Exercises with solutions

https://csci-1301.github.io/about#authors

September 5, 2023 (05:12:31 PM)

Contents

1 Homework #1

1.1 Part I — Questions

1. List five pieces of software, and three hardware components of a computer.

   Solution

   Accept operating system as a software, cellphone is not a component of a computer.

2. List four programming languages.

   Solution

   C, C++, C#, Java, JavaScript, Python, Haskell, Swift, RWBY, etc (Note: HTML is a "Markup Language" and SQL is a "Query Language", which would technically be incorrect)

3. What is a GUI?

   Solution

   Graphical User Interface.

4. What is a CLI?

   Solution

   Command-Line Interface.

5. What, if any, is the difference between a compiler and an assembler?

   Solution

   The compiler goes from High-level language to Machine language, the assembler goes from Assembly language to Machine language.

6. Is machine code (or the machine language program) is made of circuits, binary numbers, classes, or compilers?

   Solution

   Binary numbers.

7. Give a specific characteristic of C# compared to other programming languages.

   Solution

   It's a compiled language, it has automatic garbage collection, it's object-oriented, it enforces .NET common language specification, etc.

8. What happens when the source code you are giving to the compiler has a syntax error?

   Solution

   The compiler returns an error and fails to compile your code.

9. Is the C# compiler case-sensitive?

   Solution

   Yes, the C# compiler is case-sensitive (meaning that the IDE differentiates between capital and lowercase letters).

10. Suppose I replace every empty space in your source code with two empty spaces. Will your program still compile? Why, or why not?

    Solution

    Yes, because compilers do not care about empty spaces.

11. Give three keywords.

    Solution

    You can consult the "official" list of keywords as https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/. Are valid answers: `int`, `using`, `static`, `public`, etc. Note that `main` or `System` are *not* keywords.

12. Write a statement that would display, “Hi Mom!” (without the quotes) followed by a new line on the screen, once inserted in a proper method, compiled, and executed.

    Solution

    ```
    Console.WriteLine("Hi Mom!");
    ```

13. Write a statement that would display, “Hello!” (without the quotes) followed by a new line on the screen, once inserted in a proper method, compiled, and executed.

    Solution

    ```
    Console.WriteLine("Hello!");
    ```

14. What is the limitation, if any, to the number of methods you can have per class? Why is the method called `Main` special?

    Solution

    A class usually has one or more methods. No limitation, really. It is special because it is the first one to be executed, it is the entry point.

15. What is the difference between a “rule” and a “convention” in programming language?

    Solution

    A rule is required by the programming language and enforced by the compiler. A convention is only a recommendation to make your program easier for other humans to read; you can break it and your code will still work.

16. What is a namespace?

    Solution

    A named collection of related library code.

17. In a C# program, can comments start with \\ (double backslash) or with // (double (forward) slash)? Do they have to end with a ; (semicolon)?

    Solution

    With //, no.

18. Which of the following, if any, are keywords? `Welcome1 public apples int "I’m a string"`

    Solution

    `public`, `int`

19. Which of the following are programmer-defined names (or identifiers)? `BankAccount class apples int itemPerCapit`

    Solution

    `BankAccount`, `apples`, `itemPerCapita`,`statement`.

20. Why are variables called “variables”?

    Solution

    Because their value may change during the program’s execution.

21. What is string interpolation?

    Solution

    The action of inserting a variable into a string that is displayed at the screen.

22. What is the difference, if any, between `12`, and `"12"`?

    Solution

    The first one is an integer, the second is a string.

23. What is the difference, if any, between the `WriteLine` and `Write` methods?

    Solution

    The `Write` method does not advance the cursor to the next line after the message is displayed at the screen.

24. Write a statement that would display the following on the screen: `Hi Mom!↵  How are you doing?`

    Solution

    ```
    Console.Write("Hi Mom!\nHow are you doing?");
    ```

25. Assume we have a variable whose name is `myVariable`, type is `string`, and value is `"My message"`. What would be displayed on the screen by the following statement? `Console.WriteLine($"Here is my variable: {my`

    Solution

    ```
    Here is my variable: My message
    ```

26. Assume we have a variable whose name is `level`, whose type is `string`, and whose value is `"Easy"`. What would be displayed at the screen by the following statement? `Console.WriteLine($"You set the difficulty t`

    Solution

    ```
    You set the difficulty to Easy.
    ```

27. Which of the following are correct identifier names? `$myHome3 class my%variable ANewHope _train _ThisIsAVaria` Is the name `myVariable` the same as `myvariable`? If not, why?

    Solution

    `$myHome3` is correct, but not recommended, `ANewHope`, `_ThisIsAVariable`, `statement` are also correct (but, of course, this latter is weird). The names `myVariable` and `myvariable` are treated differently by C#, because it is case-sensitive.

28. Which of the following are correct identifier names? `myClass _Exo_1 Lab3-Exo1 My.Lab.Variable using Lab3_Part`

    Solution

    Identifiers: `myClass`, `Lab3_Part1`

29. Which of the following are keywords?

    ```
    myClass static Lab3-Exo1 "Hello World" using Lab3_Part1
    ```

    Solution

    Keywords: `static`, `using`

30. Which of the following are correct identifier names?

    ```
    12_Dec_2019 Lab3-Exo1 MyClass2 My.Lab.Variable string My_Var
    ```

    Solution

    ```
    OK: MyClass2, My_Var Not OK: 12_Dec_2019, Lab3-Exo1, My.Lab.Variable, string
    ```

31. Which one(s) of the following, if any, is a correct assignment (assuming that `variable`, `x` and `apples` have been declared as `int` variables)?

    ```
    5 => variable; x=5; apples= 23 x <= 23; variable =1,890;
    ```

    Solution

    Only `x=5;` is correct.

32. Write a statement that assigns the value 23 to a variable `myAge` of type `int`. You do not need to re-declare that variable.

    Solution

    ```
    myAge = 23;
    ```

33. Cross out the wrong answer in the following sentences, [~~like this (incorrect)~~ | like this (correct)]:

    - "If the code does not obey the [rules | conventions] of C#, then the compiler will complain."
    - "Every statement needs to end with [a forward slash / | a semi-colon ;]."
    - "C# is a [object-oriented | functional] programming language."
    - "A class is made up of [a body and a header | multiple `using` statements]."
    - "An identifier can contain [only lower-case letters | letters and digits]."

    Solution

    - If the code does not obey the **rules** of C#, then the compiler will complain.
    - Every statement needs to end with a **semi-colon** ;.
    - C# is an **object-oriented** programming language.
    - A class is made up of **a body and a header**.
    - An identifier can contain **letters and digits**.

1.2 Part II – Problems

1. There are 4 errors in the following code that will prevent it from compiling. Can you spot them all?

    ```
    // My first attempt.
    using System
    class Wel
    {
        static void Main();
        {
            ConsoleWriteLine("Welcome \n to the lab!");
        }
    }
    ```

    Solution

    a) Missing ; after `using System`
    b) superfluous ; after `static void Main()`
    c) missing . between `Console` and `WriteLine`
    d) missing closing } for `class Wel`

2 Homework #2

2.1 Part I — Questions

1. In C#, what is the "escape character"? Why is it useful?

   Solution

   \, to print "special characters", like new line, tabulation, quotations mark, etc.

2. Write a statement that *initializes* a variable named `myHeightInMeters` to your height in meters. What should be the datatype of `myHeightInMeters`?

   Solution

   `float myHeightInMeters = 1.68f` The datatype should be any floating-point datatype.

3. Suppose you replace every `*` in your program with the `!` symbol. Are you completely sure that your program would still compile? Why or why not?

   Solution

   No, because `/*` will be transformed into `/!`.

4. Give the values of `a` and `b` after the following four instructions have been executed.

   ```
   int a, b;
   a = 2;
   b = a * 2 + 1;
   a -= 1;
   ```

   Solution

   `a` is `1`, `b` is `5`.

5. Give the values of `a` and `b` after the following four instructions have been executed.

   ```
   int a, b;
   a = 4;
   b = a * 3 + 1;
   a /= 2;
   ```

   Solution

   `a` is `2`, `b` is `13`.

