

ASSIGNMENT 1 FRONT SHEET

Qualification	BTEC Level 5 HND Diploma in Computing		
Unit number and title	Unit 1: Programming		
Submission date	03/07/2020	Date Received 1st submission	
Re-submission Date		Date Received 2nd submission	
Student Name	Nguyen Quoc Viet	Student ID	GCC18157
Class	GCC0801	Assessor name	Duong Trung Nghia
Student declaration I certify that the assignment submission is entirely my own work and I fully understand the consequences of plagiarism. I understand that making a false declaration is a form of malpractice.			
		Student's signature	Viet Nguyen

Grading grid

P1	M1	D1

<input type="checkbox"/> Summative Feedback: 	<input type="checkbox"/> Resubmission Feedback: 	
Grade:	Assessor Signature:	Date:
Internal Verifier's Comments:		
Lecturer Signature:		

ASSIGNMENT 1 BRIEF

Qualification	BTEC Level 5 HND Diploma in Computing
Unit number and Title	Unit 1: Programming
Assignment title	Problem solving with algorithms
Academic Year	

Unit Tutor			
Issue date		Submission date	

Submission Format:	
Format:	The submission is in the form of an individual written report and a presentation. This should be written in a concise, formal business style using single spacing and font size 12. You are required to make use of headings, paragraphs and subsections as appropriate, and all work must be supported with research and referenced using the Harvard referencing system. Please also provide a bibliography using the Harvard referencing system.
Submission	Students are compulsory to submit the assignment in due date and in a way requested by the Tutors. The form of submission will be a soft copy in PDF posted on corresponding course of http://cms.greenwich.edu.vn/
Note:	The Assignment <i>must</i> be your own work, and not copied by or from another student or from books etc. If you use ideas, quotes or data (such as diagrams) from books, journals or other sources, you must reference your sources, using the Harvard style. Make sure that you know how to reference properly, and that understand the guidelines on plagiarism. <i>If you do not, you definitely get fail</i>
Assignment Brief and Guidance:	
<p>Scenario: You have applied for a post as a trainee with a software development company and have been invited for an interview. You have been asked to demonstrate your problem solving and basic programming skills. To do this you have to prepare a report on using algorithms to solve problems.</p> <p>You need to explain, using examples, how algorithms are used to solve simple business problems and the steps needed to be followed to produce a working program solution. You should make clear your assumption about your program. The problems to be solved will involve basic procedural programming instructions - sequence instructions (input, output and assignment statements), loops, conditional statements. Problems should be analysed and designed by the use of flowchart and demonstrated by the use of modules (procedures) using a menu based program.</p> <p>Tasks:</p> <ol style="list-style-type: none">1. State your simple business problems to be solved.2. Analyse the problem and design the solutions by the use of suitable methods.3. Demonstrate the compilation and running of a menu-based program4. Evaluate how the problem is solved from the designed algorithm to the execution program written by a specific programming language. <p>You also need to do a presentation of your work (it should be summary of your report).</p>	

Learning Outcomes and Assessment Criteria		
Pass	Merit	Distinction
LO1 Define basic algorithms to carry out an operation and outline the process of programming an application		
P1 Provide a definition of what an algorithm is and outline the process in building an application .	M1 Determine the steps taken from writing code to execution .	D1 Examine the implementation of an algorithm in a suitable language. Evaluate the relationship between the written algorithm and the code variant.

Contents

<i>Question 1: Describe and explain the idea of inheritance in objected oriented programming by an example codes. Save as question_1.sln</i>	4
<i>Question 2: Describe and explain the idea of constructor in objected oriented programming by an example codes. Save as question_2.sln</i>	6
<i>Question 3: FractionsSum Given an integer n, write a method that returns sum of series $1 + (\frac{1}{2})^2 + (\frac{1}{3})^2 + \dots + (\frac{1}{n})^2$. Do not use library function! Expected input and output FractionsSum(3) → 1.361111111111 FractionsSum(5) → 1.463611111111 Explain your solution. Save as question_3.sln</i>	7
<i>Question 5: Conversion Write a C# program taking one input value from the user, choosing the type of conversion and then performing a conversion depending on the type of conversion. For example, if the type of conversion is: I then the task is to convert from inches to centimeters. M then the task is to convert from mile to kilometer. P then the task is to convert from pound to kilogram. Given that: 1 inch is equal to 2.54cm 1 mile is equal to 1.6 km 1 pound is equal to 0.45kg</i>	11
<i>Reference:</i>	12

Question 1: Describe and explain the idea of inheritance in objected oriented programming by an example codes. Save as question_1.sln

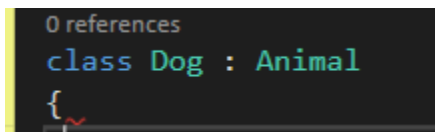
- **What is Object-Oriented Programming?**

- Object-Oriented Programming (OOP) is an object-oriented programming technique which forms the basis for today's advanced programming languages. The power of object-oriented programming is code reuse, which provides a basic layout of the software modules, hides the internal information, enabling programmers to simplify the lesson's complexity.

