

BTEC FPT INTERNATIONAL COLLEGE



INFORMATION TECHNOLOGY

ASSIGNMENT 1

UNIT: PROGRAMMING

STUDENT :

CLASS :

STUDENT ID :

SUPERVISOR : NGUYEN HOANG ANH VU

Da Nang, July 2022

ASSIGNMENT 1 FRONT SHEET

Qualification	BTEC Level 4 HND Diploma in Business		
Unit number and title	Unit: Programming		
Submission date		Date received (1st submission)	
Re-submission date		Date received (2nd submission)	
Student name		Student ID	
Class		Assessor name	Nguyen Hoang Anh Vu
<p>Student declaration</p> <p>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.</p> <p>Student's signature:</p>			

Grading grid

P1	M1	D1

☐ Summative Feedbacks: ☐ Resubmission Feedbacks:

Grade:

Assessor Signature:

Date:

Internal Verifier's Comments:

Signature & Date:

ACKNOWLEDGMENTS

First of all, I would like to thank my mentor Nguyen Hoang Anh Vu for his constant support in my studies and research, for his patience, motivation, enthusiasm and rich knowledge. His guidance has helped me throughout the time of studying and writing this thesis. Without your wonderful help, I would not have been able to achieve this.

In addition to my mentor, I would like to thank my friends who have helped me improve my knowledge of my subject. Not only that, they are always there to support me when I need it. And besides, I would like to thank the school for creating all conditions for me to have adequate facilities to help me complete my work.

Last but not least, I would like to thank my family: my parents Phan Dinh Quy and Lam Thi Tam, who gave birth to me from the beginning and supported me spiritually. They are always behind to care and help me have more motivation to complete the work well.

ASSURANCE

I declare that this is my work, based on my research, and that I have recognized all materials and sources utilized in its production, including books, papers, reports, lecture notes, and any other type of document, electronic or personal communication. I further declare that I have not previously submitted this assignment for assessment in any other unit, except where explicit permission has been granted by all unit coordinators involved, or at any other time in this unit, and that I have not duplicated or stolen ideas from the work of others in any way.

Declaration of the learner	
I verify that the work I've submitted for this assignment is all my own, and that all research sources have been properly credited.	
Signature of the student:	Date:

TABLE OF CONTENT

INSTRUCTOR/ SUPERVISOR/ ASSESSOR.....	i
REVIEWERS.....	ii
ACKNOWLEDGMENTS.....	iv
ASSURANCE.....	v
TABLE OF CONTENT.....	vi
LIST OF TABLES AND FIGURES.....	vii
LIST OF THE ACRONYM.....	ix
INTRODUCTION.....	1
CHAPTER 1: ALGORITHMS, ISSUES RELATED TO ALGORITHMS AND BASIC PROGRAMMING	2
1. Define basic algorithms to carry out an operation and outline the process of programming an Application (LO1).....	2
1.1 Provide a definition of what an algorithm is and outline the process in building an Application (P1).....	2
1.1.1 Define Algorithms.....	2
1.1.2 The process in building an application.....	11
1.2 Determine the steps taken from writing code to execution (M1).....	16
1.3 Examine the implementation of an algorithm in a suitable language. Evaluate the relationship between the written algorithm and the code variant (D1).....	18
CRITICAL EVALUATION.....	18
CONCLUSION.....	19
REFERENCES.....	20

LIST OF TABLES AND FIGURES

Figure 1-1: Algorithms.....	2
Figure 1-2: rinsing the rice.....	2
Figure 1-3: Using the proper water ratio.....	2
Figure 1-4: Bringing the water to a boil.....	3
Figure 1-5: Maintaining a low heat.....	3
Figure 1-6: Cooking should be done without looking or stirring	3
Figure 1-7: Cover the rice and set it aside to rest.....	4
Figure 1-8: Toss the rice with a fork to fluff it up.....	4
Figure 1-9: Apply warm water to your face.....	5
Figure 1-10: Use your preferred cleaner.....	5
Figure 1-11: Exfoliate your skin gently.....	5
Figure 1-12: Rinse well and pat dry.....	6
Figure 1-13: Use toner.....	6
Figure 1-14: Apply a moisturizer.....	6
Figure 1-15: Flowchart symbols.....	7
Figure 1-16: Determine which number is the biggest.....	8
Figure 1-17: Sum of a and b.....	8
Figure 1-18: Solve first-degree equations.....	9
Figure 1-19: Source code solve first-degree equations.....	10
Figure 1-20: Output solve first-degree equations.....	10
Figure 1-21: The Software development cycle.....	11
Figure 1-22: Analyzing the Problem.....	11
Figure 1-23: The Algorithms.....	11
Figure 1-24: Execution of the test.....	12
Figure 1-25: Final Documentation.....	12
Figure 1-26: The software development cycle.....	12
Figure 1-27: The software development cycle.....	13
Figure 1-28: Design and Prototyping.....	13
Figure 1-29: Source code.....	14
Figure 1-30: Try three random numbers.....	15