6. Give the values of `c` and `d` after the following four instructions have been executed.

   ```
   int c = 3, d;
   d = 2 + c;
   c = d * 2;
   d += 2;
   ```

   Solution

   `c` is `10`, `d` is `7`.

7. Is there an error in the following code? Explain the error or give the value of `b` after the second statement is executed.

   ```
   float a = 3.7f;
   int b = (int)a;
   ```

Solution

No error; 3

8. Is there an error in the following code? Explain the error or give the value of `b` after the second statement is executed.

```
decimal a = 1.6M;
int b = (int)a + a;
```

Solution

Error, the result type of the operation is a decimal, and cannot be stored in an int.

9. There is an error in the following code, at the second line. Explain the error, and how you could fix this second line using a cast operator, without changing the datatype of the `b` variable.

```
decimal a = 2.5M;
int b = a / 2;
```

Solution

The result of the expression `a / 2` is of type `decimal`, so it cannot be stored in an int. This can be fixed by adding an `(int)` cast to the variable `a` before dividing it by 2, so that the expression's result type is `int`, like this: `int b = (int)a / 2;`

10. If one of the operator's operands is of type `float` and the other is of type `int`, what will be the type of the result of the operation?

Solution

`float`

11. What is the return type of the operation `12.4 * 3`?

Solution

`double`

12. Write an explicit conversion from a `double` variable `myDoubleVar` to an `int` variable called `myIntVar`. You do not need to re-declare those variables. Assuming `myDoubleVar`'s value is 5.89, what value would be stored in `myIntVar`?

Solution

`myIntVar = (int)myDoubleVar;`, the value stored in `myIntVar` would be 5.

13. Write a statement that performs an implicit conversion between two different numeric datatypes.

Solution

```
float m = 3;
```

14. Assuming that `myLastName` and `myFirstName` are two `string` variables that have been initialized, write a statement that *concatenates* them with a space and a comma in-between, and assign the resulting `string` to a variable named `fullName`. For instance, if the value of `myLastName` is `"Holbertonand"`, and the value of `myFirstName` is `"Betty"`, then the value of `fullName` after your operation should be `"Holbertonand, Betty"`.

Solution

```
string fullName = myLastName + ", " + myFirstName;
```

15. In C#, what is the name of the method used to read input from the user?

    Solution

    `ReadLine()`.

16. What is wrong with the following code? Will the error(s) appear at compilation time or at execution time?

```
int age;
Console.WriteLine("Please enter your age:");
age = Console.ReadLine();
```

    Solution

    Conversion of `string` to `int`. At compilation time.

17. Will those statements, if placed in a proper `Main` method, compile? Could this program crash at execution time? Justify your answer.

```
int myAge;
Console.WriteLine("Please enter your age:");
myAge = int.Parse(Console.ReadLine());
```

    Solution

    The code will compile, but can crash at execution time if the end user inputs a non-integer value.

18. Write a series of statements that: a) Declare an `int` variable named `userAge`, b) Display on the screen a message asking the user to enter his or her age, c) Read the value entered by the user and store it in the `userAge` variable. You can add statement(s) performing intermediate steps if you want.

    Solution

```
int userAge;
Console.WriteLine("Please enter your age.");
userAge = int.Parse(Console.ReadLine());
```

    Note: This is not the only solution, but the most optimal.

19. Write a series of statements that: a) Declare an `string` variable named `favoriteColor`; b) Display on the screen a message asking the user to enter his or her favorite color; c) Read the value entered by the user and store it in the `favoriteColor` variable. You can combine some of the statement(s) if you want, but do not display at the screen any information that was not explicitely asked.

    Solution

```
string favoriteColor;
Console.WriteLine("Please enter your favorite color.");
favoriteColor = Console.ReadLine();
```

    Note: This is not the only solution, but the most optimal.

2.2 Part II – Problems

The following three exercises **do not** require a computer. Make sure you feel ready before starting them, try to do them with limited time and without notes, and, if you want, check your answer using your IDE.

1. This problem restates the content of the Order of Operations[1] section of the lecture notes and ask you to answer various problems.

   There are 5 different arithmetic operations available in C#:

Operation	Arithmetic Operator	Algebraic Expression	Expression
Addition	`+`	$x + 7$	`myVar + 7`
Subtraction	`-`	$x - 7$	`myVar - 7`
Multiplication	`*`	$x \times 7$	`myVar * 7`
Division	`/`	$x/7$, or $x \div 7$	`myVar / 7`
Remainder (a.k.a. modulo)	`%`	$x \bmod 7$	`myVar % 7`

   Computing operations involving one of them is straightforward:

Operation	Result
`3 + 4`	`7`
`3 - 4`	`-1`
`3 * 4`	`12`
`6 / 2`	`3`
`6 % 4`	`2`

   But things can get complicated when multiple operators are used, but no parenthesis are indicated. For instance, should

   ```
   7 / 2 - 4 * 8 % 3
   ```

   be read as

$$\begin{aligned}(7 \div 2) - ((4 \times 8) \bmod 3) &= 3.5 - (32 \bmod 3)\\ &= 3.5 - 2\\ &= 1.5\end{aligned}$$

   or as

$$\begin{aligned}(7 \div (2 - 4)) \times (8 \bmod 3) &= (7 \div (-2)) \times 2\\ &= (-3.5) \times 2\\ &= -7\end{aligned}$$

   Certainly, the result is not the same and there are other possibile ways this calculation may be performed!

   Actually, C# uses the following three rules:

   a) `*`, `/`, and `%`, called the "multiplicative operations," are always evaluated before `+` and `-`, called the "additive operations." So that, for instance,

[1] https://csci-1301.github.io/book.html#order-of-operations

```
2 - 4 * 8
```

will be evaluated as $2 - (4 * 8) = -30$.

b) If there are multiple operations of the same type, they are evaluated from left to right. For instance,

```
4 / 2 * 8
```

will be evaluated as $(4 \div 2) \times 8 = 16$ and

```
4 - 2 + 8
```

will be evaluated as $(4 - 2) + 8 = 10$.

c) Parenthesis can be used to force a particular order of evaluation, so that `2 * (3 + 4)` will be evaluated as $2 \times (3 + 4) = 2 \times 7 = 14$, not as $(2 * 3) + 4 = 6 + 4 = 10$ as it would without the parenthesis.

Answer the following:

a) Which of the following operation(s) compute the arithmetic expression $(x \times (3 \bmod 5)) - (y \times 7)$?

i. `x * 3 % 5 - y * 7`
ii. `x * (3 % 5) - y * 7`
iii. `(x * 3) % 5 - y * 7`
iv. `x * 3 % (5 - y * 7)`
v. `(x * 3 % 5) - (y * 7)`
vi. `(x * ((3 % 5) - (y * 7))`

b) State the order of evaluation of the operators in each of the following operations, and compute the resulting value:

i. `8 - 39 * 1 / 12 + 5`
ii. `12 + -23 / 12 % 3`
iii. `90 * 23 / 34 - 12 - 13`
iv. `12 % 83 - 2 * 3`

c) (Optional) Check your answers using your IDE. You can use a statement of the form:

```
Console.WriteLine($"8 - 39 * 1 / 12 + 5 is {8 - 39 * 1 / 12 + 5}");
```

2. Write down, on a piece of paper, a fully compilable program that initializes an `int` variable named `persons` with the value 5, an `int` variable named `bottles` with the value 3, and a `double` variable named `literPerBottlle` with the value 1.5. What should be the type of the `literPerPerson` variable to be able to be assigned the number of liters every person is going to get, if split equitably? Write the correct initialization of that variable and a statement that displays its value.

   Place a delimited comment with a your name and the time at which you wrote the program at the top of the program.

3. Write down, on a piece of paper, a program that:

   a) Declares a `string` variable named `userName`.
   b) Displays on the screen "Please enter your name, followed by enter:".
   c) Reads a `string` value from the keyboard and assigns the value to the `userName` variable.
   d) Declares an `int` variable named `number`.
   e) Displays on the screen "Please enter your number:".
   f) Reads an `int` value from the keyboard and assigns the value to the `number` variable.
   g) Declares a `string` variable named `id` and initializes it with the string referenced by the `userName` variable, followed by the number entered by the user (you can concatenate a string and an int using the + sign).

h) Displays on the screen, "Your id is" and the content of the `id` variable.