C # language is the most advanced object-oriented language today, it inherits the strengths of object-oriented programming languages C + + and Java, and excludes the complexities of object-oriented programming, too. For example, it removes from these two languages multiple inheritance in C++, or it allows operators to override that Java does not have ...

- *Class*

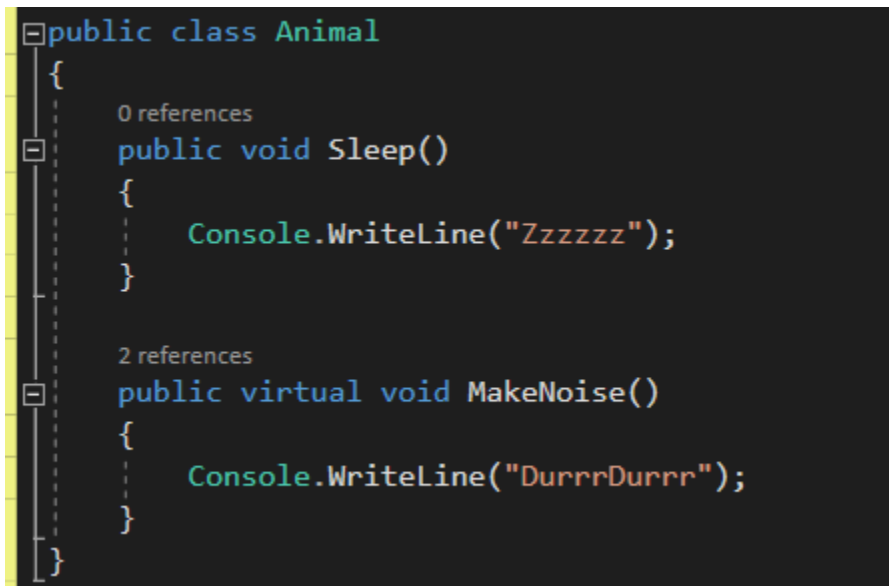
- A class represents an array of related artifacts
- A class can be treated as a kind of abstract data
- Class in OOP abstractly defines the object 's characteristics, we understand it as an object 's blueprint or model, such as a vehicle design, a building drawing, controls such as TextBox, Click, Mark.



```
0 references
class Dog : Animal
{
```

- Class is the keyword for identity declarations
- < Dog> is the user's name, which follows the naming rules
- < Animal > is the name of a class we wish to inherit its properties.

Example



```
public class Animal
{
    0 references
    public void Sleep()
    {
        Console.WriteLine("Zzzzzz");
    }

    2 references
    public virtual void MakeNoise()
    {
        Console.WriteLine("DurrDurr");
    }
}
```

- *Object*

- Object is a Class blueprint product, for example, the car design is put into production, the building drawing used for the construction, TextBox, click ... Dragged into the shape, the piece is called the model part.
- Is a base data form with all forms of data in .NET.

```
class Car
{
    string color = "red";

    0 references
    static void Main(string[] args)
    {
        Car myObj = new Car();
        Console.WriteLine(myObj.color);
    }
}
```

- Use the new keyword to construct an object from a class, then access the methods, properties ... From a Class

Question 2: Describe and explain the idea of constructor in objected oriented programming by an example codes. Save as question_2.sln

- Builder is a special member feature of a class that is executed whenever we construct new class objects.
- The builder has the exact same name as the class, no form of return and must be public

```
public <NameClass>(< private Parameter list of subclasses>) :base(<Parameter list>)
{
}
```

- <nameclass> is the subclass name (derived class)
- <parameter list of subclasses> is the parameter list of the subclass constructor.
- base is the keyword to invoke the constructor of the parent class.
- <Parameter list> is the parameter list corresponding to the constructor of the parent class
- If the child's object is cancelled, the method of cancelation will first be called, then the method of cancelation of the parent class will be cancelled to remove what the child's class can not cancel.

- **Destroyer**

- Destructor is called when deleting object from memory
- Destroyer has no criterion, must be public and must not return

- Compiler will generate a default constructor if not specified

```
using System;

namespace GeeksforGeeks
{
    4 references
    class Complex
    {
        // Class members, private
        // by default
        int real, img;

        // Defining the constructor
        1 reference
        public Complex()
        {
            real = 0;
            img = 0;
        }

        // SetValue method sets
        // value of real and img
        1 reference
        public void SetValue(int r, int i)
        {
            real = r;
            img = i;
        }

        // DisplayValue displays
        // values of real and img
        1 reference
        public void DisplayValue()
        {
            Console.WriteLine("Real = " + real);
            Console.WriteLine("Imaginary = " + img);
        }
    }
}
```