LIST OF THE ACRONYM

CLR	Common Language Runtime
CSC	Common Service Center
GUI	Graphical User interface
IBM	International Business Machines
JSC	Joint Stock Company
I/O	Input/Output
PE	Portable Executable
RAM	Random Access Memory

INTRODUCTION

Programming plays an extremely important role in life. Coming to this report, we will learn about algorithms, problems related to algorithms and basic programming. It will help us understand more deeply what is an algorithm, how is an algorithm defined? What is the process of building an algorithm, and how does the algorithm work? Let's find out in this assignment!

❖ This report includes the following:

Chapter 1: Algorithms, issues related to algorithms, and basic programming.

1. Define basic algorithms to carry out an operation and outline the process of programming an application.

1.1 Provide a definition of what an algorithm is and outline the process in building an Application.

1.2 Determine the steps taken from writing code to execution.

1.3 Examine the implementation of an algorithm in a suitable language. Evaluate the relationship between the written algorithm and the code variant.

CHAPTER 1: ALGORITHMS, ISSUES RELATED TO ALGORITHMS AND BASIC PROGRAMMING

1. Define basic algorithms to carry out an operation and outline the process of programming an application. (LO1)

1.1 Provide a definition of what an algorithm is and outline the process in building an application. (P1)

1.1.1 Define Algorithms

- An **algorithm** is a method or technique that is used to solve a problem. It works by having your computer do a series of predefined actions that define how to accomplish something, and your computer will execute it exactly the same way every time.

❏ **Examples of algorithms in real life:**

- ❖ We generally use one of two methods to represent the algorithm:
- ❖ I have an example of a simple natural language face washing algorithm:
- ❖ **Flowchart symbols:**
 - is a diagrammatic representation of an algorithm
 - It uses a variety of symbols to describe the steps necessary to solve an issue.

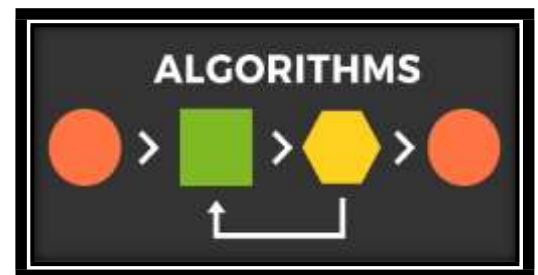


Figure 1-1: Algorithms






Symbol	Name	Function
	Start/end	An oval represents a start or end point
	Arrows	A line is a connector that shows relationships between the representative shapes
	Input/Output	A parallelogram represents input or output
	Process	A rectangle represents a process
	Decision	A diamond indicates a decision

Figure 1-15: Flowchart symbols

❖ **More information:**

- The shape of **the oval** indicates whether the algorithm is at its beginning or end. Its material is generally divided into two sections: "beginning" and "ending."

❖ The following are some examples of flow charts in schematic form:

- The program begins initially, then reads two numbers, a and b, as shown in the flow chart.

- The program proceeds by adding two numbers, a and b.

- And lastly, the program writes the result on the screen and exits.

➤ This is one of the program's most basic algorithms.

