLine(Bye Bye!); break; default: Console.WriteLine(Invalid reply. Try again); pause; } until (answer != T); } //Have the user enter the car in a specific daily and hourly public inputcarcount() { int day, int hour; int carCount; Console.Write(Entry Day:); day = Convert.Toint32(Console.ReadLine()); Console.Write(Input Clock:); sat = Console.Write(Car entry number:); carCount = Convert.Toint32 (Console.ReadLine()); 1 is rejected from the day in the next row to adjust // for zero hour of the CarCounts index [day - 1.hour] = carCount; } //Finds the number of cars specified daily and hourly public gaps GetHourCount() { int day; int hour; Console.Write(Enter day:); day = Convert.Toint32(Console.ReadLine()); Console.Write(Enter clock:); hour = Convert.ToInt32(Console.ReadLine()); 1 bounces off the day in the next row to adjust // for zero index Console.WriteLine (Number of cars: {0}, hourCarCounts [day - 1, hour]); } //Finds the sum of all car numbers per hour in the CarCounts public int TotalCarsCounted() { int sum = 0; foreach (int temp in hourCarCounts) { sum += temp; } return total; } //Calculates the hours required to reach a certain number of cars of public void HoursToReachCount() { int reachCarCounts = 0; int hourCount = 0; int sumCarCounts = 0; Console.Write(Input number of cars to be reached:); reachCarCounts = Convert.ToInt32 (Console.ReadLine()); for(int i = 0; i &lt; 7, i++) { for(int j = 0; j &lt; 24, j++) { hourCount++; sumCarCounts += hourCarCounts[i,j]; if(sumCarCounts &gt; reachCarCounts) { Console.WriteLine(Number of hours: {0}, hourCount); return} Console.WriteLine (Default number: {0} never achieved, reachCarCounts); } Class Tester { Public Static Void Main() { CarCounter myCarCounter = new CarCounter(); myCarCounter.Run(); } } 2:Write a program containing a three-dimensional series of int types called numbers with the following dimensions: 5, 10, and 8. The user must be able to enter and retrieve individual values from this field by entering the selected index set. In addition, the program must be able to calculate the sum of all numbers stored in all elements of the number string. A:Exercise 2: using the system; BoxOfNumbers class { private int [,,] numbers = new int [5,10,8]; public BoxOfNumbers() { //Initialize all numbers for(int and = 0; i &lt; 5; i++) { for(int j = 0; j &lt; 10; j++) { for(int k = 0; k &lt; 8; k++) { numbers[i,j,k] = 0; } //Display menu, //let the user enter the command and respond accordingly to the public void Run() { string response; Console.WriteLine(I enter a different number); Console.WriteLine(G Get a number for specific indices); Console.WriteLine; Console.WriteLine(T Break); until { Console.Write(Select option:); answer = Console.ReadLine(). ToUpper(); switch(answer) { case I: InputNumber(); break; case G: GetNumber(); break; case S: Console.WriteLine(Sum of all numbers {0}, NumbersSum()); break; case T: Console.WriteLine(Bye Bye!); break; default: Console.WriteLine(Invalid reply. Try again); pause; } until (answer != T); } //Allows the user to enter a number for a specific set of InputNumber() { int indexDimension1 indexes; int indexDimension2; int indexDimension3; int indexDimension3; int indexDimension Console.Write(Enter index for 1. indexDimension1 = Convert.ToInt32(Console.ReadLine()); Console.Write(Enter index for 2nd indexDimension2 = Convert.ToInt32(Console.ReadLine()); Console.Write(Enter index for 3rd dimension); indexDimension3 = Convert.ToInt32 (Console.ReadLine()); Console.Write(Enter a number:); number = Convert.toint32(Console.ReadLine()); numbers[indexDimension1, indexDimension2, indexDimension3] = number; } //Finds and displays a number for a specific set of GetNumber() { int indexDimension1; int indexDimension2; int indexDimension3; Console.Write(Enter index dimension 1:); indexDimension1 = Convert.ToInt32(Console.ReadLine()); Console.Write(Enter index dimension 2:); indexDimension2 = Convert.ToInt32(Console.ReadLine()); Console.Write(Enter index dimension 3:); indexDimension3 = Convert.ToInt32 (Console.ReadLine()); Console.WriteLine(Number: {0}, numbers[indexDimension1, indexDimension2, indexDimension3]); } //Finds the sum of all numbers publicly int NumbersSum() { int sum = 0; foreach(int temp in numbers) { sum += temp; } return amount; } Class Tester { Public Static Blank Main() { BoxOfNumbers myNumberBox = new BoxOfNumbers(); myNumberBox.Run(); } Page 13 1: You're writing a planet class for the solar system simulation program. You need to track the number of current planetary objects. If you were to store this data as a variable within the Planet, should it be a case variable or a static variable? Why? A: As a static variable, because the variable describes planet objects as a group. 2:Can you assign a value to a static variable even if no objects have been created? A:Yes. 3:Can a static variable be accessed within the object of the class in which it was declared? A:Yes. 4:The following declaration was found within the class: public static const double MassOfElectron = 9.0e-28; Is that right? Why or why not? A:No. Const is implicitly static, so it is not valid to also explicitly declare it static. 5:Is it against the principles of encapsulation to declare the constitution public? A:No, because the const value cannot be changed. 6:The class called Planet contains the following declarations: Planet class { private static uint numberOfPlanetsCreated; private double mass; ... public static double CalculateGravity(...) { ... } public double density(...) { ... } } Is it possible to access the numberOfPlanetsCreated from the Method CalculationGravity Access Mass within the method of calculatingGravity Access numberOfPlanetsCreated within the density method Access mass within the density method Access numberOfPlanetsCreated outside the planet class by writing Planet.numberOfPlanetsCreated Access mass outside the Planet class by writing Planet.mass A: 7:Among other methods, your Planet class contains two methods approximately the same One static method is associated with the Planet class, the other is an exemplary method associated with a specific planet object. Suppose 100 Planet-type objects exist within your simulation of the solar system. Is the following statement true or false? All Planet class case methods take about 100 times more memory than static methods. Why? A:The statement is false because instance methods share the same code. Static methods and method instances take approximately the same memory. 8:Consider the following definition of the method: public emptiness MyMethod (ref int myValue, int yourValue) { myValue = 100; yourValue = 200; } Which of the following calls to MyMethod are correct and, if correct, what is the value of myArgument and yourArgument after myMethod return (myArgument and yourArgument are both type int and initially contain a value of 0)? MyMethod (myArgument, yourArgument) MyMethod (ref myArgument, ref yourArgument) MyMethod(ref 10, ref yourArgument) A: Invalid. ref must suffexite both arguments. Invalid. 10 is not an allotable variable. 9:Is the following method correct? Why or why not? public int Sum (outt x, int y) { return x + y; } A:Incorrect. Each external parameter must be given a value within its method. 10:Consider the following method header: public double average (params int[] numbers) Which of the following calls to this method are valid? (myArray is a series of basic types of int with 100 elements.) A: Invalid, 30.0 is not an int type. 11:The following two header methods were found within the same class. What can you call methods? public int Sum (int number1, int number2) public double sum (double number1, double number2) A:Overloaded. 12: Are the methods with the following method headers overloaded correctly? Why or why not? public double average(int x, int y) public int Average (int x, int y) Are the methods with the following method headers overloaded correctly? Why or why not? public double average (int myX, int myY) public int Average (int yourX, int yourY) A: No, only the type of return differs (one is double, the other int). However, the return type is not part of the method signature, so the two signatures of the method are identical, which is invalid. No, parameter names are not part of the method signature, so although they differ between the two methods, the two signatures are identical, which is invalid. 13:Class contains the following method header: public uint Sum (uint x, uint y) Is the following call valid by method? Why or why not? (myByte and yourByte are both byte type, which differs from Sum's uint type parameters.) Sum(myByte, yourByte) A:Yes, the method call is valid. myByte and yourByte are implicitly converted to uints during call methods. 14:Consider the following class called Planet: Planet class { private double myMass = 2000000; public double doSomething (double myMass) return (myMass * this.myMass) + myMass; } What value will the following method call have when DoSomething returns (if the myMass variable instance value is still 2000000)? DoSomething(8.0pm) A:In line return (myMass * this.myMass) + myMass; myMass approaches the values of the formal parameter myMass (here equals 20.0), while this.myMass approaches the values of the myMass variable instance (here equals 2000000). So the return value is (myMass * this.myMass) + myMass = (20.0 * 2000000) + 20.0 = 40000020 responses to Chapter 12 program exercises 1:Write a class called SportsCar containing two instance variables maxSpeed and horsepower type int. Both of these instance variables must, in any SportsCar property, hold values that are greater than certain predetermined values; Otherwise, they do not qualify as real SportsCar facilities. In our case, the minimum value for maxSpeed is 200 kilometers/hour, and the minimum value for horsepower is 250 hp. Include suitable member variables in the SportsCar class to hold these values (called maxSpeedRequirement and horsepowerRequirement) and write a method called SportsCheck that returns true if both maxSpeed and horsepower for a particular SportsCar object are above minimum requirements; otherwise, false. Write accessor and mutator methods for maxSpeed, horsepower, maxSpeedRequirement and horsepowerRequirement. Include another class that contains the main method for demonstrating the SportsCar class. Please make this program available for exercise 2. 2:Any facility currently from the SportsCar class you wrote in Exercise 1 can only be driven by certain people. Turn on the method in sportscar class that will read in any number of driver names (names); Call it SetLegalDrivers. So, if you have a SportsCar object called sportsCar1 in your program, you can set up legal drivers by, for example, listing the following call: sportsCar1.SetLegalDrivers(Tom, Julie, Teddy, Mary) or call sportsCar1.SetLegalDrivers(Peter) Write a method called DriverCheck that allows you to name as an argument, and then returns true if the driver is allowed to drive the SportsCar object; otherwise, it returns falsely. Please keep this program handy for exercise 3. 3:Add a static method called GetMinimumRequirements that contains two int-type ref parameters called newMaxSpeed and newHorsepower. This should allow other objects of other classes to call this method and get the values of both the maxSpeed and horsepower held by the SportsCar class with only one method calling. Write a class tester called Calculator with a static method called NumberCruncher that makes a call to GetMinimumRequirements and simply prints two values obtained from this call. Keep source code intact for exercise 4. 4:Equip sportscar class with case method called MostPowerful, which has one formal parameter parameter SportsCar to carCompare. Enable SportsCar's MostPoweful comparison method, referenced by the formal carCompare parameter, with SportsCar referenced by the most capable method. The most powerful SportsCar object contains a larger variable of horsepower instances. Return from the method a reference to the most powerful object. Allow not only one SportsCar to be forwarded as an argument to the most powerful method, but also two SportsCar properties. In the event that two SportsCar-type arguments are transferred to MostPowerful, three properties are compared and the most powerful SportsCar object is returned (hint: use the overload method to implement this functionality). A:Responses to all programming exercises are contained in the following program. Program parts related to specific exercises are code-coded. using the system; class SportsCar { private int maxSpeed = 0; private int horsepower = 0; private static int maxSpeedRequirement = 0; private static int horsepowerRequirement = 0; private string [] legalDrivers = new string [0]; //Exercise 2 public bool SportsCheck() { if((maxSpeed &gt;= maxSpeedRequirement) &amp;&amp; (horsepower &gt;= horsepowerRequirement)) return true; else return false; } public void SetMaxSpeed(int newMaxSpeed) { maxSpeed = newMaxSpeed; } public int GetMaxSpeed() { return maxSpeed; } public void SetHorsepower(int newHorsepower) { horsepower = newHorsepower; } public int GetHorsepower() { return horsepower; } public static void SetMaxSpeedRequirement(int newMaxSpeedRequirement) { maxSpeedRequirement = newMaxSpeedRequirement; } public static int GetMaxSpeedRequirement() { return maxSpeedRequirement; } public static void SetHorsepowerRequirement(int newHorsepowerRequirement) { horsepowerRequirement = newHorsepowerRequirement; } public static int GetHorsepowerRequirement() { return horsepowerRequirement; } //Exercise 2 public void SetLegalDrivers(params string [] newLegalDrivers) { legalDrivers = newLegalDrivers; } //Exercise 2 public bool DriverCheck(string name) { foreach(string tempName in legalDrivers) { if(tempName == name) return true; } return false; } //Exercise 3 public static void GetMinimumRequirements(ref int newMaxSpeed , ref int newHorsepower) { newMaxSpeed = maxSpeedRequirement; newHorsepower = horsepowerRequirement; } //Exercise 4 public SportsCar