Here is an example of execution, where the user input is underlined, and hitting "enter" is represented by ↵:

```
Please enter your name, followed by enter.
Jaylah↵
Please enter your area code, followed by enter.
49391↵
Your id is Jaylah49391
Press any key to continue . . .
```

3 Homework #3

3.1 Part I – Questions

1. What is "an instance of a class"?

   Solution

   An object.

2. Fill in the blanks: "A class asserts that every objects created using it should have_____ (i.e.,"data") and_____ (i.e., "operations")."

   Solution

   Attributes and methods.

3. Give two access modifiers.

   Solution

   `public` and `private`.

4. What, if any, is the difference between a parameter and an argument?

   Solution

   "Parameters" are called "formal parameters" and are used when defining the method, while "arguments" are called "actual parameters" and are used when calling the method. A method has parameters and takes arguments.

5. Write a statement that creates a new object from the `Rectangle` class.

   Solution

   ```
   Rectangle myRec = new Rectangle();
   ```

6. What is the purpose of the keyword `new`?

   Solution

   To create objects, to instantiate classes.

7. Do different objects from the same class share their instance variables?

   Solution

   No, there is one instance variable per object.

8. Briefly explain the difference between a local variable and an instance variable.

   Solution

   A local variable exists only within a single method; instance variables are available/accessible to all methods in a class.

9. Suppose we have a `Circle` class containing

```
public void SetRadius(double radiusArgument)
{
    radius = radiusArgument;
}
```

   Write a statement that create a `Circle` object, and one statement that sets its radius to 3.5.

   Solution

```
Circle theCircle = new Circle();
theCircle.SetRadius(3.5);
```

10. Indicate the order of evaluation of the operators in each of the following C# operations by adding parenthesis or developping the expression one step at a time, and compute the resulting value:

    a) `3 * 4 - 2`
    b) `3 % 2 + 3`
    c) `2 - 3 + 3 * 2`
    d) `2 + 2 * 1 - 4`

11. Write the complete implementation of a class that contains two attributes (with different data types), a setter for one attribute, a getter for the other attribute, a custom constructor, and a `ToString` method. You can re-use an example from a lecture or a lab, as long as it satisfies those conditions, or you can invent one. No need to write an application program.

    Solution

    Note: In this solution, the Rectangle class from the lab was used, but any class fulfilling the requirements will work.

```
class Rectangle
{
    private int length;
    private int width;

    public Rectangle(int lengthP, int widthP)
    {
        length = lengthP;
        width = widthP;
    }
    public void SetLength(int lengthParameter)
    {
        length = lengthParameter;
    }
    public int GetWidth()
    {
        return width;
    }
    public override string ToString()
    {
```

```
        return "Length: " + length + ", Width: " + width;
    }
}
```

12. What does the keyword `return` do?

    Solution

    It is used by a method to return a value to the environment that called it.

13. Write a get method for an instance variable named `total` of type `int`.

    Solution

    ```
    public int GetTotal()
    {
        return total;
    }
    ```

14. Write a getter for an attribute of type `string` named `myName`.

    Solution

    ```
    public string GetMyName()
    {
        return myName;
    }
    ```

15. Write a setter for an attribute of type `int` named `myAge`.

    Solution

    ```
    public void SetMyAge(int paramMyAge)
    {
        myAge = paramMyAge;
    }
    ```

16. Assuming `name` is a `string` instance variable, there is problem with the following setter. Fix it.

    ```
    public int SetName(string val){
        name = val;
    }
    ```

    Solution

    It has the wrong return type, because it does not (and should not) return a value, i.e., replace `int` with `void`.

17. Assume we have an instance of the `Rectangle` class named `myRect` and an instance of the `Circle` class named `myCircle`. Write statement(s) that will make the radius of `myCircle` equal to the width of `myRect`.

    Solution

    ```
    myCircle.SetRadius(myRect.GetWidth());
    ```

18. Briefly describe what a format specifier is. Write a statement that uses one.

    Solution

    An indication to format a numerical value in a special way in a string. For example, `Console.WriteLine($"{65536:N");` will display "65,536.00" on the screen after applying the `:N` format specifier to the value 65536.

19. Write a statement that uses a format specifier.

    Solution

    ```
    Console.WriteLine($"myInt:C");
    ```

20. Write a method for the `Rectangle` class that divides the length and width of the calling object by a factor given as a parameter.

    Solution

    ```
    public void DivideBy(int factor)
    {
        length /= factor;
        width /= factor;
    }
    ```

21. Draw the UML diagram of a class named "Student" with a single attribute, "name", of type `string`, and two methods, `SetName` and `GetName`.

    Solution

Student
- name: string + SetName(arg: string) + GetName(): string

22. Write a `ToString` method for a `Account` class with two attributes, a `string` attribute called `name` and a `decimal` attribute called `amount`.

    Solution

    ```
    public override string ToString()
    {
        return $"Account Name: {name}\nAccount Balance: {amount}";
    }
    ```

23. Consider the following UML diagram:

Circle
- radius : float + setRadius(radiusParam : float) : void + getRadius(): float + getArea(): float

    What is the name of the class, what are the methods and attributes of the class?

    Solution

    The class is named Circle, it has a single attribute (radius) and three methods (setRadius, getRadius and getArea).

24. What does it mean to say that instance variables have a default initial value? How is that different from the variables we have been manipulating in the `Main` method?

    Solution

    When we create an object, the instance variable get a default value. When we declare a variable, it is unassigned.

25. Is it possible to have more than one constructor defined for a class? If yes, how can C# know which one is called?

    Solution

    Yes, by looking at the signature.

26. What is the name of a constructor method? What is the return type of a constructor?

    Solution

    The name of the class. It does not have a return type, not even `void`.

27. Write a constructor for a `Soda` class with one `string` attribute called `name`.

    Solution

    ```
    public Soda()
    {
    name = "Generic";
    }
    ```

28. Assume we have a `Polygon` class, that have only one attribute, an `int` called `numberOfSides`. Write a constructor for that class.

    Solution

    ```
    public Polygon (int numberOfSidesParam)
    {
        numberOfSides = numberOfSidesParam;
    }
    ```

29. What is the "default" constructor? Do we always have the possibility of using it?

    Solution

    The constructor provided with the class by default, that set all the attributes to their default values. If we define our own constructor, this one disappears.

30. What is the return type of a `ToString` method? How many arguments does it usually take?

    Solution

    `string`; 0.

31. Consider the following partial class definition:

    ```
    public class Book
    {
        private string title;
        private string author;
        private string publisher;
        private int copiesSold;
    }
    ```

    a) Write a statement that would create a `Book` object.

b) Write a "getter" and a "setter" for the `title` attribute.
c) Write a constructor for the `Book` class taking at least one argument (you are free to decide which one(s)).

Solution

a) `Book myBook = new Book();` 2 and 3.

```
public string GetTitle()
{
    return title;
}

public void SetTitle(string titleP)
{
    title = titleP;
}

public Book (string titleP, string authorP, string pubP, int copiesP)
{
    title = titleP;
    author = authorP;
    publisher = pubP;
    copiesSold = copiesP
}
```

32. Consider the following partial class definition:

```
class DVD
{
    private string title;
    private decimal price;
}
```

a) Write a "setter" for the `title` attribute.
b) Write a constructor for the `DVD` class that takes two arguments.
c) Write a method called `Discount` that decreases the `price` attribute by 20.55%.
d) Write a (good, informative) `ToString` method for the class.
e) Write statements that ask the user to enter a price and then create a `DVD` object with a `price` attribute equal to the price the user entered. (The object's `title` attribute can be anything you choose).
f) Draw the UML class diagram for the class you obtained by adding the above four methods to our original class definition.

Solution

```
class DVD
{
    private string title;
    private decimal price;

    public void SetTitle(string titleP)
    {
        title = titleP;
    }

    public DVD(string titleP, decimal priceP)
```

```
    {
        title = titleP;
        price = priceP;
    }

    public void Discount()
    {
        price -= price * 0.2055
    }

    public override string ToString()
    {
        return $"DVD Title: {title}\nDVD Price: {price}"
    }
}

//program.cs
Console.WriteLine("What should the price be for the movie "Ender's Game?");
DVD myDVD = new DVD("Ender's Game", decimal.Parse(Console.ReadLine()));
```

DVD
- title : string
- price : decimal
+ SetTitle(titleP : string) : void
+ <> DVD(titleP : string, priceP : double)
+ Discount() : void
+ ToString() : string

Note: This is not the only answer, but is the most optimal.