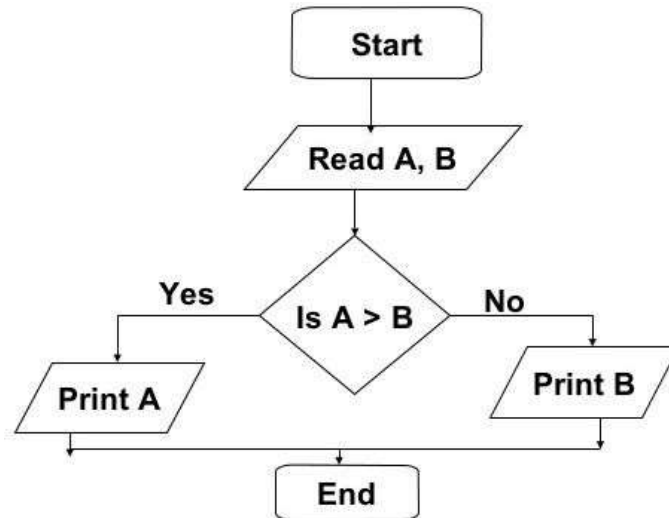


Figure 1-16: Determine which number is the biggest

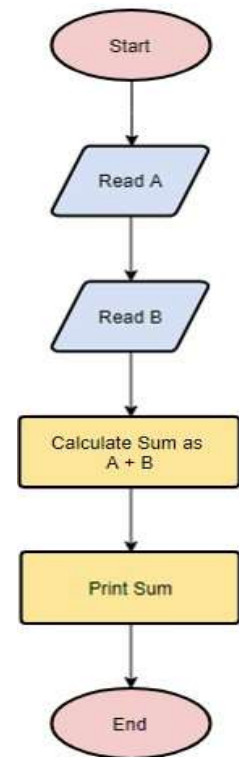


Figure 1-17: Sum of a and b

✚ The next program is a comparison to determine which number is the biggest:

- To begin, the software reads the integers a and b.

- To determine which number is higher, we utilize the if algorithm:

+ If a is greater than b, the program will return true. and print the number a is the largest number.

+ Conversely, if a is not greater than b, the program will return false and print the number b is the largest number.

- Finally, the performance comes to a close.

❖ **Create a good algorithm by following these steps:**

- The issue can have a variety of solutions, but it will still produce outcomes. A good algorithm, on the other hand, will make our software run quicker, the code will be shorter, and it will help our program run with as few or no mistakes as possible, and the computer will do it faster than if the algorithm is done poorly.

- Your software does not necessitate the usage of a high-end machine.

❖ **To design an algorithm, we must first implement it and consider the following issues:**

- **Finality:** The algorithm requires an endpoint, and the loops that are produced require a breakpoint to avoid infinite loops. The algorithm's number of steps is limited.
- **Efficiency and dependability:** The problem's output must not be inconsistent or contradictory to the problem's requirements.
- **Exactly:** The algorithm's findings satisfy the problem's requirements.
- **Efficiency and resiliency:** When it comes to comparable issues, algorithms must be adaptable.
- **Nimble:** The best algorithm will meet the processing time and deliver quick results.

❖ **Examples of how to use block diagrams, create excellent algorithms, and apply algorithm characteristics include:**

✚ **Problem solving first-degree equations:**

- **Input:** Enter number a and b.
- **Output:** Display the solution of the equation.