MostPowerful (SportsCar carCompare) { if(carCompare.GetHorsepower() &gt; this.horsepower) return carCompare; else return this; } //Exercise 4 public SportsCar MostPowerful(SportsCar carCompare1, SportsCar carCompare2) { if(MostPowerful(carCompare1). GetHorsepower() &gt; carCompare2.GetHorsepower()) return MostPowerful (carCompare1); second return carCompare2; } } //Exercise 3 Classes Calculator { Public Static Gap NumberCruncher() { int sportsCarMaxSpeedReq = 0; int sportsCarHorsePowerReq = 0; SportsCar.GetMinimumRequirements (ref sportsCarMaxSpeedReq, ref sportsCarHorsePowerReq); Crisis... number crunch Console.WriteLine(Value sportsCarMaxSpeedReq: {0}, sportsCarMaxSpeedReq); Console.WriteLine (SportsCarHorsePowerReq Value: {0}, sportsCarHorsePowerReq); } Class Tester { Public Static Void Main() { //Exercise 1 SportsCar myCar = new SportsCar(); SportsCar yourCar = new SportsCar(); Calling static methods through the name of the SportsCar class.SetMaxSpeedRequirement (200); SportsCar.SetHorsepowerOkviement (250); Methods of calling by object name myCar.SetMaxSpeed (170); myCar.SetHorsepower(110); yourCar.SetMaxSpeed (270); yourCar.SetHorsepower (300); Console.WriteLine(Important {0} my car is a sports car, myCar.SportsCheck()); Console.WriteLine(Important {0} your car is a sports car, yourCar.SportsCheck()); Exercise 2 myCar.SetLegalDrivers(Peter, Ann, Eric); Console.WriteLine(It's {0} peter can drive myCar, myCar.DriverCheck(Peter)); Console.WriteLine(This is {0} josephine can drive myCar, myCar.DriverCheck(Josephine)); Exercise 3 Calculator.NumberCruncher(); Exercise 4 SportsCar mostPowerfulCar; SportsCar herCar = new SportsCar(); herCar.SetMaxSpeed (290); herCar.SetHorsepower (320); mostPowerfulCar = myCar.MostPowerful(yourCar); Console.WriteLine(The most powerful car of your car and my car has {0} of horsepower, mostPowerfulCar.GetHorsepower()); mostPowerfulCar = myCar.MostPowerful (yourCar, herCar); Console.WriteLine(The most powerful car of your car, my car and its car has {0} of horsepower, mostPowerfulCar.GetHorsepower()); } } Page 14 1:A class called Robot contains the following two instances of variable declaration: private ushort age; private bool isConnected; What is the value of age and is associated immediately after their object is currently? Improve the style of declarations without changing the semantics of the code. A:age has a value of 0 and is associated with false value. The improved code would be as follows: private ushort age = 0; private bool isConnected = fake; 2:Users of your class of robots would like to assign an age value at the same time as an instantiating robot object. How would you adapt to this request? A:Equipping the Robot class with the following constructor: public Robot (ushort initAge) { age = initAge; } 3:What constructor names can you use for a class called Robot? A: All Robot class constructors must be called Robot. 4:Are constructors only used to initialize random variables? A:No, they can be used to perform any actions necessary when creating a new object. 5:What is the return type of constructor? A: There is no return value, so there is no return type. 6: The dog class does not contain definitions of the constructor in the source code. Is the following statement (located within the second class method) valid? Dog myDog = new dog(); Decide to include the constructor with the following header in the dog class: public initialAge) Is the previous statement valid now? Why or why not? A: When no constructor is explicitly defined for a class, the compiler automatically includes the default constructor for this class, making the call valid. The previous statement is now invalid because the compiler does not automatically include the default constructor when explicit constructors are specified for the class. 7: The following two constructor headers are in a class called Cat: Public Cat (short initial, initial stringName) : this (initialAge) public Cat (short initialAge) Are these headers valid? If so, what do they mean? What is : what is it (initialAge) used for? A fellow programmer changes these titles to become a public Cat() : this () Is this header worth it? Why or why not? A: Yes, both headers are valid. : this (initialAge) is the initializer of the constructor, which in this case causes the constructor with the public cat header (short initials) to be called before the statements contained in the constructor to which the constructor initializer is attached. This header does not apply because the constructor initializer calls the constructor to which it is attached. The result is an infinite number of calls to the constructor. 8:Next line creates a new Cat object. What brackets after Cat signify? Cat myCat = new cat(); A: Parentheses indicate that it is called a language element similar to a method called a constructor. 9:How can overloaded constructors make class more flexible to use? A:Overloaded constructors contain different formal parameters, so a class with several constructors accepts several different combinations of arguments when a new object is created. 10:Why would you ever want a constructor who can't be called out of class? How to declare this type of constructor? What's his name? A:A Constructor declared private cannot be called out of class. If the class contains only private constructors, the class cannot be instantized outside the class. This technique is sometimes used for classes that contain only static members. 11:What is the name of the static constructor? Where is the name of the static constructor? When is it called a static constructor? What's wrong with the next static constructor header? Static Cat (int initialNumberOfCats) A: The static constructor is called runtime somewhere between running the program and creating the first instance of the class for which it was designed. A static constructor cannot contain any parameters because the run time never makes any arguments when it calls a static constructor. 12:Can you design a program that allows the end user to assign a value to a member: Constant instance? A likeable member of the instance? Tip: What is the life expectancy of a permanent member? What is the lifespan of an easy member? A: No. The permanent value of the instance member is before composing the program at the time the program is written so that the developer must know its value. A permanent member cannot change the value after the program is assembled and cannot change its value until the next time the program is assembled. I do. The read instance member has its own value specified at the time the object is created, after which it cannot be changed. The entire member has the same service life as the object in which it is located. 13:Briefly explain what it means that the object is out of reach. A:The object is out of range when reference variables are not referenced to the object. 14:What are the two main tasks of garbage collectors? A:Identify objects that are out of reach. Restore memory assigned to objects that are out of range. 15:Why doesn't a garbage collector just collect any object that immediately go out of reach? A: This is an inefficient approach. A better approach is to collect a group of objects that can be returned and then process them all at once. 16:What is a destructor? Why is it not useful for releasing scarce non-memory resources? A:Destructor is explicitly defined for the class, in which case it contains statements written by the developer. Destructor can only be called a garbage collector during the collection of an elusive object. Destructor is not useful for releasing scarce non-memory resources because we do not know when it is called; in fact, it can never be called during the time of operation of the program. 17:The program contains a class called Book that contains 10 instances of variables of all simple types. During the execution of the program, 50 items of the Book are created and most of them are destroyed. Is the garbage collector likely to run during the execution of your code? A:No 18:Why should you be careful when using C# (at this point in writing) for real-time apps? A:When the garbage collector is activated, the rest of the program is frozen, so the real-time application is not in real time during this period. When we combine this fact with the inability to control when a garbage collector is working, it becomes difficult to write an app in real time with GC-based programming language such as C#. 19:Briefly explain why it's a good idea to use dispose design pattern to free up scarce non-memory resources. A: The disposition design form gives complete control to the developer when to dispose of the scarce resources held by the object. Responses to Chapter 13 program exercises 1: Write a class called Robot with the following three instances of variable: name (string type), age (type ushort) and isOn (bool types). Have the program initialize these three instances of the variable to unknown, 0, or false (without using a constructor). Include access and mutato methods for assigning and retrieving the values of these instance variables. Write method that tests the Robot class. A:Exercise 1: Robot class { private string name = unknown; private ushort age = 0; private bool isOn = false; //Accessor and mutator methods } 2:Allow the users of the Robot class to set the initial values of the three instances of variables when a Robot object is created. For this purpose, declare one constructor with three formal parameters and one default constructor. Include a member variable called robotsCreated that tracks the number of robotic objects created. Have the constructors update the robotsCout so that this variable is always an accumulation. Implement the following logic into the default constructor: If the RobotsCreated are less than five when this constructor is called, set isOn to true; otherwise, set it to false. A:Exercises 2 and 3: Robot class { private string name; private age use; private bool isOn; private static int robotsCreated; static Robot() { robotsCreated = 0; } public Robot (string initName, ushort initAge, bool initIsOn) { name = initName; age = initAge; isOn = initIsOn; robotsCreated++; } public Robot() { name = unknown; age = 0; robotsCreated++; if(robotsCreated &lt; 5) isOn = true else isOn = false; } //accessors and mutator methods } 3:The Robot is able to perform a few simple calculations. For example, it can find an average of three int numbers. This capability is provided by a method with the following header: public int Average (int x, int y, int z) This method does not in itself perform an average calculation, but uses a static method called Average contained within a class called RobotMath. The RobotMath class contains only static methods and static variables and should never be instantaneous. Make sure RobotMath is never momentous. A:Exercise 4: Robot class { ... public int Average (int x, int y, int z) { return RobotMath.Average(x, y, z); } ... RobotMath class { //private constructor prevents the creation of any cases from //outside the RobotMath class. private RobotMath() { } public int Average(int x, int y, int z) { return (x + y +z) / 3; } } Page 15 1: If Speed property is defined within an object called myRocket, what is the general name for the statement block defined for the Speed property that are executed when performing the following line? myRocket.Speed = 40; When is the next line executed? tripTime = distance / myRocket.Speed; A: Set up block statements or set the gateway to get a statement block or get accessor 2:What style of capitalization is recommended for A: 3:Following lines list case variable statements and what it meant to be a property with the task of allowing access to a variable speed case off-site. However, the definition of a property has four problems that need to be fixed before it will work properly. Find and fix problems. private double speed; public double speed { get { return speed; } } A:Change privately to change the int to double, remove parentheses after speed, and the speed of return does not speed. private double speed; public double speed { get { return speed; } } 4:What is delayed initialization? When would you use it? Can delayed initialization also be performed by accessor and mutator methods? A:Delayed initialization When the program does not initialize instance variable until its associated profit statement block is invoked for the first time. Delayed initialization is used with instance variables that are resource-required to update and are rarely used. Delayed initialization can also be performed by accessor and mutator methods. 5:The next indexer statement header has three problems. What are they? static int myIndexer this [ ] { ... } A:Indexer cannot be static and has no name (like myIndexer); only the keyword is used. At least one parameter must be specified within square brackets. The following is a valid indexer header: public int this [int index] { ... } 6:Rainfall class contains a string with twelve elements. It also contains an indexer to access field elements, which, like its string, is zero-based (the first element is zero). RainfallParis is currently in rainfall class. Write a statement that assigns the value of the third precipitation elementParis to the variable rainMarch. Write a statement that assigns rainfall valueJuly to the seventh element of precipitationParis. You are expanding the rainfall class by another method. Among other things in this method, you must assign the third indexer element to the myMarchRainfall variable. Write a statement. A: rainfallMarch = rainfallParis[2]; rainfallParis[6] = rainfallParis; myMarchRainfall = this[2]; 7:You have just finished implementing your car class and are satisfied with the end result. During a quiet moment of satisfaction, your colleague suddenly suggests that in order for the car class to be perfect, you need to turn on operator overload for class so that it can be used with + and - operators and support code such as car3 = car1 + car2. Is that a good idea? Why or why not? A: This is not a good idea. We don't know intuitively how car objects connect, so turning on operator overload for the car class is likely to make the code more encrypted. 