33. Consider the following partial class definition:

```
class Book
{
    private string title;
    private decimal price;
}
```

a) Write a "getter" for the `title` attribute.
b) Write a constructor for the `Book` class that takes two arguments.
c) Write a method called `AddTaxes` that increases the `price` attribute by 6.35%.
d) Write a (good, informative) `ToString` method for that class.
e) Write statements that ask the user to enter a price and then create a `Book` object with a `price` attribute equal to the price the user entered. (The object's `title` attribute can be anything you choose).
f) Draw the UML class diagram for the class you obtained by adding the above four methods to our original class definition.

Solution

```
class Book
{
    private string title;
    private decimal price;
```

```
    public string GetTitle() { return title; }

    public Book(string titleP, decimal priceP)
    {
        title = titleP;
        price = priceP;
    }

    public void AddTaxes()
    {
        price *= 1.0635m;
    }

    public override string ToString()
    {
        return $"Book Title: {title}, Price: {price}.";
    }
}

//program.cs
Console.WriteLine("What should the price be for the book "Ender's Game?");
Book myBook = new Book("Ender's Game", decimal.Parse(Console.ReadLine()));
```

Book
- title : string
- price : decimal
+ GetTitle() : string
+ <> Book(titleP : string, priceP : double
+ AddTaxes() : void
+ ToString() : string

Note: This is not the only answer, but is the most optimal.

34. Assume that my `Pet` class contains one custom constructor:

```
public Pet(string nameP, char genderP){
    name = nameP;
    gender = genderP;
}
```

What is the problem with the following statement?

```
Pet myPet = new Pet('M', "Bob");
```

Solution

The method has for parameters a `string` and a `char`, but the problematic statement calls it with a `char` and a `string`: the order of the arguments is wrong, it should be `Pet myPet = new Pet("Bob", 'M');`.

35. Why would one want to define a constructor for a class?

Solution

To be able to set the instance variables directly when creating the objects.

3.2 Part II – Problems

There is only one problem this time, and it is harder than what you'll be asked to do during the exam. Being able to solve it is an excellent sign that you are ready.

1. You are going to design a class named `Triangle`. A triangle has three angles, but knowing the value of only two angles is sufficient to determine the value of the third, since they always add up to 180°. Hence, it is sufficient to have only two `double` attributes, `angle1` and `angle2`. We want to define several methods:

   - a no-arg constructor that sets the value of `angle1` to 60.0 and the value of `angle2` to 60.0,
   - another constructor, that takes two arguments, and assigns to `angle1` the value of the first argument, and assigns to `angle2` the value of the second argument,
   - getters for `angle1` and `angle2`,
   - a method that computes and returns the value of the third angle, that we name `ComputeAngle3`,
   - a method that rotate the triangle: the value of the first angle should be replaced with the value of the second angle, and the value of the second angle should be replaced with the value of the third angle.

   a) Write the UML diagram for the `Triangle` class.
   b) Write the full, compilable implementation of the `Triangle` class.

   Solution

Triangle
- angle1: double - angle2: double + Triangle() + Triangle(angle1P: double, angle2P : double) + GetAngle1(): double + GetAngle2(): double + ComputeAngle3(): double + Rotate(): void

```
class Triangle
{
    private double angle1;
    private double angle2;
    public Triangle()
    {
        angle1 = 60;
        angle2 = 60;
    }
    public Triangle(double angle1Param, double angle2Param)
    {
        angle1 = angle1Param;
        angle2 = angle2Param;
    }
    public double GetAngle1()
    {
        return angle1;
    }
    public double GetAngle2()
    {
```

```
        return angle2;
    }
    public double ComputeAngle3()
    {
        return 180 - angle1 - angle2;
    }
    public void Rotate()
    {
        double oldAngle3 = ComputeAngle3();
        angle1 = angle2;
        angle2 = oldAngle3;
    }
}
```

4 Homework #4

4.1 Part I — Questions

1. What is sequential processing?

   Solution

   When the code is executed sequentially, without any branching. It implies that the code is processed in the order in which it is presented in the source code: the statement at line n will always be executed after the statement at line $n-1$ and before the statement at line $n+1$.

2. What is a decision structure?

   Solution

   A decision structure is a test and one or multiple statement blocks that may or may not be executed based on the outcome of the test. Selection and iteration are two examples of decision structures: in the first one, a statement block can be "skipped over" if the test evaluates to false, in the second one, a statement block can be repeated multiple times, as long as the test evaluates to true. A decision structure makes it possible to have portions of the code executed conditionally.

3. Decide if the following boolean expressions will evaluate to `true` or `false`:

   a) `3 > 2.0 && false`
   b) `(4 != 3) ||  false`
   c) `'A' == 'b' && ! false`
   d) `(! false) == (true ||  4 == 3)`

   Solution

   i. `false`
   ii. `true`
   iii. `false`
   iv. `true`

4. Decide if the following Boolean expressions will evaluate to `true` or `false`:

   a) `3 > 2.0 || true`
   b) `(4 != 3) && false`
   c) `'A' == 'b' || ! false`
   d) `(! true) == (true ||  4 != 3)`

   Solution

   i. `true`

ii. `false`
iii. `true`
iv. `false`

5. For each of the following Boolean expressions, decide if it will evaluate to `true` or `false`:

   a) `('y' == 'Y') && true`
   b) `6 + 2 < 8 ||  3 > 4`
   c) `(true && 4 == 3) == false`
   d) `4 > 4 && !false`

   Solution

   i. `false`
   ii. `false`
   iii. `true`
   iv. `false`

6. For each of the following Boolean expressions, decide if it will evaluate to `true` or `false`:

   a) `('y' != 'Y') && true`
   b) `6 + 2 < 12 ||  3 > 4`
   c) `(true && 4 >= 3) == false`
   d) `13 <= 4 * 3 ||  !false`

   Solution

   i. `true`
   ii. `true`
   iii. `false`
   iv. `true`

7. What relational operator is used to determine whenever two values are different?

   Solution

   `!=`

8. How do you store the result of a Boolean expression?

   Solution

   In a variable of type `bool`. Its two possible values are `true` and `false`.

9. Give three relational operators, and then two logical operators.

   Solution

   Possible answers: `<=`, `>=`, `==`, `!=`, `>`, `<` and `!`, `&&`, `||`

10. What would be displayed on the screen by the following code?

```
if (false)
{
    Console.WriteLine("Hello!");
}
Console.WriteLine("Hi!");
```

    Solution

    "Hi!"

11. Is there a simpler way to write the expression `over21 == true`, assuming that `over21` is a Boolean variable?

    Solution

    We can simply write `over21`, which will always evaluate to the same value as `over21 == true`.

12. Assume that `x` and `y` are two `int` variables that have already been initialized (i.e., declared and assigned), write an `if` statement that assigns 10 to `x` if `y` is (strictly) greater than 5.

    Solution

    ```
    if (y > 5) x = 10;
    ```

13. In C#, is there a difference between = and ==? Write a statement that uses =.

    Solution

    Yes, one equal sign serves to write assignment operator, and two equal signs serve to compare. An example of statement that uses comparison first and assignment second could be: `if (x == 9)x = 12;`

14. Assuming a `name` string was declared and initialized with a value given by the user, write an `if` statement that displays "I have the same name!" if `name` contains your first name.

    Solution

    ```
    if(name=="Clément")
        Console.WriteLine("I have the same name!");
    ```

15. Is the following statement correct, i.e., would it compile, assuming `myFlag` is a `bool` variable, and `myAge` is an initialized `int` variable?

    ```
    if ( myAge > 20 )
    {
        myFlag = true
    };
    ```

    Solution

    No, the semi-colon should come before the closing brace.

16. If we write a statement that begins with `if(false)`, then the IDE returns a warning, "Unreachable code detected". What does it mean?

    Solution

    The IDE is warning us that the statements in the block will never be executed.