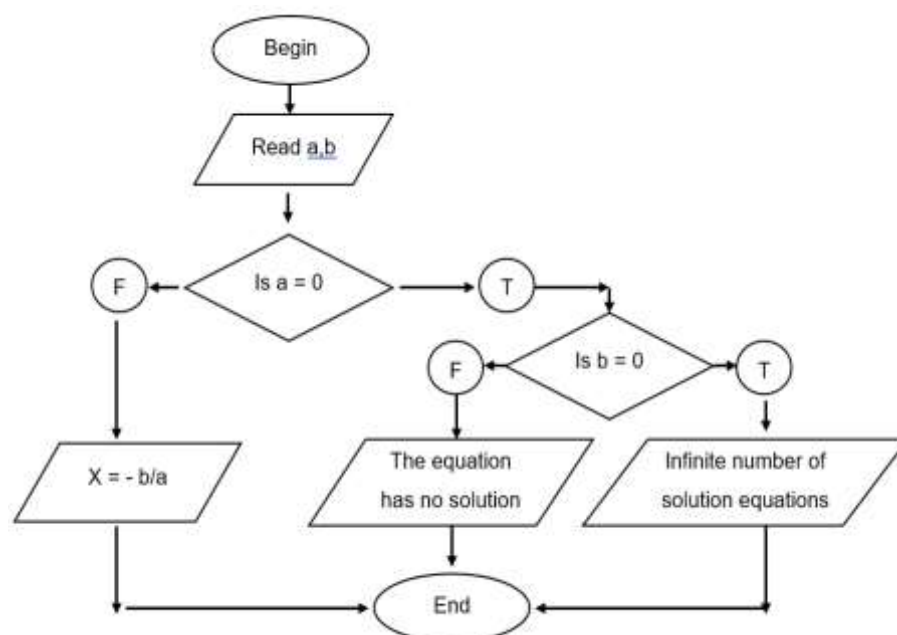


Figure 1-18: Solve first-degree equations

❖ Source code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace FirstDegreeEquations
{
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            float x;
            Console.WriteLine("Input your number a: ");
            int a = int.Parse(Console.ReadLine());
            Console.WriteLine("Input your number b: ");
            int b = int.Parse(Console.ReadLine());
            if (a == 0)
            {
                if (b == 0)
                {
                    Console.WriteLine("Infinite number of solution equations");
                }
                else
                {
                    Console.WriteLine("The equations has no solution");
                }
            }
            else
            {
                x = - b / a;
                Console.WriteLine($"x: {x}");
            }
            Console.ReadKey();
        }
    }
}
```

Figure 1-19: Source code solve first-degree equations