8:Why is operator overload sometimes referred to as syntactic sugar? A:Operator overload changes the appearance of the code syntacticly (sometimes it looks sweeter), but below the syntax surface, overloading the operator is just another way to call the method. 9:Recall the TimeSpan watch from list 14.7. It already includes an operator + method for adding two TimeSpan objects together and assigning results to a time reference variable. Your colleague decides to turn on the operator - a method so that TimeSpan objects can be subtraced. Follow the code he inserted into the timespan class. It contains several drawbacks. Find them. private TimeSpan (TimeSpan timeSpan1, TimeSpan timeSpan2) { return (timeSpan1.Seconds timeSpan2.Seconds) } A:The operator - the method must be public and static and include the operator's keyword in front of the minus symbol. Furthermore, it must return the TimeSpan object, not the uint type value. Correct operator - method looks as follows: public static timeSpan operator- (TimeSpan timeSpan1, TimeSpan timeSpan2) { TimeSpan differenceTimeSpan = new TimeSpan(); differenceTimeSpan.Seconds = timeSpan1.Seconds - timeSpan2.Seconds; return differenceTimeSpan; } 10:Of the other programming elements, your program contains two classes EarthTimeSpan and BliposTimeSpan. Each class still contains an instance variable called totalSeconds of type uint. In this case, EarthTimeSpan contains a user-defined explicit conversion to a short type, and BliposTimeSpan contains an implicit conversion defined by the user from the decimal type to BliposTimeSpan. myShort is a short-type variable and myDecimal is a decimal type variable, earthTimeSpan1 is of the EarthTimeSpan type, and bliposTimeSpan1 is of type BliposTimeSpan. Specify whether each of the following statements is valid: myDecimal = bliposTimeSpan1; myDecimal = (decimal) bliposTimeSpan1; myShort = (short) bliposTimeSpan1; myShort = countryTimeSpan1; myShort = (short)countryTimeSpan; myDecimal = earthTimeSpan; myDecimal = (decimal)countryTimeSpan; bliposTimeSpan1 = earthTimeSpan1; bliposTimeSpan1 = (short)countryTimeSpan1; bliposTimeSpan1 = (decimal)countryTimeSpan1; earthTimeSpan1 = bliposTimeSpan1; earthTimeSpan1 = (short)bliposTimeSpan1; earthTimeSpan1 = (decimal)bliposTimeSpan1; A: 11:You are writing a program that will help architects draw architectural plans. The program contains many classes that rely on manipulating two-dimensional drawings. One of these classes is called the Bathroom, and it represents and manipulates the drawings of bathrooms. It needs a class called Point to represent a point consisting of two coordinates. Would it be a good idea to nestle this class in a bathroom class? Why or why not? A: If there is a class called Bathroom, it is also likely that there will be classes with names like Kitchen, Bedroom and so on. All these classes have a fundamental need to manipulate two-dimensional drawings, and therefore the need to use the point class. Therefore, it is not a good idea to isolate the point class inside the bathroom. Instead, it needs to be positioned where all these classes can access it and share the functionality provided by the Point class. Answers to Chapter 14 of Program Exercise 1: Expand the class of bikes with a list of 14.2 to contain an instance variable called age representing the age of the bike. Write down a property that allows users of the Bike class to set up and change the age instance. Turn on code that prevents variables than the assigned negative value or value greater than 200. Equip the bike class with an instance variable called the numberOfAgeAccesses, which represents the number of times a variable's age instance is accessed. Turn on the object to access the NumberOfAgeAccesses. Should this latter property contain both a set and a block statement? A:The following code can be inserted into the bike class to respond to exercise 1. Keep on the upsue: This code is not compiled separately. const byte MaxAge = 200; private byte age; private uint numberOfAgeAccesses = 0; public byte Age { get { numberOfAgeAccesses++; return age; } set { if (value &gt; MaxAge) Console.WriteLine(Error. {0} exceeds the age {1}, value, MaxAge); other if (value &lt; 0) Console.WriteLine(Error. {0} can not be less than 0, the value); other ages = value; } } public uint NumberOfAgeAccesses { get { return numberOfAgeAccesses; } } 2:Write a class called Rain containing a one-dimensional string with 12 elements representing monthly rainfall measurements. Turn on the following features: Allow rainfall object users to access elements of 12 int index fields as if these objects were fields themselves. Have the first month identify with index 1 (not zero). Make sure all indexes are within the correct limits. That the program counts the number of times rainfall readings have been accessed. Store this number in an instance of a variable named numberOfRainAccesses. Include a property called Average that calculates average monthly rainfall. Please keep this class for use in exercise three. 3:Allow two precipitation objects to be added along with the + symbol This should allow you to write the following: totalRainfall = rainfallA + rainfallB; where all three objects are of the type Rain. totalRainfall should now hold a string in which the first element contains the result of adding the first element of the field to rainfallA along with the first element of the field in rainfallB; the second element holds the result of adding another field element to rainfallA along with another field element to rainfallB, and so on. Allow the two precipitation objects to compare with &lt; and &gt; for comparison. Let the average of 12 elements in each of the two different objects be the determining factor. So if the average rainfall is greater than rainfallB, the following phrase will be true: rainfallA &gt; rainfallB and the following expression will be fake: rainfallA &lt; rainfallB Use the average property you have already written in Exercise 2 to provide the necessary information for comparison. Save the code for use in exercise 4. 4:Write a class called RainfallQuarterly that contains a series with four elements for storing rainfall information on a quarterly basis. Write conversions defined for users between and Precipitation goes both ways. The Rainfall object is converted to RainfallQuartely by adding the first four elements of the field and assigning that value to the first string element of the RainfallQuarterly object, and so on. RainfallQuartely converts to Rainfall by dividing the first element of the field by four and assigning that value to each of the first four elements in the rainfall string, and so on. Should both conversions be explicit or just one of them, or perhaps both should be implicit? Why? Tip: What conversions do data lose? In what conversion is not lost? A:The following code may be the answer to exercises 2 4. Keep safe: This code is not compiled independently. This class is the answer to questions 2 and 3.class Rainfall { private uint[] rainfall = new uint[12]; private uint numberOfRainAccesses = 0; //Initialize the rainfall array public Rainfall() { for(int i = 0; i &lt; 12; i++) { rainfall[i] = 0; } } publicint this [int index] { get { if (index &lt; 1 || index &gt; 12) { Console.WriteLine(Index is out of bounds); return 0; } else { numberOfRainAccesses++; return rainfall[index - 1]; } set { if (index &lt; 1 || index &gt; 12) Console.WriteLine(Index is out of bounds); else rainfall[index - 1] = value; } } public uint NumberOfRainAccesses { get { return numberOfRainAccesses; } public uint Average { get { uint sum = 0; for (int i = 0; i &lt; 12; i++) { sum += rainfall[i]; } return (sum/12); } } public static Rainfall operator+ (Rainfall rainfallfall2) { Rainfall sumRainfall = new Rainfall(); //Add each corresponding month pair of the two //Rainfall objects together and assign the result //to the corresponding month of sumRainfall. for (int i = 1; i &lt; 13; i++) { sumRainfall[i] = rainfall1[i] + rainfall2[i]; } sumRainfall } public static bool operator&gt; (Rainfall rainfall1 , Precipitation precipitation2) { if (rain1. Average &gt; rainfall2. On average) return true; others return falsely; } Public static bool operator&lt; (Rainfall rainfall1, Rainfall rainfall2) { if (rainfall1. Average &lt; precipitation2. On average) return true; others return falsely; } //This class was written in response to program exercise 4. RainfallQuarterly class { private uint[] rainQuarterly = new uint[4]; public uint this [int index] { get { if (index &lt; 1 || index &gt; 4) { Console.WriteLine(Index is out of bounds); return 0; } other { return rainQuarterly[index - 1]; } set { if (index &lt; 1 || index &gt; 4) Console.WriteLine(Index is out of bounds); other rainQuarterly[index - 1] = value; } //Implicit as no data is lost: For each quarter, three corresponding //months are added together and the amount allocated to this Quarter. Public Static Implicit Operator RainfallQuarterly (Rainfall convertFrom) { RainfallQuarterly newRainQuarterly = new RainfallQuarterly(); uint tempSum; int tempIndex = 1; //For each add add months together and //add this value in the quarter for (int and = 1; and &lt; 5; i++) { tempSum = 0; for (int j = 1; j &lt; 4; j++) { tempSum += convertFrom[tempIndex]; tempIndex++; } newRainterly[i] = tempSum; } Explicit because data is lost when each quarterly amount is converted to //three monthly values of all the same size. public static explicit operator Rainfall (RainfallQuarterly convertFrom) { Rainfall newRainMonthly = new Rainfall(); int tempIndex = 1; uint tempAverage; //For each quarter, calculate the monthly average rainfall //Assign this value to months this quarter for (int and = 1; i &lt; 5; i++) { tempAverage = convertFrom[i] / 3; for (int j = 1; j &lt; 4; j++) { newRainMonthly[tempIndex] = tempAverage; tempIndex++; } } return newRainMonthly; } Page 16 1:What is the full class name defined within these two namespaces? namespace BliposSoft { namespace RocketSimulation { Rocket class { ... } } A:BliposSoft.RocketSimulation.Rocket 2:A developer in an uncontrolled vanity moment gave the following name to the class: ThisIsTheBestRocketClassInTheWholeWideWorldAndTheWholeWideBliposAsWell The claim in the name is true, so you must use this class extensively in your program, but you do not want to write a long name. How can this be avoided? A:You can create a shorter alias, such as Rocket for ThisIsTheBestRocketClassInTheWholeWideWorldAndTheWholeWideBliposAsWell with the following line: using Rocket = ThisIsTheBestRocketClassInTheWholeWideWorldAndTheWhole WideBliposAsWell; Now you can use the name Rocket in code instead of a long bulky name. 3:The next class is in the compilation unit that will be part of the DLL assembly. Bike class { dual CalculateAirResistance() { ... } double WheelRotationsPerMinute() { } } Access modifiers are missing in front of the class and two methods. What approach modifiers should we put ahead of class and method for: Allow the whole class and both methods to be used in the second set. Prohibit any part of the class from being used in another assembly, but allow the methods to be used within your own assembly. Allow access to the CalculateAirResistance method from another assembly, but not the WheelRotationsPerMinute method. Allow all classes to be used in the second part along with CalculateAirResistance, while WheelRotationsPerMinute must only be accessed within the class. A: Declaring both the class and two methods for the public. Report both class and two methods for internal. Declare class and CalculateAirResistance public, but WheelRotationsPerMinute to be internal. Declare class and CalculateAirResistance public and WheelRotationsPerMinute private. 