Question 3: FractionsSum Given an integer n , write a method that returns sum of series $1 + (\frac{1}{2})^2 + (\frac{1}{3})^2 + \dots + (\frac{1}{n})^2$. Do not use library function! Expected input and output `FractionsSum(3) → 1.3611111111111` `FractionsSum(5) → 1.4636111111111` Explain your solution. Save as `question_3.sln`

```

C# question_3
1  using System;
2
3  namespace question_3
4  {
5      0 references
6      class Program
7      {
8          0 references
9          class vietnhe
10         {
11             3 references
12             static double Vietnhe(int num)
13             {
14                 double sum = 0.0;
15                 for (int i = 1; i <= num; i++)
16                 {
17                     sum += (1 / (double)(i * i));
18                 }
19                 return sum;
20             }
21
22             0 references
23             static void Main(string[] args)
24             {
25                 Console.WriteLine(Vietnhe(2));
26                 Console.WriteLine(Vietnhe(5));
27                 Console.WriteLine(Vietnhe(3));
28             }
29         }
30     }
31
32
33
34
35
74 % | No issues found

```

```

3 references
static double Vietnhe(int num)
{
    double sum = 0.0;
    for (int i = 1; i <= num; i++)
    {
        sum += (1 / (double)(i * i));
    }

    return sum;
}

```

Because of the process series

· End function of the modal series

Input: 2, 5, 3

Result:


```
1.25
b) 1.4636111111111112
a) 1.3611111111111112
```

Question 4: Sumation Given a phone number, write a function to calculate the sum of the individual digits of that given phone number. Expected input and output `PhoneSum(0939123123) → 0 + 9 + 3 + 9 + 1 + 2 + 3 + 1 + 2 + 3 = 33` Explain your solution. Save as `question_4.sln`

```
1  using System;
2
3  namespace asm1
4  {
5      0 references
6      class Program
7      {
8          1 reference
9          public static int SumCal(int n)
10         {
11             string n1 = Convert.ToString(n);
12             int sum = 0;
13             for (int i = 0; i < n1.Length; i++)
14                 sum += Convert.ToInt32(n1.Substring(i, 1));
15             return sum;
16         }
17
18         0 references
19         public static void Main()
20         {
21             int num;
22             Console.WriteLine("\n\nFunction : To calculate the sum of the individual digits of a number :\n");
23             Console.WriteLine("-----\n");
24             Console.WriteLine("Enter a number: ");
25             num = Convert.ToInt32(Console.ReadLine());
26             Console.WriteLine("The sum of the digits of the number {0} is : {1} \n", num, SumCal(num));
27         }
28     }
29 }
```

✓ No issues found

```
1 reference
public static int SumCal(int n)
{
    string n1 = Convert.ToString(n);
    int sum = 0;
    for (int i = 0; i < n1.Length; i++)
        sum += Convert.ToInt32(n1.Substring(i, 1));
    return sum;
}
```

Declare the variable n and use the function sum to add the natural numbers

Converts an integral list into a series of numbers

Entry planned: 0939201808

Result

```
Function : To calculate the sum of the individual digits of a number :  
-----  
Enter a number: 0939201808  
The sum of the digits of the number 939201808 is : 40
```

Question 5: Conversion Write a C# program taking one input value from the user, choosing the type of conversion and then performing a conversion depending on the type of conversion. For example, if the type of conversion is: *I* then the task is to convert from inches to centimeters. *M* then the task is to convert from mile to kilometer. *P* then the task is to convert from pound to kilogram. Given that: 1 inch is equal to 2.54cm 1 mile is equal to 1.6 km 1 pound is equal to 0.45kg

```
using System;

namespace question_5
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            int a, b, c;
            Console.WriteLine("write the number of inch");
            a = int.Parse(Console.ReadLine());
            Console.WriteLine("write the number of mile");
            b = int.Parse(Console.ReadLine());
            Console.WriteLine("write the number of pound");
            c = int.Parse(Console.ReadLine());

            double cm = a * 2.45;
            double km = b * 1.6;
            double kg = c * 0.45;
            Console.WriteLine($"{a} inch will equal {cm} centimeters");
            Console.WriteLine($"{b} mile will equal {km} kilometers");
            Console.WriteLine($"{c} pound will equal {kg} kilogram");
            Console.ReadKey();
        }
    }
}
```

✓ No issues found

Idea: 1 inch is equal to 2.45cm, 1 mile is 1.6 km, 1 pound is equivalent to 0.45kg, so we want to convert need to multiply by variable. The commands: int, console.writeline, console readkey, ...

InPut: 2, 3, 4

Result:

```
write the number of inch
2
write the number of mile
3
write the number of pound
4
2 inch will equal 4.9 centimeters
3 mile will equal 4.800000000000001 kilometers
4 pound will equal 1.8 kilogram
```

Reference:

- [1] Chen, C., Shi, R., & Xi, H. (2004, June). A typeful approach to object-oriented programming with multiple inheritance. In *International Symposium on Practical Aspects of Declarative Languages* (pp. 23-38). Springer, Berlin, Heidelberg.
- [2] Sarcar, V. (2017). *Interactive C#: Fundamentals, Core Concepts and Patterns*. Apress.