```
Input your number a: 3
Input your number b: 6
x: -2

```

Figure 1-20: Output solve first-degree equations

- ❖ **Note:** When multiple algorithms are used to solve a problem, the solution is the same. We should select the most efficient algorithm. It will aid in the faster and better performance of our software.

1.1.2 The process in building an application.

❖ A set of basic algorithm steps for carrying out a task:

- Defining or analyzing the problem
- Design (Algorithm)
- Coding
- Documenting the program
- Compiling and running the program
- Debugging and testing
- Maintenance



Figure 1-21: The software development cycle

❖ Defining and Analyzing the Problem:

- We investigate the issue in this stage. we determine the most effective method for resolving these issues. Studying an issue is also important since it aids a programmer in making decisions concerning the following:



Figure 1-22: Analyzing the Problem

- The data and statistics required for the program's development.
- The way the program will be put together.
- In addition, the language in which the program will work best.
- What is the desired output and in what format is it required, ...etc.

❖ Example of the software development cycle: Solve quadratic equation

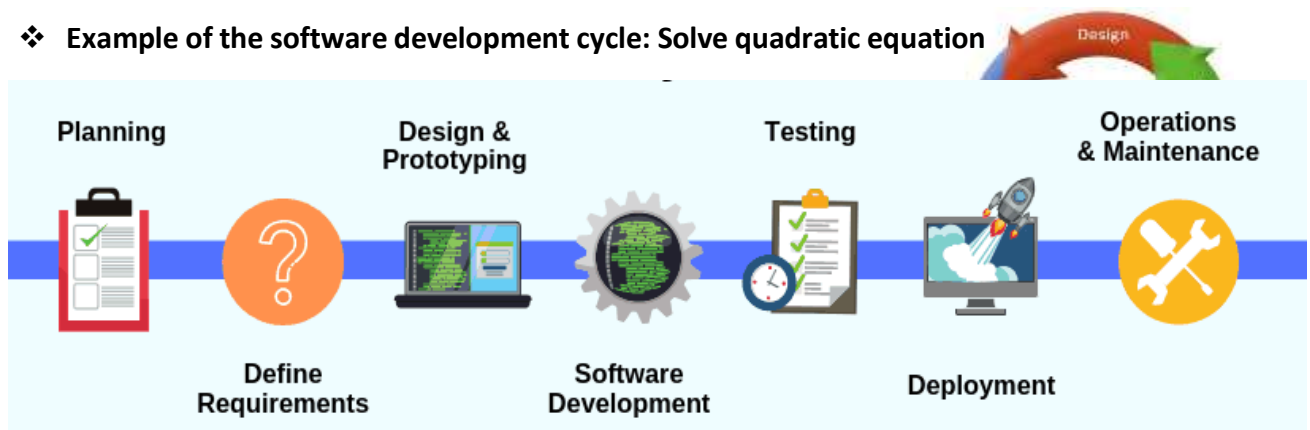


Figure 1-26: The software development cycle

- **Step 1: Planning.** We will use IDE Microsoft visual studio with C# programming language.



Figure 1-27: The software development cycle

- **Step 2: Define Requirements.**
 - **Input:** Three numbers: a, b, c of the quadratic equation
 - **Output:** The solutions of the equation
- **Step 3: Design and Prototyping.**

✚ **Flow chart:**

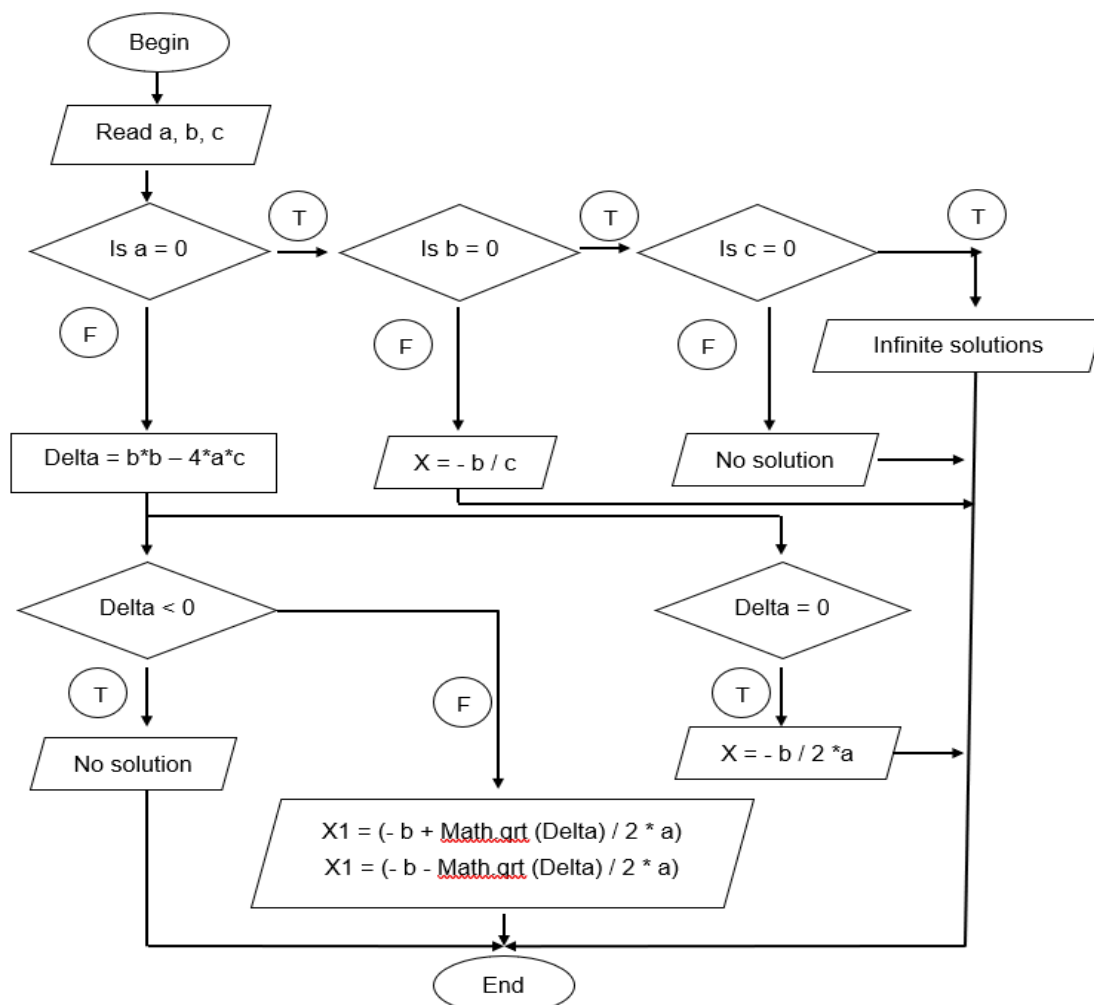


Figure 1-28: Design and Prototyping

▪ Step 4: Software Development

Source code:

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using System.Threading.Tasks;
6
7  namespace solveQuadraticEquations
8  {
9      0 references
10     class Program
11     {
12         0 references
13         static void Main(string[] args)
14         {
15             double x1, x2, delta;
16             Console.Write("Input your number a: ");
17             double a = double.Parse(Console.ReadLine());
18             Console.Write("Input your number b: ");
19             double b = double.Parse(Console.ReadLine());
20             Console.Write("Input your number c: ");
21             double c = double.Parse(Console.ReadLine());
22             if (a == 0)
23             {
24                 if (b == 0)
25                 {
26                     if (c == 0)
27                     {
28                         Console.WriteLine("The equation has infinitely many solutions");
29                     }
30                     else
31                     {
32                         Console.WriteLine("The equation has no solution");
33                     }
34                 }
35                 else
36                 {
37                     Console.WriteLine("The solution of the equation is: " + (-b / c));
38                 }
39             }
40             else
41             {
42                 delta = ((double)Math.Pow(b, 2) - 4 * a * c);
43                 if (delta < 0)
44                 {
45                     Console.WriteLine("The equation has no solution");
46                 }
47                 else if (delta == 0)
48                 {
49                     Console.WriteLine("The solution of the equation is: " + (-b / 2 * a));
50                 }
51                 else
52                 {
53                     x1 = (-b + Math.Sqrt(delta) / 2 * a);
54                     x2 = (-b - Math.Sqrt(delta) / 2 * a);
55                     Console.WriteLine("The solution of the equation is: \n x1 = {0} \n x2 = {1}", x1, x2);
56                 }
57             }
58             Console.ReadKey();
59         }
60     }
```