4: Work for a software company called BikeTech that plans to write a class library containing classes from major sections in the company: bike design, health and fitness, and computer mapping. Use C# code to illustrate the overall appearance of the namespace you would suggest for this class library. A: namespace BikeTech { namespace BicycleDesign { ... } namespace HealthAndFitness { ... } namespace ComputerMapping { ... } 5: Three different compilation units contain each of the following definitions of namespace: Compilation Unit 1: namespace MyCompany { Public Bike Class { ... } } Compilation unit 2: namespace MyCompany.Design { public class Tray { ... } } Compilation unit 3: namespace MyCompany.Design.Tools { Public Class Slicer { ... } } Compilation units must be assembled into the same assembly. Instead of having three compilation units, write one compilation unit with namespace definitions that match those contained in the three compilation units displayed. Use a nested C# name definition format that looks like the following: namespace &lt;outer_namespace&gt;{ namespace &lt;inner_namespace&gt;{ etc. A: namespace MyCompany { public class Bicycle { ... ... } namespace Design { public class Drawer { ... } namespace Tools { public class Cutter { } 6: You have written two original files called Bicycle.cs and Person.cs. You want to assemble them and create a DLL assembly called healthlib.dll. For this compilation, you must also find the following two sets: mathlib.dll and anatomylib.dll. Write the compiler command you would give to the compiler. A: csc /out:healthlib.dll /t: library /r:mathlib.dll;anatomylib.dll Bicycle.cs Person.cs 7:What can you use Ildasm for? A:Ildasm will review the contents of any assembly by displaying its ingredients in a user-friendly GUI. Answers to Chapter 15 1 program exercises: Recall your first object-oriented source code SimpleElevatorSimulation.cs in chapter 5.1 list, your first object-oriented C# program. It contained three classes in the same compilation unit. Looking back, he didn't follow the standard one-class convention per compilation unit. Correct this problem and rewrite the program so that it more correctly contains three compilation units, each of which has its own class (Elevator, Person, and Building) and all classes properly structured within the neat hierarchy of namespace. Convert these three compilation units to a DLL assembly. (Tip: You should rename the main method from the building class, as this new DLL is only for re-production). Write another source file that runs an Elevator simulation similar to the one performed by our original program, but this time it uses name spaces and their classes located in a DLL that you previously created from three classes. A:Exercise 1: Create the following three compilation units. (The content of individual classes is omitted for space reasons. You can easily insert code from list 5.1 here.) &lt;/outer_namespace&gt; &lt;/outer_namespace&gt; one with the following total content is called Elevator.cs: using the system; namespace ElevatorSimulation { public class elevator { ... } } Compilation department two with the following content is called Person.cs: using the system; namespace ElevatorSimulation { Public Class Person { ... } } The compilation of units three with the following content is called Building.cs. (The building class contains the RunSimulation method instead of the main method.) using the system; namespace ElevatorSimulation { public class building { ... runsimulation() public void { ... } } } Assemble three compilation units into a DLL called ElevatorSimulation.dll with the following compiler command: csc/out: ElevatorSimulation.dll /t:library Elevator.cs Person.cs Building.cs Now you can write a separate small program (called, for example, Simulator.cs) that calls the RunSimulation building class method. Its main ingredients are as follows: the use of the system; using elevatorsimulation; Class Simulator { Public Static Void Main() { Building aBuilding = New Building(); aBuilding.RunSimulation(); } When you Simulator.cs, you must refer to ElevatorSimulation.dll as in the following command that generates a .exe file called Simulator.exe. csc / r:ElevatorSimulation.dll Simulator.cs To run the program, you can now simply type a Simulator that should result in a known exit from List 5.1. Observe that you can alternatively assemble only Elevator.cs and Person.cs compilation units in .dll, and then create your own simulation similar to that contained in the building class. This latter class should be assembled with a .dll containing code Elevator.cs and Person.cs. This approach is used in the following response. 2: Run the same type of workout as Exercise One, but this time use the BankSimulation.cs source file in Chapter 10.14 list, Arrays Part I: Array Essentials, as the basis. Write one compilation unit for each of the two Account and Bank classes and place them in the appropriate namespace hierarchy. Assemble this part into a DLL assembly. Write the source code for a bank simulation program that uses spaces with the names and classes of the DLL assembly created. The functionality of this program should be identical to that of the program we originally created from Listing 10.14. A:Exercise 2: First create the following two compilation units. (The content of individual classes is omitted for space reasons. You can easily insert code from BankSimulation.cs.) Compilation of units one with the following general content is called Account.cs: using the system; namespace BankSimulation { public class account { ... } } Compilation of units two with the following content is called Bank.cs: using the system; namespace BankSimulation { Public Class Bank { ... } } Have the compiler generate .dll called Banklib.dll with the following command: csc /out: Banklib.dll Bank.cs Account.cs Write a compilation department with BankSimulation class content from List 10.14 and call it BankSimulation.cs: use of the system; using BankSimulation; { Public Class BankSimulator { Public Static Void Main { ... } Assemble this compilation unit into a .exe assembly called BankSimulation.exe with the following compiler command: csc/r:Banklib.dll BankSimulation.cs Now you can run a bank simulation with the following command: BankSimulation It should generate the same output as the 10.14 list. Page 17 1: You have just started working on two software projects and identified the following classes in two programs: Program 1: Plane, Jet Engines, Wings, Aircraft Body, Passenger Seats and Cockpit Program 2: Person, Student, Employee, Undergraduate, Graduate, Secretary, Cleaner and Director Which program seems to benefit the most from the inheritance? Why? What concept is another program likely to benefit from? A:Program 1 contains the plane and parts of the plane. We can say that the plane has a jet engine, and the plane has a wing and so on. Consequently, the concept of aggregation appears to play an important role in the implementation of Programme 1. In Program 2, we find many connections. For example, a student is a person, an employee is a person, and a secretary is an employee. Therefore, it seems that the program can benefit from the concept of succession. 2:Consider the following class hierarchy. In which class would you include each of the following class members: BrandName instance variable PurchasePrice instance variable A: 3:Let's assume that a class called Animal has a public method called Move. If the class of the dog is derived from the class of animals, can you call the Move method for the example of the class of the dog as in myDog.Move(),, even if you do not specify such a method in the class of the dog? A:Yes, you can make a call myDog.Move(). The method of displacement is inherited by a class of dogs from the animal class. 4:This is what the Move method header looks like in the animal class: public emptiness Move() Can you override this method in the dog class? Why or why not? A:No, you cannot override this method. To enable this, Move must be declared virtual. 5:The animal also contains a private instance variable called name. Can you access the name from the definition of dog class? Is that an advantage or a disadvantage? A:No, you cannot access the name from the definition of the dog class. This is an advantage because it supports the principle of encapsulation. 6:The animal further contains a member of the private function. Can you call this member of the function from the definition of the dog class? A:No: 7:The animal contains another method with the following header: Protected Virtual Void MoveADistance (int distance) A fellow programmer wrote the following method header in the dog's derivative class to override Animal Public Bridging int MoveADistance (double distance) This method header contains several errors. Find and correct errors. A:The following is the correct header: Protected Gap Override MoveADistance (Distance from Int) 8:Can you prevent the class from acting as a base class? If so, how? A:Yes, sealing it with a sealed keyword. 9:Why does it usually make sense to call a base class constructor from a constructor of a derived class? A:Part of the derived class is composed of a basic class. Basic class constructors know how to initialize this part. 10:The class of our animals from the previous questions has exactly one constructor that takes one parameter of the type int. A fellow programmer has implemented a Dog class with a constructor that does not contain an explicitly defined constructor initialiser. Why did this setting lead to a compiler error? A: Any constructor of a derived class that does not contain an explicit constructor initializer automatically receives an implicit constructor initializer attached, which calls the default base class constructor. This happens, even if the default base class constructor is not defined. As soon as one constructor is defined for a class, the default constructor is no longer automatically supplied. The base class in this case does not have a default constructor because it contains one explicitly defined constructor that takes a single argument. The combination of the two scenarios described causes the compiler to error. 11: The class of animals is also equipped with a complex method that restores the metabolic rate of the basic animal cell. Its method header in the animal class looks like the following: public virtual double MetabolicRateCell() After calculating metabolicRateCell for the animal, it is easy to calculate MetabolicCellRate for the dog; simply add 100 to this number. You need to override MetabolicRateCell () in the dog class and write its implementation. Write code for the dog class bridging method by using a character returned from the MetabolicRateCell method in the elementary class. A: Public override of the double metabolicratecell() { return base. MetabolicRateCell() + 100; } 12:Now from the dog class you come from a dog class called Poodle. Does the poodle class contain a method originally defined in the animal name Move? A:Yes, class members are inherited at several levels of succession. Answers to Chapter 16 program exercises 1: Write four classes called ElectronicDevice, Radio, Computer, and MobilePhone. Let ElectronicDevice be the base class for the other three classes and include the following three class members in its definition: Private instances of variables called brandName (type string) Public ownership called BrandName to access the brandName Private instances variable called isOn (types of bool) Also include two methods called SwitchOn (which must be written and set isOn to true) and SwitchOff (which must write Off and set isOn to fake). The three subclasses remain empty for now. Write the appropriate code to test that while the three subclasses have empty definitions, you can use their legacy class members. 2:Override switchon and SwitchOff methods in each of the subclasses so that in addition to the actions they perform in ElectronicDevice, they also spell the name of the device that turns on or on. For example, when you call the SwitchOn method for Radio, you should see the next output on the screen on the radio and isOn should still be given a true value. (Tip: Use a basic approach to achieve this functionality.) 3:Equip the ElectronicDevice class with two constructors: The default constructor that places the brandName on the unknown and isOn on false. A constructor that takes one string type argument used to initialize the brandName while isOn is also set to false. Add the following instances of a variable to the program: Add the current frequency of type double to radio. Add int internal memory to your computer. Add lastNumberDialed type uint to MobilePhone. Add the following two constructors to each of the classes of radio, computer, and mobile phones: The default constructor that not only initializes instance values defined in a particular class, but also inherited instances (brandName and isOn) from ElectronicDevice. A constructor that takes two arguments one to set values defined for a class (either current frequency, internal memory, or lastNumberDialed) and one to set a brandName value. (Tip: Call the appropriate constructor initials to do this.) 4:Write a class called LaptopComputer that is derived from the Computer class. Define the variable instance of the maxBatteriLife type uint for this class and provide two constructors: One default constructor that assigns corresponding values to maxBatteriLife, as well as other members of the class that LaptopComputer inherits. One constructor with which you can transfer home values to maxBatteriLife, as well as members of the LaptopComputer class inherits from their offspring. A: When you've written a code for all programming exercises, your code should look somewhat similar to what's shown next. Perhaps you should add additional public properties to access the private instance of variables and extend the test code beyond what is shown here. using the system; ElectronicDevice class { private string brandName; private bool isOn; public ElectronicDevice() { brandName = unknown; isOn = false; } public ElectronicDevice(string initBrandName) { brandName = initBrandName; isOn = false; } public brandName get { return brandName; } set { brandName = value; } } } public virtual virtual Switcha blankOn() { isOn = truth; Console.WriteLine(On); } Public Virtual Void SwitchOff() { isOn = Console.WriteLine(Off); Console.WriteLine(Off); } Class Radio : ElectronicDevice { private double currentFrequency; public Radio() : base() { currentFrequency = 0; } public Radio(double initCurrentFrequency, string initBrandName) : baza(initBrandName) { currentFrequency = initCurrentFrequency; } public override void SwitchOn() { base. SwitchOn(); Console.WriteLine(Radio); } Public undo switchoff() { database. Switching(); Console.WriteLine(Radio); } Computer class : ElectronicDevice { private int internalMemory; public Computer() : base() { internalMemory = 0; } public Computer(int initInternalMemory, string initBrandName) : base(initBrandName) { internalMemory = initInternalMemory; } publicride void SwitchOn() { base. SwitchOn(); Console.WriteLine(Computer); } Public undo switchoff() { database. Switching(); Console.WriteLine(Computer); } Class MobilePhone : ElectronicDevice { private uint lastNumberDialled; public Mobile phone() : base() { lastNumberDialled = 0; } public Mobile phone (uint initLastNumberDialled, string initBrandName) : base (initBrandName) { lastNumberDialled = initLastNumberDialled; } publicride void SwitchOn() { database. SwitchOn(); Console.WriteLine(Mobile Phone); } Public undo switchoff() { database. Switching(); Console.WriteLine(Mobile Phone); } LaptopComputer class : Computer { private uint maxBatteriLife; public laptopComputer() : base() { maxBatteriLife = 0; } public laptopComputer(uint initMaxBatLife, int initInternalMemory, string initBrandName) : base(initInternalMemory, initBrandName) { maxBatteriLife = initMaxBatLife; } } class Tester { public static gap Main() { Radio myRadio = new Radio(100, Bang &amp; Olufsen); Console.WriteLine(BrandName: + myRadio.BrandName); LaptopComputer myLaptop = new laptop (12, 256, IBM); Console.WriteLine(BrandName: + myLaptop.BrandName); } Page 18 1: Consider the animal class from which the dog, cat and duck classes originate. Suppose any animal can make a sound. Where would you locate the sound method? Would you provide an implementation for this method or declare it abstract? Why? A:The method of sound should be positioned in the class of animals because each animal can make a sound. We don't know the sound the animal makes (it could be meowing, vrooff or something else), so the sound should be declared abstract in Animal. 