17. Write an `if` statement that prints "Bonjour !" if the value of the `char` variable `lang` is `'f'`.

    Solution

    ```
    if (lang == 'f')
        Console.WriteLine("Bonjour !");
    ```

18. For each of the following boolean expressions, decide if it will evaluate to `true` or `false` when the boolean variables `x`, `y` and `z` are all set to `true`:

    - `x || y && z`
    - `!x || y && z`
    - `!(x || y) && (z && y)`
    - `(!x && x) || (!x || x)`

Do the same when they are all set to `false`.

Solution

For each expression, we give first the value if all the variables are set to `true`, then the value if all the variables are set to `false`.

- `true`, `false`
- `true`, `true`
- `false`, `false`
- `true`, `true`

You can check the answer using for instance the following code.

```
using System;
class MainClass {
    public static void Main (string[] args) {
        bool x, y, z;
        x = true;
        y = true;
        z = true;
        Console.WriteLine(x || y && z);
        Console.WriteLine(!x || y && z);
        Console.WriteLine(!(x || y) && (z && y));
        Console.WriteLine ((!x && x) || (!x || x));
        x = false;
        y = false;
        z = false;
        Console.WriteLine(x || y && z);
        Console.WriteLine(!x || y && z);
        Console.WriteLine(!(x || y) && (z && y));
        Console.WriteLine ((!x && x) || (!x || x));
    }
}
```

19. Write a `boolean` expression that evaluates to `true` if a variable `x` is between 3 (excluded) and 5 (included).

    Solution

    ```
    x>3 && 5>=x
    ```

20. Write an `if-else` statement that assigns 0.1 to `z` if `y` is greater or equal than 0, and that assigns -0.1 to `z` otherwise.

    Solution

    ```
    if(y >= 0){
        z = 0.1;
    }
    else{
        z = -0.1;
    }
    ```

21. Write an `if-else` statement that assigns `"Minor"` to an already declared `string` variable `legalStatus` if `age` is strictly less than 18, and that assigns `"Major"` to `legalStatus` otherwise.

    Solution

```
if(age < 18){
    legalStatus = "Minor";
}
else{
    legalStatus = "Major";
}
```

22. Write an `if-else` statement that displays "It's free for you!" if an `int` variable `age` is between 0 and 18, and "It's $5.00." otherwise.

    Solution

```
if(age <= 18 && age >= 0){
    Console.WriteLine("It's free for you!");
}
else{
    Console.WriteLine($"It's {5M:C}.");
}
```

23. Assume we initialized an `int` variable called `courseNumber` and a `string` variable called `courseCode`. Write a series of statements that will display:

    a) "I'm taking this class!" if `courseNumber` is 1301 and `courseCode` is `CSCI`;
    b) "That's my major!" if `courseCode` is `CSCI`;
    c) "Is that an elective?" if `courseNumber` is greater than 3000; or
    d) "Is it a good class?" otherwise.

    Your program should display exactly one message.

    Solution

```
if(courseNumber == 1301 && courseCode == "CSCI")
{
    Console.WriteLine("I'm taking this class!");
}
else if(courseCode == "CSCI")
{
    Console.WriteLine("That's my major!");
}
else if(courseNumber > 3000)
{
    Console.WriteLine("Is that an elective?");
}
else
{
    Console.WriteLine("Is it a good class?");
}
```

24. Assume we previously initialized an `int` variable called `graduationYear` and a `string` variable called `graduationSemester`. Write a series of statements that will display:

    a) "I will graduate at the same time!" if `graduationYear` is 2023 and `graduationSemester` is `Fall`;
    b) "I love this season." if `graduationSemester` is `Spring`;
    c) "That is in a long time!" if `graduationYear` is greater than 2025; or
    d) "I hope you'll have an in-person ceremony!" otherwise.

    Your program should display exactly one message.

    Solution

```
if(graduationYear == 2023 && graduationSemester == "Fall")
{
    Console.WriteLine("I will graduate at the same time!");
}
else if(graduationSemester == "Spring")
{
    Console.WriteLine("I love this season.");
}
else if(graduationSemester > 2025)
{
    Console.WriteLine("That's in a long time!");
}
else
{
    Console.WriteLine("I hope you'll have an in-person ceremony!");
}
```

25. Assume we previously initialized a `char` variable called `myChar`. Write a series of statements that will display if the character is…

    a) Uppercase
    b) Lowercase
    c) A number
    d) or none of those.

    Your program should display exactly one message. Bonus: Make your message also display the ASCII value of the character.

    Solution

```
if ((myChar >= 'a') && (myChar <= 'z'))
{
    Console.WriteLine($"Your character ({(int)myChar}) is lower-case!");
}
else if ((myChar >= 'A') && (myChar <= 'Z'))
{
    Console.WriteLine($"Your character ({(int)myChar}) is upper-case!");
}
else if ((myChar >= '0') && (myChar <= '9'))
{
    Console.WriteLine($"Your character ({(int)myChar}) is a number!");
}
else
{
    Console.WriteLine($"Your character ({(int)myChar}) is not a letter or number!");
}
```

26. What will be displayed on the screen by the following program?

```
int x = 3, y = 2, z = 4;
if (x > y) {z += y;}
if (x > z) {y -= 4;}
Console.WriteLine($"x is {x}, y is {y}, and z is {z}.");
```

    Solution

    "x is 3, y is 2, and z is 6."

27. What will be displayed on the screen by the following program?

```
int x = 3, y = 2, z = 4;
if (x >= z) {z += y;} else if (x != y) {z *= y;}
y -= 4;
Console.WriteLine($"x is {x}, y is {y}, and z is {z}.");
```

Solution

"x is 3, y is -2, and z is 8."

28. (*We'll use the 24-hour clock, sometimes called "military time".*) Assuming that an `int` variable `hour` has been initialized, write part of a program that would display on the screen "Good morning" if `hours` is less than or equal to 12, and "Hello" otherwise.

Solution

```
if (hours <= 12)
{
    Console.WriteLine("Good morning!");
}
else
{
    Console.WriteLine(("Hello");
}
```

29. Assuming that `myString` is a string variable, write a statement that prints "Hello, Melody!" if the value of `myString` is equal to `Melody`, and nothing otherwise.

Solution

```
if (myString == "Melody")
{
    Console.WriteLine("Hello, Melody!");
}
```

30. What will be displayed on the screen by the following program?

```
int x = 3, y = 2, z = 4;
if (y >= z) {z += y;}
else if (x != y) { if (false) {z -= 3;} else {z += x;}}
Console.WriteLine($"x is {x}, y is {y}, and z is {z}.");
```

Solution

x is 3, y is 2, and z is 7.

31. Rewrite, if possible, the three following `if-else-if` statements as `switch` statements:

```
if (myLang == 'f')
{
    Console.WriteLine("Vous parlez Français ?");
}
else if (myLang == 'e')
{
    Console.WriteLine("Do you speak English?");
}
else if (myLang == 'd')
{
    Console.WriteLine("Sprechen Sie Deutsch?");
```

```
}
else
{
    Console.WriteLine("I do not know your language!");
}

if (myCity == "Augusta")
{
    Console.WriteLine("I also live here!");
}
else if (myCity == "Ithaca" || myCity == "Providence")
{
    Console.WriteLine("I used to live there!");
    }
else
{
    Console.WriteLine("I never lived there.");
}

if (temp == 100.0)
{
    Console.WriteLine("It's ready!");
}
else if (temp >= 90.0)
{
    Console.WriteLine("Almost ready!");
}
else
{
    Console.WriteLine("You have to wait.");
}
```

If you think it is not possible or not feasible, explain why.

Solution

```
switch (myLang)
{
    case 'f':
        Console.WriteLine("Vous parlez Français ?");
    break;
    case 'e':
        Console.WriteLine("Do you speak English?");
    break;
    case 'd':
        Console.WriteLine("Sprechen Sie Deutsch?");
    break;
    default:
        Console.WriteLine("I do not know your language!");
    break;
}

switch (myCity)
{
    case "Augusta":
        Console.WriteLine("I also live here!");
```

```
        break;
        case "Ithaca":
        case "Providence":
            Console.WriteLine("I used to live there!");
        break;
        default:
            Console.WriteLine("I never lived there.");
        break;
    }
```

The last one is impossible, since we cannot write 'switch' statements comparing all the possible `float` values!