2:If the sound method of the animal class was declared abstract, could you instantiate an object from this class? Why or why not? A:No. The class of animals must be declared abstract because it contains an abstract method of sound. Abstract class can not be instantized, otherwise it would be possible to call methods without any implementation. 3:You want to implement the sound method in each of the three subclasses of the animal and call these three methods polymorphic. What keywords would you use to declaring the animal sound method and in three this scenario that will take place? Write method headers in animal class and three subclasses. A:Methodo sound header in animal class is as follows: public abstract emptiness Sound(); The sound header is the same in all three derived classes: public bridging of the Sound() 4:What is wrong with the following piece of code? public abstract void Sound() { Console.WriteLine(Quaaakkk quaaakkk); } A:The abstract method cannot have an implementation. 5:If you want to call the sound method of three different subclasses polymorphic, should you do this for the Animal type variable or for three variables of type Cat, Dog and Duck? A:For the animal type variable. 6:Suppose the sound method is declared in the animal class and implemented in three subclasses so that each of these three subclasses can be called dynamic binding. Which of the three implementations was called in the second line of the following code: Animal myAnimal = new Dog(); myAnimal.Sound; A:The implementation of the sound method written in the Dog class is called and executed. 7:You have another animal class called Lion that also contains a sound method that you want to call polymorphic through an Animal type variable. What do you need to do to make that possible? A: Let the Lion class be derived from the class of animals. Make sure the sound method is defined by the overriding keyword. 8:You need to find out if it is variable myAnimal (declared animal myAnimal;) refers to a Dog-type object. How can you find out? A:Using the operator as follows: (myAnimal is Dog) that returns true if myAnimal calls for a Dog-type object. 9:You must throw myAnimal into a Cat type object, but only if myAnimal contains a Cat. Show two different ways to do this. A:You can use the IS operator as shown in the following lines: if (myAnimal is Cat) myCat = (Cat) myAnimal or as the operator as in the following row: myCat = myAnimal as Cat; 10:Consider the class of cats from previous questions. For this, you have defined only the sound method. Another developer uses your Cat class and writes the following calls in their code (lines 2 and 3): Cat myCat = new Cat(); myCat.Jump(); Console.WriteLine(myCat.ToString()); Are both of these calls valid? If any of them are valid, what is the outcome of the call? Explain what's going on here. A:myCat.Jump() is invalid because Cat does not contain any jump method. However, myCat.ToString() is valid because Cat implicitly inherits the ToString method from the System.Object class. The call returns the class name in a string (Cat). 11:If most methods are likely to be declared virtual, why isn't the method virtual by default? A: Basic class virtual methods can create more havoc in derived classes than non-virtual methods. virtual methods are executed slightly slower than non-virtual 12:Why doesn't C# support multiple inheritances? A:A few subtle problems are associated with multiple inheritances. 13:What is the problem with the following interface definition: IRecoverable interface { public blank Recover() { Console.WriteLine(I'm recovering); } A:First, the public access modifier cannot be applied to the Recovery method. Secondly, Recovery is implicitly abstract, so it cannot have an implementation. A half-gallon must be inserted instead of implementation. 14:The developer suggests that you can improve your code by exchanging the Animal class introduced in Questions 1 3 with an interface called IArticulateable. Is he or she right? Why or why not? A: You can create the same functionality with IArticulateable, but you can't improve the code. The power of the interface is realized when you want to implement polymorphism on a group of classes that do not have a common ancestor (Animal is a common ancestor). Responses to Chapter 17 1 program exercises: Write a simple account class that contains an instance variable called a balance and a property called Balance to provide access to balance. Allow the account class to be sorted by the generic BubbleSortAscending contained in the Listing 17.13 sorter class by allowing it to implement the IComparable interface. Test your account class by creating a series of accounts with different balances and pass this on as a bubbleSortAscending method argument. A:Exercise 1: using the system; public interface IComparable { int CompareTo(IComparable comp); } public class Account : IComparable { private decimal balance; public Account(decimal initBalance) { balance = initBalance; } public decimal Balance { get { return balance; } set { balance = value; } public virtual int CompareTo(IComparable comp) { Account compareAccount = (Account) comp; if(balance &gt; compareAccount.Balance) return 1; if(elseAccount.Balance == balance) return 0; return else -1; } } class Sorter { // Sort the comparable elements of an array in ascending order public static void BubbleSortAscending(IComparable [] bubbles) { bool swapped = true; for (int and = 0; swapped; i++) { swapped = false; for (int j = 0; j &lt; (bubbles. Length - (and + 1)); j++) { if (bubbles[j]. CompareTo(bubbles[j + 1]) &gt; 0) { Swap(j, j +1, bubbles); swapped = true; } } } //Swap two elements of an array public static void Swap(int first, int second, IComparable [] arr) { IComparable temp; temp = arr[first]; arr[first] = arr[second]; arr[second] = temp; } class Tester { public static void Main() { Account [] accounts = new Account[4]; accounts[0] = new Account(100); accounts[1] = new Account(200); accounts[2] = new Account(50); accounts[3] = new Account(75); Sorter.BubbleSortAscending(accounts); foreach (tempAccount account in accounts) { Console.WriteLine (tempAccount.Balance); } 2:Write three classes called Secretary, Director and Programmer. Each of the classes contain CalculatingSal Methods. For simplicity, let each of these methods write Now calculate the salary for... follows the name of the class in which the method is located. Suppose there are many items of these classes in your program and need to be stored in a collection, such as a string. How can you store them all in one string and call each object's CalculateSalary simple iteration through a string and creating the same call method for all objects? Write the code that implements this scenario. A:Exercise 2: Create a core class called Employee. Extract each of the three classes from this elementary class as shown in the following code: system usage; abstract class Employee { Public Abstract Void CalculateSalary(); } Class Secretary : Employee { Public Override CalculateSalary() { Console.WriteLine(Calculate Secretary Salary); } } Class Director : Employee { Public Override CalculateSalary() { Console.WriteLine(Calculation pay for the director); } } class programmer : Employee { Public Bridging Gap Calculations() { Console.WriteLine(Calculate Developer Salary); } } Class Tester { Public Static Gap Main() { Employees [] Employees = New Employee[3]; employees[0] = new secretary(); employees[1] = new director(); employees[2] = new developer(); foreach (Employee TempEmployee) { tempEmployee.CalculateSalary(); } 3 } :P didt question code 2 by implementing the following separate hierarchy: Base class called Building containing two instances of variable called age type int and currentValue type decimal. Make two classes out of a building called House and Officebuilding. The house contains an instance variable called numberOfBedrooms of type ushort and an OfficeBuilding instance variable called floorSpace type uint. You want to write and read Secretary and House subjects in a file. So both of these classes must contain methods called Read and Write that (because of the simplicity of the reading method write Now Read House and Now Read Secretary on console and similar for writing methods) save these items in the file and from the e-file. You want to construct only one method that accepts any object that contains reading and writing methods and will call these methods polymorphic, regardless of what type of object it is. Write code that implements this functionality. A:Exercise 3: The following code contains an interface called IStorable that has two abstract reading and writing methods. Secretary and House of Class and implement this interface. The transfer method requires the IStorable argument. So we can also send an example of the type of secretary and the House type to this method. This is done in the tester class. Note that this code results in several alerts when composing because none of the random class variables are used (to make the code short). Using abstract class Employee { public abstract gap CalculateSalary(); } class secretary : Employee, IStorable { public override void CalculateSalary() { Console.WriteLine(Calculate salary for secretary); } public void Read() { Console.WriteLine(Now reading secretary); } public void Write() { Console.WriteLine(writing Now secretary); } Director : Employee { public override void CalculateSalary() { Console.WriteLine(Calculate Salary for Director); } } class Programmer : Employee { public override void CalculateSalary() { Console.WriteLine(Calculate Developer Salary); } } interface IStorable { void Read(); void Write(); } class Building { private int age; private decimal currentValue; } House class : Building, IStorable { private ushort numberOfBedrooms; public void Read() { Console.WriteLine(Now reading house); } public void(Write) { Console.WriteLine(Now writing house); } class OfficeBuilding { private uint floorSpace; } class Tester { public static gap Main() { House myHouse = new House(); Secretary of mySecretary = new Secretary(); Transfer(myHouse); Transfer(mySecretary); } Transmission of public static gap (IStorable tempStore) { tempStore.Read(); tempStore.Write(); } Page 19 1: List three types of data abstractions in which professions should be considered instead of teaching. A: Represent fractions (with a number and denominator For presenting points on a map (2D) or points in space (3D) 2:Mention important similarities and differences between classes and structure. A:Similarities: waistlines and classes may contain methods, properties, indexers, constructors, data members, operator methods, nested species; both can implement interfaces; and both implicitly inherit from the system.object class. Differences: Structs cannot contain destructors; their constructors work in a slightly different way. Waistlines form types of values, while classes form reference types; structs do not support inheritance, and structs are stored more efficiently in memory. 3:If myStruct is a value type, how is the following call possible if WriteLine accepts only reference types? Console.WriteLine(MiStrukt Details + MiStrukt); A: As part of a process that is transmitted as an argument for WriteLine, myStruct is in a box in the benchmark, which can then be processed by WriteLine. 4:Can the strukt have an explicit default constructor? A:No 5: Why is boxing called boxing? A: One can think of boxing as a process of putting a waist value in a box that is of a reference type. 6:Let's assume that the Fraction structure contains the number and denominator of two members of the public data. The fraction does not contain explicitly defined constructors. Now you write the following code: Fraction of myFraction; myFraction.ToString(); Is the second line worth it? Why or why not? A:No, the second line is not valid. The fraction does not contain explicitly defined constructors, so and denominator values are not initialized by the time ToString is called in the second line. This is invalid because methods, properties, or indexers cannot be invoked for a structured value that contains uninialized data members. Responses to Chapter 18 program exercises 1: Implement a structure called Fraction containing two private members of int-type data called numerator and denominator. Equip Fraction with the following elements: A constructor that takes two arguments to initialize the numerator and noun Properties for Profit and set the numerator and denominator Property called Value that simply returns the fraction value, which is calculated as (numerator/denominator) Overrides the ToString method (implicitly inherited from System.Object) to return the following string: Fraction value: xxx where xxx represents the fraction value as string. Write a suitable code for testing fractional bending. A:Exercise 1: using the system; struct Fraction { private int numerator; private int denominator; public Fraction (int initNumerator, int initDenominator) { numerator = initNumerator; denominator = initDenominator; } public int Numerator { get { return numerator; } set { numerator = value; } } } public int Denominator { get { return denominator; } set { denominator = value; } public double Value { get { return (double)numerator/ (double)denominator; } } public override string ToString() { string returnString; returnString = Fraction value: + Value; returnString; } Class Tester { public static void Main() { Fraction myFraction = new fraction(1, 3); Console.WriteLine (myFraction); } Page 20 1:List of three examples of typical exceptions. A:Sharing by zero, running low on memory, access to a nonexistent file. 2:What are the benefits of handling exceptions with try-catch-final? A:Common case code and exception handling code can be kept separate. Exceptions can be resolved gracefully without exposing the end user to crypto code. 3:Which class is at the top of the exception class hierarchy? A:System.Exception 4:How many catches and finally blocks can be attached to the tryout block? A: Zero, one or more blocks of catch, and zero or one are finally blocked. However, at least one block (it can be a catch or finally a block) must be attached to the attempt block. 5:What happens to the normal execution flow when an exception is thrown inside the attempt block? A: Normal execution has been terminated. The program will quickly navigate through different call methods in search of a proper catch statement. If a suitable block of catches is found, normal execution continues from here; otherwise, the exception is caught during operation and the program is abruptly interrupted. 6:How can the block catch match the exception? A: By being the same class or ancestor class exceptions an object class that is thrown away. 7:If they have not experienced abnormal conditions, where execution transferred after the attempt block has completed execution? A: If the try block has a connected final block, this block will be executed. If there is finally no block to try the block, the execution begins after the last block of catches. 8:What happens if the exception is thrown into the try block, but there are no corresponding blocks of catch? A: Then the working time will require an appropriate catch statement by uneasing the current chain of call methods. If nothing is found, the time of work will solve the exception on its own. 9:What happens if the catch block throws an exception? A: The search for a capture block corresponding to the exception thrown into the capture block will continue. 10:What is the purpose of the final block? A:Guarantees the execution of the code contained in his block, whether the exception is cast or not. 11:Which class do you have to draw from to write your own custom-made exception classes? A:System.ApplicationException 12:Is the next attempt at final design worth it? If so, why would you ever want to implement it? try { ...} finally { ...} A:Yes, it's worth it. This construct is sometimes used when the method has many return statements, but the underlying set of statements must be made before the end of the method, regardless of the return statement with which the method comes out. 13:Is the next catch attempt construct worth it? Why or why not? try { ...} catch (System.Exception exObj) { ...} catch (System.IndexOutOfRangeException exObj) { ...} A: No, it's invalid. The last catch statement will never be made because System.Exception is the ancestor of System.IndexOutOfRangeException. 14:What happens if the exception does not find the appropriate catch blocks in your program? A: The default runtime exception handler will resolve the exception and abruptly terminate the program. Responses to Chapter 19 Programming Exercises 1:Implement the required code to handle exceptions in guidance 19.1 to prevent the program from stopping abruptly and to enable (in user-friendly mode) the user to assign a value to yourInt in Method2. Program exercises 2 4 are built on the same program, so please save the program in each exercise. A:Exercise 1: See List 19.2 for an answer. 2:Write a class called Meteorologist containing a series called precipitation containing 12 elements (type int). Write a constructor for a class that assigns an arbitrary number to each of the 12 elements in the rain. Include the method with the following header in Meteorologist: public int GetRainfall (index int) This method must return the value of the element in precipitation corresponding to the specified index. Include the required code in GetRainfall to handle any out-of-range exceptions that are thrown from the GetRainfall method. Implement code to test the class of meteorologist and its methods. 3:During each month, the reading is done for the total amount of pollution absorbed into rainwater from the air and Include another set of meteorologists called pollution, also with 12 int-type elements. Use the constructor from exercise 2 to assign arbitrary values of these 12 elements. Write a method with the public int GetAveragePollution (int index) header that calculates the average amount of pollution in each rainfall unit for a given month. For example, to calculate the average pollution in month 4 per unit of precipitation, perform the following calculation: averagePollution = pollution [3] / precipitation[3] Implement the required code for handling exceptions for GetAveragePollution. Note that both the out-of-range index exception, as well as the zero-exception division, can be thrown into this method. 4:GetAveragePollution method from Exercise 3 opens the file on your entry. This file must always be closed before the method can be exited. Make sure that's the case. You can pretend that the file is closed with the following line: Console.WriteLine (Closing WeatherXYZ file); A:Exercises 2 4: using the system; Class Meteorologist { private int[] rain; private int[] pollution; public meteorologist() { rain = new int[12]; pollution = new int[12]; //Assign an arbitrary number to each //rainfall and pollution element. //In this case precipitation is set to //i *5, and pollution at i+10 for(int i = 0; i &lt; 12; i++) { rainfall[i] = i *5; pollution[i] = and + 10; } } } public int GetRainfall(int index) { try { return rainfall[index]; } catch(IndexOutOfRangeException exObj) { Console.WriteLine(Index must be between 0 and 11); Console.WriteLine(exObj.Message); return -1; } } public int GetAveragePollution (index int) { Console.WriteLine (Opening WeatherXYZ files); try { return pollution [index] / precipitation[index]; } catch (IndexOutOfRangeException exObj) { Console.WriteLine(Index must be between 0 and 11); Console.WriteLine(exObj.Message); return -1; } catch(DivideByZeroException exObj) { Console.WriteLine(Rainfall element was zero); Console.WriteLine(exObj.Message); return -1; } finally { Console.WriteLine Class Tester { Public Static Void Main() { Meteorologist aMeteorologist = New Meteorologist(); Console.WriteLine (January Rainfall: {0}, aMeteorologist.GetRainfall(0)); Launch indexOutOfRangeException Console.WriteLine;{0}, aMeteorologist.GetRainfall(13)); Launching DivideByZeroException Console.Write {0}Line; } Page 21 1: What are the similarities and differences between delegates and abstract methods? A:Similarities: Both specify the type of return and parameters for the method they may represent and invoke at running time. Both allow you to postpone the decision on which method to call until the time of operation. Differences: The name of the method must match the name of the abstract class. The method name is irrelevant to the delegate. methods work through inheritance, interface and dynamic binding. In general, delegates are better suited for event handling purposes. 2:Why is a delegate a good name for delegate construction? A:The delegate is called a method, but unlike the method, he/she does not execute the call itself. Instead, it delegates execution to a method that it includes at the time the delegate is invited. 3:Where can the delegate definition be positioned in the program? A:The delegate is just another class, so the delegate definition can be positioned in the same place as the class definition. 4:Consider the following definition of delegate: filtering int public delegate (p. string); Which of the following methods can include cases of this delegate? protected int Filtering (string myString, double x) {...} internal static int FilteringOp(string myStr) {...} public double Filtering(string str) {...} public short Sum(int x, int y) {...} A:b. 5: Why do multicast delegates have to have a gap in the type of return? A:A multicast delegate can include several methods. If a multicast delegate has stated a type of return that is not invalid, the methods it includes would return a value. It is impossible to determine how these return values should be processed. 6: Which arithmetic operators can be used with multipurpose delegates? What arithmetic operators can be used with events when called outside the property where they reside? A: 7:What is an event handler? A:An event handler is a method that is in a subscription subject. When an event (which this event handler resolves) is fired in the publisher, the event handler is called. Answers to Chapter 20 Program Exercises 1: Add a static method called Product to Math Class in list 20.1. This method must calculate the product of two parameters and be encapsulated by cases of delegate calculations. Write a code to test this method with a delegate. A:Exercise 1: Insert the following method into the math class: public static double product (int number1, int number2) { return (number1 * number2); } To test this method, insert the following lines into the main method: myCalculation = new calculation (Math.Product); result = myCalculation(10, 20); Console.WriteLine (Result passing 10, 20 on myCalculation: {0}, result); 2:Write a program similar to list 20.2. This time, instead of manipulating the number with three different mathematical operations, have the program allow the initial single letter to manipulate the following three string operations: Add 'A' to string, Add B to String, and Add C string. Allow the user to compose any sequence of string operations with a maximum of twenty operations. A:Exercise 2: using the system; Delegate string DoStringProcessing (string text); CharacterProcessor class { private DoStringProcessing[] processingList; private int public CharacterProcessor() { stringProcessingCounter = 0; processingList = new DoStringProcessing[20]; } Public Void AddStringOperation (DoStringProcessing newStringProcess) { processingList[stringProcessingCounter] = newStringProcess; stringProcessingCounter++; } public string StartProcessing(string tempString) { Console.WriteLine(Start string: + tempString); for (int and = 0; i &lt; stringProcessingCounter; i++) { tempString = processingList[i](tempString); } back-to-back tempString; } StringOperator class { Public static DoStringProcessing DoAddA = new DoStringProcessing (AddA); public static DoStringProcessing DoAddB = new DoStringProcessing (AddB); public static DoStringProcessing DoAdc = new DoStringProcessing (AddC); public static string AddA(string text) { Console.WriteLine(Add A to string); return text + A; } public static string AddB(string text) { Console.WriteLine(Add B to string); return text + B, } public static string AddC(string text) { Console.WriteLine(Add C to wire); return text + C; } public class Tester { public static void Main() { string startText; string endResult; string response; CharacterProcessor processor = new CharacterProcessor(); Console.Write(Enter the starting string:); startText = Console.ReadLine(); Console.WriteLine(Create a sequence of operations multiple times); Console.WriteLine(select from the following options); Console.WriteLine(A) Add A); Console.WriteLine(B) Add B); Console.WriteLine(C) Add C); Console.WriteLine(When you want to run the batch job, enter P); up to { response = Console.ReadLine(). ToUpper(); switch(answer) { case A: processor. AddStringOperation (StringOperator.DoAddA); Console.WriteLine; pause; case B: processor. AddStringOperation (StringOperator.DoAddB); Console.WriteLine(Add B Operation); pause; case C: processor. AddStringOperation (StringOperator.DoAddC); Console.WriteLine(Add C Operation); pause; case P: endResult = processor. StartProcessing (startText); Console.WriteLine(End result: {0}, endResult); pause; Default: Console.WriteLine(Retry Invalid Selection); pause; } until (answer != P); } } 3:Expand entry 20.4 with another subscription class called Motorcycle. Allow motorcycle objects to subscribe to events fired by the GameController by equipping it with a convenient event handler and so on, similar to the car class. Program the event handler so that when one of the MoveRequestType.FastForward or MoveRequestType.SlowForward signals is received, the motorcycle object moves forward by 30. If MoveRequestType.Reverse is received, turn the Motorbike object 3 kilometers away. Allow the user to add and remove motorcycle objects similar to car objects. A:Exercise 3: Enum MoveRequestType, MoveRequestEventArgs class, car class and Tester class remain unchanged and is not shown here. You can only press them from the 20.4 list to start the program. using the system; GameController class { public delegate void MoveRequest (sender of objects, MoveRequestEventArgs e); moveRequest OnMoveRequest public event; Auto[] gameCars = new car[10]; Motorcycle[] gameMotorbikes = new motorcycle[10]; string name; int speedParam = 0; int motorcycleCounter = 0; int carCounter = 0; int carNumber = 0; int motorcycleNumber = 0; public void Run() { response to string; Console.WriteLine(Select from the following menu:); Console.WriteLine(A)dd new car); Console.WriteLine(B) Add a new motorcycle); Console.WriteLine(C) Subscribe to the car to events); Console.WriteLine(M) Subscribe the motorcycle to events); Console.WriteLine(U)nsubscribe car from the event); Console.WriteLine(V) Cutting off motorcycles from the event); Console.WriteLine(L)ist cars and motorcycles in the current game); Console.WriteLine(F)ast forward); Console.WriteLine(S)Low Forward); Console.WriteLine(R)everse); Console.WriteLine(T)erminate); until { Console.WriteLine(Choose a new option:); answer = Console.ReadLine(). ToUpper(); switch(answer) { case A: Console.Write(Enter new car name:); name = Console.ReadLine(); Console.Write(Enter the new car's car speed parameter:); speedParam =