32. Give an example of an `if` statement that could not be rewritten as a `switch`.

    Solution

    Any condition not making a simple comparison is a good attempt. For instance, trying to convert `if(age % 2 == 0){Console.WriteLine("Your age is even.");.` into a `switch` would require to list all the even values, which is not realistic.

33. Write a `switch` statement that sets a `double discount` variable to `0.5` if a string `day` variable contains `"Monday"` or `"Wednesday"`, `0.25` if `day` contains `"Saturday"`, and `0.5` otherwise.

    Solution

```
switch (day)
{
    case "Saturday":
    {
        discount = 0.25;
        break;
    }
    default:
    {
        discount = 0.5;
        break;
    }
}
```

4.2 Part II – Problems

This time, the two exercises **do not** require a computer, and are here to craft on your problem-solving skills. Make sure you feel ready before starting them, try to do them with a limited amount of time and without notes, and check your answer using your IDE.

1. Write a program that asks the user to write a country name and stores the user's input into a string variable. Then, compare that string with `"france"`: if it is equal, then display at the screen `"Bienvenue en France !"`. Then, compare that string with `"usa"`: if it is equal, then display at the screen `"Welcome to the US!"`. If the string is different from both `"france"` and `"usa"`, then display at the screen `"Welcome to"` followed by the name of the country the user typed in. Can you think of two ways to implement this program, one using `if-else-if` statements, the other using `switch`?

    Solution

```
Console.WriteLine("Country?");
string c = Console.ReadLine();
switch (c)
{
    case "usa":
        Console.WriteLine("Welcome to the US!");
    break;
    case "fr":
        Console.Write("Bienvenue en France!");
    break;
    default:
        Console.WriteLine($"Welcome to {c}");
    break;
}
```

2. You want to write a small program for an on-line printing company. Your program should ask the user to chose a format (10×15 centimeters, or 8×11 inches), ask if it is the first time the customer order through your company, and a number of copies. Then, calculate the total cost of printing those pictures, knowing that

   - Printing a 10×15 centimeters picture costs \$0.20, printing a 8×11 inches picture costs \$0.25,
   - A new customer gets a \$3 coupon if the order is more than \$5,
   - A 10% discount is given if more than 50 copies were ordered,
   - The two previous offers can be cumulated.

   Display on the screen a message starting by "Welcome!", then a new line, then "We cherish our new customers" if it is the first time the user uses your company, ", so we're giving you a \$3 discount!" if the user is allowed to get the coupon, then print the total and "You had a 10% discount!" if the user ordered more than 50 copies. See below for examples of execution, where the user input is underlined, and hitting carriage return is represented by ↵.

   ```
   Enter 'c' for 10x15cm, anything else for 8x11in.
   c ↵
   Is this your first time here? Type 'y' for 'yes'.
   y ↵
   Enter a number of copies.
   90 ↵
   Welcome!
   We cherish our new customers, so we are giving you a $3 discount!
   Your total is $13.50. You had a 10% discount!
   ```

   ```
   Enter 'c' for 10x15cm, anything else for 8x11in.
   p ↵
   Is this your first time here? Type 'y' for 'yes'.
   Not_at_all↵
   Enter a number of copies.
   120 ↵
   Your total is $27.00. You had a 10% discount!
   ```

   Solution

   ```
   Console.WriteLine("Enter 'c' for 10x15cm, anything else for 8x11in");
   char sizeChoice = char.Parse(Console.ReadLine());
   Console.WriteLine("Is this your first time here? Type 'y' for 'yes'.");
   char firstTime = char.Parse(Console.ReadLine());
   Console.WriteLine("Enter a number of copies");
   ```

```
int copies = int.Parse(Console.ReadLine());
decimal totalCost = sizeChoice == 'c' ? copies * 0.2m : copies * 0.25m;
if(firstTime == 'y')
{
    Console.WriteLine("Welcome!");
    string message = "We cherish our new customers";
    if(totalCost > 5m)
    {
        totalCost -= 3m;
        message += ", so we're giving you a $3 discount!";
    }
    Console.WriteLine(message);
}
if(copies > 50)
{
    totalCost -= totalCost * 0.1m;
    Console.WriteLine($"Your total is {totalCost:C}. You had a 10% discount!");
}
else
{
    Console.WriteLine($"Your total is {totalCost:C}.");
}
```

5 Homework #5

5.1 Part I — Questions

1. Assume you are given an un-assigned `string` variable `letterGrade`, and an already assigned `float` variable `numberGrade`. Write a small program that assigns `"A"` to `letterGrade` if `numberGrade` is between 100 and 90 (both included), `"B"` to `letterGrade` if `numberGrade` is between 90 (excluded) and 80 (included), etc., and `"Invalid data"` if `numberGrade` is strictly lower than 0 or strictly greater than 100. Should you use a **`switch`** statement or a **`if`** …**`else if`** …**`else`**?

   Solution

   An **`if`** …**`else if`** …**`else`** is the right structure for this task:

   ```
   float numberGrade;
   string letterGrade;
   numberGrade = -60; // This is just an example, feel free to change it.
   if(numberGrade > 100 || numberGrade < 0)
   {
   // It's actually easier to get rid of the "invalid" cases first.
       letterGrade = "Invalid Data";
   }
   else if (numberGrade >= 90)
   {
       letterGrade = "A";
   }
   else if(numberGrade >= 80)
   {
       letterGrade = "B";
   }
   ```

```
else if(numberGrade >= 70)
{
    letterGrade = "C";
}
else if(numberGrade >= 60)
{
    letterGrade = "D";
}
else
{
// We know the value is greater than 0 but strictly lower than 60.
    letterGrade = "F";
}
Console.WriteLine(numberGrade + " corresponds to " + letterGrade);
```

2. Given an `int` variable `counter`, write three statements to decrement its value by 1.

   Solution

   We actually know four ways to do that:

   ```
   counter = counter - 1;
   counter -= 1;
   counter--;
   --counter;
   ```

3. What will be displayed on the screen?

   ```
   int x = 3, y = 7;
   Console.WriteLine (x++ +" and "+ --y);
   ```

   Solution

   "3 and 6"

4. What will be displayed on the screen by the following program?

   ```
   int counter = 2;
   while (counter != 5)
   {
       Console.Write(counter + "\n");
       counter++;
   }
   ```

   Solution

   ```
   2
   3
   4
   ```

5. What will be displayed on the screen by the following program? Write the spaces and new line explicitly.

   ```
   int counter = 10;
   while (counter > 5)
   {
       counter--;
       Console.Write(counter + "\n");
       if (counter == 7) {
           Console.WriteLine("Bingo");
   ```

```
    }
}
```

Solution

```
9
8
7
Bingo
6
5
```

6. What will be displayed on the screen by the following program?

```
int counter = 10;
while (counter != 5) ;
Console.Write(counter + "\n");
counter--;
```

Solution

Nothing, and it will loop forever.

7. What will be displayed on the screen by the following program?

```
int counter = 7;
while (counter != 2)
    Console.Write(counter + "\n");
counter--;
```

Solution

7 infinitely many times.

8. What is input validation? Name a control structure that can be used to perform it. Why is it important?

Solution

Making sure the user's input is valid. The `while` loop. Because we cannot trust the user.

9. What do we name a variable that is increased by some value at every iteration of a loop, i.e., that keeps the running total?

Solution

An accumulator.

10. What is a sentinel value?

Solution

It's a value that will trigger an exit from a loop. It is a value that was agreed on, and that signifies "I now want to exit the loop.".

11. Write a program that asks the user to enter a value between 0 and 10, and asks again as long as the user enters integers outside that range.

Solution

```
int answer;
do{
    Console.WriteLine("Enter a value between 0 and 10 (both included).");
    answer = int.Parse(Console.ReadLine());
}while(answer > 10 || answer < 0);
```

12. Write a small program that asks the user for an integer, and displays "It is positive" if the number entered is positive, "It is negative" if the number entered is negative, and "Not a number" if the user entered a string that is not an integer.

    Solution

```
int answer;
Console.WriteLine("Enter an integer");
if(!int.TryParse(Console.ReadLine(), out answer)){
    Console.WriteLine("Not a number");
}
else if (answer > 0){
    Console.WriteLine("Positive");
}
else{
    Console.WriteLine("Negative");
}
```

13. Write a program containing a `while` loop that would display the numbers between -100 and 100 (both included) with a space between them when executed.

    Solution

```
int counter = -100;
while(counter <= 100){
    Console.Write(counter++ + " ");
}
```

14. Assume you are given an initialized `string` variable `name`, and a `string` variable `field`. Write a small program that assigns to `field`

    - "CS" if `name` is "Turing" or "Liskov",
    - "Math." if `name` is "Aryabhata" or "Noether",
    - "Unknown" otherwise.

    Solution

```
string name;
name = "Turing"; // Value given as an example, change it to test.
string field;
switch(name){
    case("Turing"):
    case("Liskov"):
        field = "CS";
        break;
    case("Aryabhata"):
    case("Noether"):
        field = "Math.";
        break;
    default:
        field = "Unknown";
        break;
```

```
}
Console.WriteLine(name + " worked in " + field + ".");
```

15. Write a program that asks the user to enter a value between 1900 and 1999 (both included), and asks again as long as the user enters integers outside that range.

    Solution

```
int answer;
do{
    Console.WriteLine("Enter a value between 1900 and 1999 (both included).");
    answer = int.Parse(Console.ReadLine());
}while(answer < 1900 || answer > 1999);
```

5.2 Part II – Problems

1. Write a `switch` statement that calculates the number of days in a particular month. You should assume that you are given already assigned `month` and `year` `int` variables, and that your program should set an already declared `int` `numberOfDays` variable to 28, 29, 30 or 31 depending on the month / year combination. Your program should start with a `switch` matching `month` against certain values, and, if `month` is 2, uses an `if` statement to decide whenever the number of days is 28 or 29. You can use something like

```
switch (month) {
    …
    case (2):
        if …
        …
        break;
    …
}
```

    Solution

```
int month = 2;
int year = 2000;
int numDays = 0;
switch (month) {
    case 1: case 3: case 5:
    case 7: case 8: case 10:
    case 12:
        numDays = 31;
    break;
    case 4: case 6:
    case 9: case 11:
        numDays = 30;
    break;
    case 2:
        if (((year % 4 == 0) && !(year % 100 == 0))
            || (year % 400 == 0))
            numDays = 29;
        else
            numDays = 28;
    break;
    default:
```

```
        Console.WriteLine("Invalid month.");
    break;
}
Console.WriteLine("Number of Days = " + numDays);
```

6 Homework #6

6.1 Part I — Questions

1. Write a statement that creates a 10-element `int` array named `numbers`.

   Solution

   ```
   int[] numbers = new int[10];
   ```

2. In the following, what is the value of the size declarator? What is the value of the index?

   ```
   int[] numbers;
   numbers = new int[8];
   numbers[4] = 9;
   ```

   Solution

   The size declarator is 8, the subscript, or index, is 4.

3. What is wrong with the following array declaration?

   ```
   int[] books = new int[-1];
   ```

   Solution

   The size declarator cannot be negative.

4. Draw the content of the `scores` array once those statements have been executed.

   ```
   int[] scores = new int[3];
   scores[0] = 13;
   scores[2] = 25;
   ```

   Solution

index	0	1	2
value	13	0	25

5. What will be displayed on the screen by the following program?

   ```
   for (int num = 3 ; num <= 5 ; num++)
       Console.Write(num + " ");
   ```

   Solution

   ```
   3 4 5
   ```

6. Write a 'for' loop that displays on the screen the sequence "1, 2, 3, 4, 5, 6, 7, 8, 9, 10, ".

   Solution

```
for (int x = 1 ; x <= 10 ; x ++)
    Console.Write(x + ", ");
```

7. Write a 'for' loop that displays on the screen the sequence "1, 2, 3, 4, 5, 6, 7, 8, 9, 10", (note that there is no comma after 10).

   Solution

```
for (int x = 1 ; x <= 10 ; x ++)
{
    Console.Write(x);
    if (x < 10) Console.Write(" ,");
}
```

8. Write a 'for' loop that displays on the screen the sequence "1 3 5 7 9 ".

   Solution

```
for (int x = 1 ; x <= 10 ; x+= 2)
    Console.Write(x + " ");
```

9. Given an `int` variable `myVar` initialized with a positive value, write a loop that sums the integers between 0 and `myVar` (i.e., $0 + 1 + \cdots + (\text{myVar} - 1) + \text{myVar}$).

   Solution

```
int sum = 0;
for (int x = 1 ; x <= myVar ; x++)
    sum += x;
```

10. Consider the following code:

```
for (int y = 1; y <= 3; y++)
{
    for (int z = 1; z < 5; z++)
        Console.Write("Scene " + y + ", take " + z + ". " );
    Console.WriteLine();
}
```

    How many times does the outer loop iterate (i.e., how many scenes are shot)? How many times does the inner loop iterate (i.e., how many takes for each scene)? Finally, what is the total number of iterations of the nested loops (i.e., how many takes are made, total)?

    Solution

    3, 4, 12.

11. Which of the following are pre-test loops? **`do while`**, **`switch`**, **`while`**, **`for`** and **`if-else-if`**.

    Solution

    `for` and **`while`** are pretest loops.

12. What will be displayed on the screen by the following code?

```
int[] values = new int[6];
for (int i = 0 ; i < 6 ; i++)
    values[i] = (i*2);
foreach (int j in values)
    Console.WriteLine(j);
```

    Solution

```
0
2
4
6
8
10
```

13. Suppose we are given an `int` array `dailyPushUp` with 7 elements. Write a piece of code that display the value of the elements stored in the array `dailyPushUp`.

    Solution

    ```
    for (int j = 0 ; j < 7 ; j++)
        Console.WriteLine(dailyPushUp[j]);
    ```

14. What is "array bounds checking"? When does it happen?

    Solution

    It is C# making sure that you are not using a subscript outside the allowed range. It happens at run-time.

15. Is there an error with the following code? If you think there is one, explain it, otherwise draw the content of the `myIncomes` array once those statements have been executed.

    ```
    double[] myIncomes = new double[5];
    myIncomes[1] = 3.5;
    // No income on day two.
    myIncomes[3] = 5.8;
    myIncomes[4] = 0.5;
    myIncomes[5] = 1.5;
    ```

    Solution

    The subscripts are off, they should go from 0 to 4: `myIncomes[5] = 1.5;` will casue an error.

16. What would be the size of the `test` array after the following statement has been executed?

    ```
    int[] test = {3, 5, 7, 0, 9};
    ```

    Solution

    5

17. What is the difference, if any, between the following two statements?

    ```
    int[] num, scores;
    int num[], scores;
    ```

    Solution

    In the second one, only `num` is a reference to an `int` array, and `scores` is just an `int`.

18. Write a statement that creates and initializes a `double` array with the values 12.5, 89.0 and 3.24.

    Solution

    ```
    double[] question = {12.5, 89.0, 3.24};
    ```

19. What is the value of `count` and the content of `number` once the following has been executed?

```
int count=2;
int[] number={3, 5, 7};
number[count--] = 8;
number[count]--;
```

Solution

count is 1. numbers is 3, 4, 8.

20. Suppose we have an array named `temp` that has been declared and initialized. How can we get the number of elements in this array?

    Solution

    By using the `Length` field: `temp.Length`.

21. Describe what the following code would do.

```
int[] record = { 3, 8, 11 };
int accumulator = 0;
foreach (int i in record)
    accumulator += i;
```

    Solution

    Declare and initialize an `int` array with the values 3, 8 and 11, and then sum those values in an accumulator variable.

22. Assuming we have two `int` arrays of the same size, `firstA` and `secondA`, write a program that copies the content of `firstA` into `secondA`.

    Solution

```
for (int k = 0 ; k < firstA.Length ; k++)
   secondA[k] = firstA[k];
```

23. Assuming we are given an `int` array named `arrayF`, write a program that adds one to each of its elements. That is, if `arrayF` contains 3, 5, 7 and -2 before your program is executed, it should then contain 4, 6, 8 and -1 after your program was executed.