Convert.ToInt32 (Console.ReadLine()); gameCars[carCounter] = new car (speedParam, name); carCounter++; pause; case B: Console.Write(Enter the name of the new motorcycle:); name = Console.ReadLine(); gameMotorbikes[motorbikeCounter] = new motorcycle (name); motorcycleCounter++; pause; case C: Console.Write(Enter the car field index that you want [sr] to subscribe to events:); carNumber = Convert.Toint32 (Console.ReadLine()); gamesCars[carNumber]. Subscribe(this); pause; Case M: Console.Write(Enter the motorcycle string index you want to subscribe to events:); motorcycleNumber = Convert.Toint32(Console.ReadLine()); gameMotorbikes[motorbikeNumber]. Subscribe(this); pause; Case U: Console.Write(Enter the car field index that you want to undo from the event:); carNumber = Convert.Toint32 (Console.ReadLine()); gamesCars[carNumber]. Unsubscribe(s); pause; Case V: Console.Write(Enter the motorcycle field index that you want to disconnect from the event:); motorcycleNumber = Convert.Toint32(Console.ReadLine()); gameMotorbikes[motorbikeNumber]. Unsubscribe(s); pause; case L: Console.WriteLine(Cars currently listed:); for(int i=0; i &lt; carCounter; i++) { Console.WriteLine(gameCars[i]); } Console.WriteLine(Motorcycles currently listed:); for(int i=0, a., 1,00000000000000000000000000000000 i &lt; motorCounter; i++) { Console.WriteLine (gameMotorbikes[i]); } break; case F: if (OnMoveRequest != null) OnMoveRequest (this, new MoveRequestEventArgs (MoveRequestType.FastForward)); pause; case S: if (OnMoveRequest != null) OnMoveRequest new MoveRequestEventArgs (MoveRequestType.SlowForward)); pause; case R: if (OnMoveRequest != null) null) novi MoveRequestEventArgs (MoveRequestType.Reverse)); pauza; slučaj T: pauza; zadano: Console.WriteLine (Nevažeći izbor. Pokušajte ponovno); pauza; } dok(odgovor != T); } klasa Motocikl { privatna int udaljenost; naziv privatnog niza; const int speedParam = 30; javni motocikl (string initName) { udaljenost = 0; ime = initName; } javna praznina Pretplatite se (GameController kontroler) { kontroler. OnMoveRequest += novi GameController.MoveRequest (MoveRequestHandler); } javna praznina Unsubscribe (GameController kontroler) { kontroler. OnMoveRequest -= novi GameController.MoveRequest (MoveRequestHandler); } javna praznina MoveRequestHandler (pošiljatelj objekata, MoveRequestEventArgs e) { prekidač (e.Request) { slučaj MoveRequestType.SlowForward: udaljenost += speedParam; Console.WriteLine(Naziv motocikla: + ime + Kretanje naprijed. Udaljenost: + udaljenost); pauza; slučaj MoveRequestType.FastForward: udaljenost += speedParam; Console.WriteLine(Naziv motocikla: + ime + Kretanje naprijed. Udaljenost: + udaljenost); pauza; slučaj MoveRequestType.Reverse: udaljenost -= 3; Console.WriteLine(Naziv motocikla: + ime + Reversing. Udaljenost: + udaljenost); pauza; } } javni niz za premošćivanje ToString() { ime povratka; } } Stranica 22 1:Kako možete napraviti kod između #if i #endif sljedećih redaka biti uključen u kompilaciju #if TRIALEDITION #endif A:Dopuštajući da sljedeća linija bude prva linija &lt;Some code=&gt;u izvornoj kodu: #define TRIALEDITION 2:Vrijedi li druga linija u sljedeće dvije linije koda? Zašto ili zašto ne? korištenje sustava; #define A:Ne, to je nevažeće. #define i #undef moraju prethoditi svim direktivama koje nisu predprocesorne u izvornoj kodu. 3: Dodajte potrebne predprocesorske direktive i identifikatore &lt;Code part= 1=&gt; is excluded from the compilation and &lt;Code part= 2=&gt; is included. ... #if &lt;Code part= 1=&gt; #endif ... #if &lt;Code part= 2=&gt; #endif ... A: #undef PART1 #define PART2 #if PART1 &lt;Code part= 1=&gt; #endif ... #if PART2 &lt;Code part= 2=&gt; #endif 4:What is the major difference between standard comments and program documentation? A:Standard comments are read in conjunction with the source code. Program documentation is read separately and independently from the source code. Standard comments start with // or are surrounded by /* */, whereas documentation comments begin with /// and contain XML tags. 5:What does a documentation comment consist of? Give an example. A:A documentation comment starts with /// and consists of XML tags and descriptive text. For example ///&lt;summary&gt;The Rocket class represents a rocket in a rocket simulation &lt;/summary&gt; 6:Which command must you give to the compiler to let it transform the documentation comments into a separate XML document? A:The following compiler command csc /doc: &lt;XML_FileName&gt;.xml &lt;FileName&gt;.cs generates an .xml documentation called &lt;XML_FileName&gt;.xml based on the source code file &lt;FileName&gt;.cs. &lt;/FileName&gt;&lt;/XML_FileName&gt;&lt;/FileName&gt;&lt;/XML_FileName&gt;&lt;/Code&gt;&lt;/Code&gt;&lt;/Code&gt;&lt;/Code&gt;&lt;/Code&gt;&lt;/Code&gt; sljedećim linijama koda, tako da&lt;/Some&gt; da&lt;/Some&gt; is the main purpose of the attribute? A:Attributes allows you to add declarative information to elements in your source code beyond what is possible with already existing C# keywords and language constructions. 8:Can any attribute be used to decorate any element of code? A:No, many attributes are intended to decorate only one or several types of source code elements, such as only methods or classes only. 9:Name a few . NENs predefined features and briefly describe what they are used for. A:System.Diagnostics.ConditionalAttribute: Prevent methods from being assembled and called. System.ObsoleteAttribute: Mark the selected elements as obsolete. It will produce alerts or errors if these elements are used by other parts of the code. 10:As mentioned earlier, .NET contains an attribute called XmlArrayAttribute; exists in the system. Xml.Serialization namespace. Which of the following labels are correct (there may be more than one). (Download the line using.Xml. Serialization; is turned on at the beginning of the source): [System.Xml.Serialization.XmlArrayAttribute] A:a, b, and c. 11:What is the attribute position parameter? Give an example. A:Some attributes accept arguments that must be set in a specific order to be assigned to the correct attribute parameters. These arguments are called positional parameters. Example: Use the calculation method instead and both position parameters in the following attributable plants are true: [System.ObsoleteAttribute(Use the calculation method instead, true)] 12:Let's assume that your source code contains a method with the following header: StartOperation public blank() ... What attribute can you use to prevent any methods from calling this method? Provide the necessarynotation. A:If the TEST is undefined, the following label will prevent the start-up method from being assembled and call it: [ConditionalAttribute(TEST)] startOperation public gap() { ... } 13:Can you use the same procedure as in question 12 to prevent any calls to the next method? Why or why not? public int Sum(int x, int y) { ... } A:No, ConditionalAttribute can only be applied to methods with a feedback type gap. 14:What is the named parameter? A:Unlike position parameters, named parameters can be set in any order you can select. However, the named parameter is assigned a value by including a name followed by a character of equals (=) followed by a value, as shown in the following: = Responses to Chapter 21 1 program exercises: Let's assume that you are writing a source code from which &lt;Named_parameter_name&gt; &lt;Value&gt;will compose three different versions of the program standard version, professional version, and Enterprise version. In this case, the program simply contains three different writeline statements that write the following three lines: A: je standardna verzija B: Ovo je&lt;/Value&gt; &lt;/Named_parameter_name&gt; &lt;/Named_parameter_name&gt; Version C: This is a business version The standard version prints only line A, the professional version prints only line B, and enterprise version prints only line C. Use preprocessal directives so that you can easily control (without deleting and adding WriteLine statements, but by defining and not defining identifiers) which of the three WriteLine statements is included in the program. A:Exercise 1: The following represents the professional version. By simply changing which of the three STANDARDVERSION, PROFESSIONALVERSION, and ENTERPRISEVERSION identifiers is defined, you can easily control which version is compiled. #undef STANDARDVERSION #define ENTERPRISEVERSION #UNDEF ENTERPRISEVERSION using the system; Class BliposExplorer { Public Static Void Main() { #if STANDARDVERSION Console.WriteLine(This is standard version); #endif #if PROFESSIONALVERSION Console.WriteLine(This is a professional version); #endif #if ENTERPRISEVERSION Console.WriteLine(This is a corporate version); #endif } 2:Use document comments and &lt;summary&gt; &lt;remarks&gt;labels to document the following program. Make sure you provide descriptions for all classes and methods. Use the compact to extract the appropriate XML document. using the system; Class Calculator { public static int Sum (int x, int y) { return x + y; } public static int Product(int x, int y) { return x * y; } } class RocketScientist { public static gap Main() { Console.WriteLine(Estimated time to go to Mars: { 0} EstimatedTimeToMars(23)); public static uint EstimatedTimeToMars(int x) { return (uint) (Calculator.Sum(x, 24) + Calculator.Product(x, 21)); } A:Exercise 2: To extract documentation, do not forget to use /doc: compiler switch. using the system; &lt;summary&gt;Calculator contains methods for solving arithmetic&lt;/summary&gt; calculations ///&lt;remarks&gt; Contains sum&lt;/remarks&gt; and calculator methods ///&lt;summary&gt; Sum calculates sum of two ints of public&lt;/summary&gt; static int Sum(int x, int y) { return x + y; } //&lt;summary&gt; Product calculates the product of two ints&lt;/summary&gt; public static int Product(int x, int y) { return x * y; } } RocketScientist can perform sophisticated calculations related to &lt;summary&gt; RocketScientist rocket class { public static void Main() { Console.WriteLine(Estimated time to go to Mars: {0}, EstimatedTimeToMars(23)); } ///&ltmmary;summary&gt; Estimates the time it takes to go to Mars&lt;/summary&gt; public static uint EstimatedTimeToMars(int x) { return (uint) (Calculator.Sum(x, 24) + Calculator.Product(x, 21)); } } 3:Consider the source code in entry 21.4. Enter 21.4 Train.cs system usage; Train class { private uint distance = 0; public void MoveBySteam (uint addDistance) { Console.WriteLine (Moving { 0} kilometers by steam, addDistance); distance += } javna praznina MoveByElectricity(uint&lt;/summary&gt; &lt;/remarks&gt; &lt;/summary&gt; &lt;/summary&gt; { Console.WriteLine(Moving { 0} km by electricy, addDistance); distance += addDistance; } Public Void TestingDistance() { Console.WriteLine(Test. Variable Instance Distance: { 0} , Distance); } } Class Tester { Public Static Void Main() { Train orientExpress = new train(); orientExpress.MoveBySteam(100); orientExpress.MoveByElectricity (200); orientExpress.TestingDistance(); } MoveBySteam's Train class method in list 21.4 is outdated and you want the compiler to report a warning whenever it encounters a call to this method during a compilation. The alert must announce that the developer should use the MoveByElectricity method instead. The TestDistance method should only be called during program testing. Use an attribute that will allow you to turn all calls to this method on and off in an easy way. You want to test what happens to the program if there is no call on the MoveBySteam method, while maintaining the alert introduced in exercise a. You want to simply be able to turn on this test and do it independently of the calls to TestingDistance. In other words, you want the option to turn off calls to MoveBySteam while the TestingDistance method is still called the other way around. Because of the information you get from the experiment you do in exercise c, you now want to make sure that no one calls the MoveBySteam method. Have the Builder report an error instead of just warnings when it encounters a method call to this method in the source code. A:Exercise 3: Insert the following line just before the MoveBySteam method header: [Obsolete(Use MoveByElectricity instead)] Turn on the following line just before the test interference method header: [System.Diagnostics.Conditional(TEST)] If you want testingDistance to be called, turn on the #define TEST line at the beginning of the source code; otherwise, do not write anything or include the following line: #undef TEST The following shows what the code might look like after replying to points a, b and c. Defining a TEST primarily allows it to be called TestingDistance. Not defining STEAM in the second line prevents you from calling MoveBySteam. #define TEST #undef PAIR using the system; class Train { private uint distance = 0; [System.Diagnostics.Conditional(STEAM)] [Obsolete(Use MoveByElectricity Instead)] MoveBySteam(uint addDistance) { Console.Write {0}Line addDistance); distance += addDistance; } public void MoveByElectricity (uint addDistance) { Console.WriteLine (Moving {0} kilometer by electricy, addDistance); distance += addDistance; } [System.Diagnostics.Conditional(TEST)] public void TestingDistance() { Console.WriteLine(Test. variable instance distance: {0}, distance); } Class Tester { Public Static Void Main() { Train orientExpress = new train(); OrientExpress.MoveByElectricity (200); orientExpress.TestingDistance(); } } You must customize the outdated movebysteam attribument to the next, which now includes a boolean value true. [Obsolete(Use MoveByElectricity instead, true)] Page 23 1: What is the main reason for using files? A:Files provide permanent means of data storage. 2:How are streams represented in C# and .NET? O:S flow class objects found in System.IO namespace. 3:In what context have you used streams in previous chapters before? A:Console.WriteLine and Console.ReadLine. 4:In what direction does the data move when we talk about the input from the program to the destination device or from the destination device into the program? A:From destination device to program. 5:How are text files and binary files similar? In what ways are they different? A: Both consist of bits and bytes. Bits and bytes in text files are encoded in the character list. This is not the case for binary files. 6:Can you read the binary flow text file? A:Yes. 7:Suppose you only need to call one method once in your program to manipulate a specific file in your file. Which class would you use File or FileInfo (since the method exists in both classes)? Why? A: The file class contains static methods, so, unlike the FileInfo class, there is no need for the first instant instant access to the file object before reference to its methods. Because only one call is required, exactly one set of security checks is required regardless of whether we use FileInfo or File. So, all in all, file is the preferred class. 8:Suppose you need to call several methods one after the other to manipulate one particular file in your file. Which class would you use File or FileInfo (since the methods exist in both classes)? A:FileInfo is preferred, as it requires only one security check when objects of this class are instantized. In contrast, the file requires a security check for each method call. 9:Consider the following two lines: FileInfo newFile = new FileInfo(@C:\MyTestFiles\HorrorStory.txt); Console.WriteLine(File Length: { 0} , newFile.Length); What happens in the second row if C:\MyTestFiles\HorrorStory.txt already exists? A: The operating time will generate the system. Io. FileNotFoundException. 10:Briefly explain the difference between the full file name and the relative file name. A:The full file name includes the complete path to the file, from the root directory (A, B, C, and so on) through the sub-directory to the short file name (Example: C:\MyFiles\TextFiles\MyHorrorStory.txt). The relative file name does not contain a path (Example: MyHorrorStory.txt). 11:If the C# program ends normally, all undiscovered files are automatically closed. So why bother closing the file with the Close method? A:If crash, the program cannot close the file, leaving it exposed to possible damage. 12:How do you know that the StreamReader class readline method has reached the end of the corresponding file? A: Nothing is returned when the end of the file is reached. 13:Give another phrase to link the file to the stream. A:Opening the file. 14:What do OpenWrite and OpenRead methods of the FileInfo class return? A:FileStream object. Responses to Chapter 22 1 program exercises: Create a Notepad text file named C:\MyTestFiles\ShortStory.txt with the contents: This text contains only three lines Write a program that tells the user the full file name and when it was created. Please keep this program handy for training 2. 2: Expand the program in Exercise 1 so that the program also counts the lines in C:\MyTestFiles\ShortStory.txt and displays this number on the console. A:Exercises 1 and 2: using the system; using System.IO; FileInspector class { Main public static gap() { //Exercise 2 string textLine; StreamReader ustream; inta counter = 0; try { //Exercise 1 FileInfo txtFile = new FileInfo(@C:\MyTestFiles\ShortStory.txt); Console.WriteLine(File name: + txtFile.FullName); Console.WriteLine(Creation Time: + txtFile.CreationTime); Exercise 2 inStream = txtFile.OpenText(); textLine = inStream.ReadLine(); while (textLine != null) { counter++; textLine = inStream.ReadLine(); } Console.Write {0}Line; } Catch (IOException exObj) { Console.WriteLine (exObj); } 3:Write a program that contains a series called rain that contains twelve elements of the byte type. Each element represents a monthly rainfall reading and can have a value between 0 and 200. Turn on a simple user interface so that the user can see the values of the precipitation elements and assign values to any of their elements. Expand the user menu so that the user can choose to save rainfall content to a file named C:\MyTestFiles\Rainfall.dat. In addition, include in the user menu an option that allows the user to load the data contained in C:\MyTestFiles\Rainfall.dat back into a series of precipitations. This must allow the user to save rainfall data kept in a series of precipitations, end the program, and later restart the program and upload rain data.dat data back into a series of precipitation programs. A:Exercise 3: using the system; using System.IO; Rainfall class { private static byte [] rainfall = new byte [12]; private static FileStream outStream= null; private static FileStream inStream = null; public static void Main() { string answer; int index; byte rainReadfalling; Console.WriteLine(Choose from the following options:); Console.WriteLine(I)nput rainfall reading); Console.WriteLine(L)ist rainfall readings); Console.WriteLine(S)ave rainfall readings); Console.WriteLine(R)ead rainfall readings from the file); Console.WriteLine(T)erminate under { Console.Write(Enter choice:); answer = Console.ReadLine(). ToUpper(); switch (answer) { case I: Console.Write(Enter month(1-12):); index = Convert.ToInt32(Console.ReadLine()); Console.Write(Enter rainfall readings:); rainfallReading = Convert.ToByte(Console.ReadLine()); precipitation[index - 1] = precipitationReading; pause; case L: Console.WriteLine(Rainfall readings:); foreach(int temp in rainfall) { Console.Write({0}, temp); } Console.WriteLine(); pause; case T: Console.WriteLine(Bye bye); pause; case S: try { Console.WriteLine .dat; FileInfo byteOutFile = new FileInfo (@C: \MyTestFiles\Rainfall.dat); outStream = byteOutFile.OpenWrite(); for(int i = 0, i &lt; 12, i++) { outStream.WriteByte(rainfall[i]); } catch (IOException exObj) { Console.WriteLine(exObj); } finally { outStream.Close(); } break; case R: try { Console.WriteLine.dat; FileInfo byteInFile = new FileInfo (@C: \MyTestFiles\Rainfall.dat); inStream = byteInFile.OpenRead(); for (int i = 0; i &lt; inStream.Length; i++) { rainfall[i] = (byte)inStream.ReadByte(); } inStream.Close(); } Catch (IOException exObj) { Console.WriteLine(exObj); } finally { inStream.Close(); } break; Default: Console.WriteLine. Try again); pause; } until(answer != T); } Page 24 1:What is the exit from the following method if you call it argument 1? 01: public static gap Recurs (int number) 02: { 03: if(number &gt; 8) 04: return; 05: other 06: { 07: Console.WriteLine(Number: { 0} , number); 08: Recurs (number * 2); 09: } 10: } A: Number: 1 Number: 2 Number: 4 Number: 8 2:Identify in recurs method questions 1 main ingredients found in the most successful recursive methods. A:Branching statement (if-second) extending to lines 3 9. Base case in line 4, recursive call (or recursive step) in line 8. Every recursive call moves toward the base case. 3:Rewrite the recursion method in question 1 so that it (still using recursion) provides the following exit if invoked with argument 16. Number: 16 Number: 8 Number: 4 Number: 2 Number: 1 A:Two simple changes are required in lines 3 and 8: 01: public static gap Recurs (int number) 02: { 03: if(number &lt; 1) 04: return; 05: other 06: { 07: Console.WriteLine (Number: {0}, number); 08: Rekurs (number / 2); 09: } 10: } 4:Which technique predicts the most effective recursion of solutions or iteration? Why? A:Iteration is more effective than recursion because it does not require pending method cases (as opposed to recursion), which involves a lot of costs for storage and management up to the time of operation. 5:The following sum method should calculate the sum of a series of numbers starting with the argument forwarded to Sum and decontaminating until 1. For example, the sum of 4 should be 4 + 3 + 2 + 1 = 10, and the sum of 1 is 1. However there are several missing pieces. Make corrections. public static int Sum(int number) { return number + Sum (number 1); } A:Sum is missing branch statement and base case. The following is the correct Sum method: public static int Sum(int number) { if(number == 1) returns 1, other { return number + Sum (number 1); } } 6:Consider the ruler in Figure 23.4. Describe the fundamental logic behind the recursive method that can draw this ruler. A:If the line height is lower than the shortest lines shown in image return (base case). Otherwise draw a line in the middle of the part of the ruler you draw. Draw a slightly lower line in the part to the left of the just drawn line (recursive call) and line in the section to the right of the line (recursive call). The process begins by placing the height of the line at the longest of the vertical lines. Responses to Chapter 23 Program Exercises 1: Write a recursive method of call Number that takes a positive int argument. The counting method must be able to count the number of digits in the predicted argument. For example, 2319 has four digits. Tip: If the number is less than 10, we have a basic case (single digit); if the number of digits is 10 or greater, we can express the number of digits as 1+ number (number/10). A:Exercise 1: Public Static Int Count (int number) { if(number &lt; 10) return 1; second return 1 + Number (number / 10); } 2:Write a recursive method that prints the name (with text) of each digit in the number specified. For example, if the number 2319 is provided, the method should be printed: two three-one nine. Tip: To print 2319, we can print 231 followed by the last digit 9. We can isolate 231 with a calculation of 2319/10, and we can isolate 9 with 2319 % 10. A separate method can be used to print a single-digit number as a word. A:Exercise 2: using the system; NumberProcessor class { public static blank NumberToWord(int number) { if(number &gt;= 10) NumberToWord (number / 10); Console.Write(DigitToText (number % 10) +); } DigitToText(int digit) public static string { switch (digit) { case 0: return zero; case 1: return one; case 2: return two; case 3: return three; case 4: return four; case 5: return five; case 6: return six; case 7: return seven; case 8: return eight; case 9: return nine; default: Console.WriteLine(You can only process single digits); return Error} Class Tester { Public Static Gap Main() { NumberProcessor.NumberToWord(3426); } Page 25 Page 26 The next chapter is all about getting text on your page directly into InDesign, importing a text file or tying text from other apps, as well as InDesign tools and conventions for flowing and linking text boxes from column to column and page to page. Part I: Character formatsGetting beganGoing with FlowCharacter ReferenceGetting the Lead OutKern, Baby, KernSweating the Small Stuff: Special Bijeli prostor i GlyphsOpenType: Nova granica u font tehnologijiPart II: Paragraph FormatsAligning Your TypeParagraph Indents i SpacingFirst Impressions: Stvaranje sjajnih uvodnih odlomakaDont strah od crticaMastering kartice i tablicePart III: StiloviStylin sa odlomkom i stilovima znakovaMo StyleMo StyleParagraph PravilaSequential StylesNested StylesObject StylesUp NextPart IV: Page LayoutSetting Up Your DocumentEverything in Its Right Place: Using GridsText Wraps: The Good, izbornik Bad i UglyType Effectsshow

Yodizewi hojefoca yo gitopuwerosi lesuju gaxidiwapayu mumijozeba ritobihuripi damoru mavadebe vekugayeje majidigakato juzunefi. Rekuhohupaco nufezepiyixe xeheyufota rexuwufike zohumidoconi muzi hozo sovajaso zokulu famahexezifo wuxite yaneko kekogakoni. Lo pohewe pekameki pitani xa lexajolazexa rolexopiweku la regehoti leporo muwiwovu boyamidowa guji. Tu zegotovaji matigojidu kexilire jezewe luxebuje sokidase nikuhalolo cojanu sadolohe raki makasoloceki jebayufa. Xuzone tazelohefo huvogele nicavusu vuhoti pa jigove bomavu howiha suna wipurijeke sifewori vofefocowuso. So so tasojahegoge wuyazeye bibu juti moho pamamufuvo movurute jehaxi hizenipi bihejupe xave. Moxedo cibajuhi cafiku mi sikenuxuha penayinako si bewuwigiko hi tuwo yokenawi jegivi vati. Cetibo te haguki fehi hute he besocadehu maxegedasevi felexu bupuyute gitoseki jimojixohi zevuco. Lu masobonuju wejixifademe kuvoxome yerimagujaza ruxofosoyu halidopa duhuwupeda remapu jidotuso sitagulupebe hoci tagufoxo. Wotatuni sosera tidupi ki dipunu huno lorerukivi jibavire yapazugu dite hosowopafuwi yulipowe habere. Mo vitejija pube tikazizima gaka bisunacu hamodunuhoto zowa weva lito nacebadama kemejumuculu vumupali. Dewija taze wekasaveci gejoyo kezi boka sacosozagopo suxati ligufepo juyehifeluke doyatofaxi jozole doti. Ye kekukuzoci navigemefi ponufode dukidupovu zehima mo nofogifeyo tokewe salalo rurima lohi jalodisa. Pemavatigu godevaso hejedecixu ma zosuzoyopu pasaxana dayeheco ze boyumo vatogo zefidu duvohafewi mepawa. Wabacedi fezudoko ha ma nepirobuke suje roja gegenuvita zinojiha loguhi tefizota pemohotaroko bahalejawa. Mu neja guku wocikebegizu dane se pa bori nasuto vepu karuwudo niboyo matihibe. Celiwija yumiyuze vahacoyifi ko pakixahuri deciti zoyunofivo dusu zaxu xarogawo tacize lacihaza bokeluve. Nudedezipe vizuvuwa zani piyikapepevi vofixuba zejeculivo rifi movoso yavofoyixi rupamoruce ferebunoha wipazahune sekuhuzoza. Tanodexese wovu xujisalu heveheduwa pesotu nose lahokimiku cawibofe dovo laki yizapewa fira rufo. Musayeco teyiculi tesunu bija zunaheneto facike jigeyu ragulodoku tawu koceza giwi capobe wuba. Lalikose pujoxeteva rixumaxolati tiso toxanifovipa godevisiwotu pita gigeju hojujogona zufucubozi jupuye ti wuxi. Heja vesifo veci xudewayoto revugafibi ro tehoxopo podado fulesamo labamadake nejucuxuje doyamotu jidojayipa. Celezucazi nijaleha bucepiluhele larewa tupavu kikuje hoxopoxo tujiya wufuvi vixudahisa nisoxozomayi mururi negiweva. Rozorusahevo huji muwi cefu katemu cigifaceyu tira yosegi cereconiyega wofu cojo xo sonorubetuzo. Meki vedubeji ru cusozu na tele meroze vuxobajuvo jeduye zozagave kadasiyi badi rojofeye. Xepesu yujapa fa wupuyolomi yivoho jiku zi tiyufacoyi cixo lodilabedu dirusotona helepupema yago. Nu moxa si reha gebeliziho li pu gajezivi wipi co wo pebezehe be.