    Solution

```
for (int k = 0 ; k < arrayF.Length ; k++)
    arrayF[k] += 1;
```

24. Assuming we are given an `int` array named `arrayF`, write a program that displays the product of its elements. That is, if `arrayF` contains 2, 3 and -1, then your program should display -6.

    Solution

```
int prod = 1;
for (int k = 0 ; k < arrayF.Length ; k++)
    prod *= arrayF[k];
Console.WriteLine(prod);
```

25. Write a static method (header included) that takes as argument an `int` array, and display on the screen the value of each element of that array.

    Solution

```
public static void p(int[] a){
    foreach (int k in a)Console.WriteLine(k);
}
```

26. Write a static method (header included) that takes as argument an `int` array, and stores the value 10 in each element of that array.

    Solution

```
public static void z(int[] a){
    for (int j = 0 ; j < a.length ; j++) a[j]=10;
}
```

7 Quizzes

Those quizzes are given as *examples*, to help you practise. They were given at week 4 and 7.

7.1 Quiz 1

1. (3 pts) List three keywords.

   Solution

   Keywords are words that have special meanings and cannot be used as identifiers in your program. Examples of C# Keywords include `int`, `string`, `if`, `while`, `void`, `else`, `bool` etc.

2. (4pts) Circle the correct identifiers:

   - `%Rate`
   - `static`
   - `my-variable`
   - `User.Input`
   - `YoUrNaMe21`
   - `test_train`
   - `_myIdentifier`

   Solution

   Valid identifiers can not include reserved words, can not start with a number, and must contain only numbers, letters, and/or the underscore character. The identifiers below follow these rules. `my-variable` `YoUrNaMe21` `test_train` `_myIdentifier`

3. (4 pts) For each of the following, indicate if they are a "rule" of C# or a "convention" between programers by ticking the appropriate column. The first answer is given as an example.

Statement	Rule	Convention
Code should be commented.		✓
Case matters.		
Variable names should be descriptive.		
Keywords cannot be used as identifiers.		
Each ".cs" file should contain exactly one class.		

   Solution

Statement	Rule	Convention
Case matters.	✓	
Variable names should be descriptive.		✓

Statement	Rule	Convention
Keywords cannot be used as identifiers.	✓	
Each ".cs" file should contain exactly one class.		✓

4. (4 pts) Write a statement that would display, "Hi Mom!" (*with* the quotes) followed by a new line on the screen.

   Solution

   The key is to escpae the " character: `Console.WriteLine("\"Hi Mom!\"\n")`

5. (5 pts) Write a series of statements that would

   a) declare an `int` variable called "myAge",
   b) assign your age to that variable,
   c) display "My age is", the value of the "myAge" variable, a period, and finally a new line.

   Solution

   ```
   int myAge;
   myAge = 21;
   Console.WriteLine("My age is " + myAge + ".\n");
   ```

6. (Bonus) Give examples of situations where the adage "*Spaces and new lines do not matter in programs*" is actually erroneous.

   Solution

   Spaces and newlines matter when they are used in string data, as blank space in strings is formatted exactly how it's typed. Blank spaces also matters in between words: words in C# must have at least one space between them in order to be compiled correctly (e.g., `static void Main()` and `int days = 7;`). If there were no spaces in either of the examples, neither of them would compile. They also matter for in-line comments.

   ```
   // My comment
   int x;
   x = 10;
   ```

   If you remove the first newline, the program would not compile.

7.2 Quiz 2

1. (2 pts) What is the relational operator used to determine whenever two values are equal?

   Solution

   The operator is two equal signs, ==, not to be confused with a single equal sign, =, used for assignment.

2. (2 pts) Write a `boolean` expression that evaluates to `true` if a variable `yourName` is different from `"Marc"` and `"Mark"`.

   Solution

   A possible solution is: `yourName != "Marc" && yourName != "Mark"`. Note that there was no need to write an `if` statement or a complete statement: only the expression was required.

3. (5pts) Write a `boolean` expression that evaluates to `true` if a variable `x` is between -10 (excluded) and 10 (included).

   Solution

   A possible solution is: `x > -10 && x < 10`

   Note that there was no need to write an `if` statement or a complete statement: onyl the expression was required.

4. (3 pts) What will be displayed on the screen by the following program?

```
int x = 5, y = 1, z = 2;
if (x != z || y < z ) {x += y;}
if (z > y) {if (z % 2 == 0) {z -= 3;} else {z += x;}} else {x = y + z;}
Console.WriteLine($"x is {x}, y is {y}, and z is {z}.");
```

   Solution

   This program would display:

```
x is 6, y is 1, and z is -1.

Press any key to continue...
```

5. (8 pts) Assume we initialized a `string` variable named `month` and a `double` variable named `temperature`. Write a series of statements that will display *exactly* one of the following messages: "What a nice summer day!" if `month` is "July" and `temperature` is less than 90 (included); "Better wear a jacket" if `temperature` is between 45 and 60 (both included); "Happy holidays!" if `month` is "December"; or "Have a nice day" otherwise.

   Solution

```
if(month=="July" && temperature<=90)
    Console.WriteLine("What a nice summer day!");
else if (temperature >= 45 && temperature <= 60)
    Console.WriteLine("Better wear a jacket!");
else if (month == "December")
    Console.WriteLine("Happy Holidays!");
else
    Console.WriteLine("Have a nice day!")
```

6. (Bonus) Give a program that displays "Leap year" if a `year` variable is

   a) divisible by 4; and
   b) not divisible by 100, unless it is also divisible by 400.

   Your program should correctly identify 2000 and 2400 as leap years, and 1800, 1900, 2100, 2200, 2300, or 2500 as *not* leap years.

   Solution

```
if (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0))
    {
        Console.WriteLine("The year " + year + " is a leap year.");
    }
else
    {
        Console.WriteLine("The year " + year + " is not a leap year.");
    }
```

7.3 Quiz 3

1. (7 pts) Write a `switch` statement that sets a `decimal` `price` variable to 1.50 if a string variable `iceCreamFlavor` contains `"vanilla"`, 1.75 if it contains `"chocolate"` or `"mint"`, and -1 otherwise.

   Solution

   ```
   switch(iceCreamFlavor)
   {
       case "vanilla":
           price = 1.50m;
           break;
       case "chocolate":
       case "mint":
           price = 1.75m;
           break;
       default:
           price = -1;
           break;
   }
   ```

2. (2 pts) Write a statement that applies the increment operator in prefix position to a variable `test`.

   Solution

   The increment operator (++) in **pre**fix position (that is, applied *before* anything else) applied to the variable `test` gives: `++test;`.

3. (3 pts) What will be displayed on the screen by the following program?

   ```
   int x = 10;
   x--;
   while(x > 0){
       Console.Write(x);
   }
   ```

   Solution

   This would display `9` (without new line or space) forever.

4. (8 pts) Write a `while` loop that displays "1 2 3 4 5 6 7 8 9 10 11 12 13" (spaces included!) at the screen.

   Solution

   There are numerous different ways of writing such a program. A compact one could be:

   ```
   int num = 1;
   while(num <= 13)Console.Write($"{num++} ");
   ```

   while a more verbose could be:

   ```
   int num = 1;
   while(num <= 13){
       Console.Write($"{num} ");
       num++;
   }
   ```

5. (+3+2 pt, bonus) Give a program that displays every multiple of 3 between 0 and 10,000 (that is, "0 3 6 9 12 … 9999"). Bonus (again!): display how many such numbers there are in total.

   Solution

   We can either

   a) Display the counter only if it is a multiple of three:

```
int counter = 0;
while(counter <= 1000){
    if (counter % 3 == 0){Console.Write(counter + " ");}
    counter++;
}
```

   b) Increment the counter 3 by 3:

```
int counter = 0;
while(counter <= 1000){
    Console.Write(counter + " ");
    counter+=3;
}
```

   To answer the "bonus within the bonus", we need two counters:

```
int threeCtr = 0; // Counter for the multiple of 3.
int multipleCtr = 0; // Counter that sum the number of values displayed
while(threeCtr < 10000)
{
    Console.Write($"{threeCtr} ");
    threeCtr += 3; // We move to the next multiple of 3.
    multipleCtr++; // We increment the number of values displayed.
}

Console.WriteLine($"\nThere are {multipleCtr} multiples of 3 from 0 to 10,000.");
```