Figure 1-29: Source code

- **Step 5: Testing**

- Firstly, I **try three random numbers** to see if the equation works like it should?

```
Input your number a: 1
Input your number b: 2
Input your number c: -3
The solution of the equation is:
x1 = 0
x2 = -4
—
```

Figure 1-30: Try three random numbers

- Next, I let **a = 0**, **b** and **c** be **2 random numbers**.

```
Input your number a: 0
Input your number b: 3
Input your number c: 5
The solution of the equation is: -0,6
```

Figure 1-31: a = 0, b and c be 2 random numbers

- Next, I let **a = 0**, **b = 0** and **c** be a random number.

```
Input your number a: 0
Input your number b: 0
Input your number c: 3
The equation has no solution
—
```

Figure 1-32: a = 0, b = 0, c be a random number.

- **a = 0, b = 0 and c = 0.**

```
Input your number a: 0
Input your number b: 0
Input your number c: 0
The equation has infinitely many solutions
```

Figure 1-33: a = 0, b = 0 and c = 0.

- **Step 6: Deployment**

- In software and online development, deployment refers to the process of moving changes or updates from one deployment environment to another.



Figure 1-34: Deployment

- When building a website, you'll always have your live website, often known as the production environment or live environment.

▪ **Step 7: Operation and Maintenance**

- Web Application Development is vital, but so is system maintenance.
- It keeps solutions agile in order to adapt with rapidly changing technology and business conditions.
- Once the operation is completed, it is followed by a lengthy management phase.
- Software maintenance include enhancing software performance by improved development, lowering mistakes, and removing unnecessary development.



Figure 1-35: Operation and Maintenance

1.2 Determine the steps taken from writing code to execution. (M1)

✚ The steps for compiling a C# program are as follows:

- Creating a Managed Module from Source Code.
 - Adding a new managed module to the assembly / assemblies.
 - The CLR is being loaded.
 - CLR is in charge of putting the assembly together.
- Every .NET program is initially compiled using the proper compiler, for example, if we develop a program in C#, it is compiled using the C# compiler.

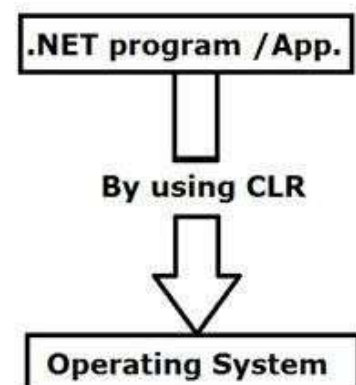


Figure 1-36: Compile a C# program

- CLR is used by every program in the .NET framework to execute (communicate) in an operating system (Common Language Runtime).

❖ Compiling Source Code into Managed Module:

- The .NET framework has its own compiler for each language. JavaScript has a Jscript compiler, for example, and C# has a C# compiler.

- A Managed Module is created whenever source code is generated by the proper compiler.

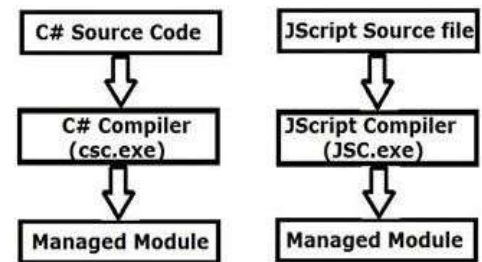


Figure 1-37: Module that is being managed

- If we write a console program and build it, we will get an .exe file. When we construct a web application, .dll files are generated.

- The Managed Module is a Windows Portable Executable (PE) file that contains the following components:

- PE Header
- CLR Header
- Metadata

❖ Combining newly created managed module into the assembly / assemblies:

- The terms "assembly" and ".dll" or ".exe" refer to the same thing. There are several software files in a web application that are developed in various languages. This assembly integrates and assembles into a single unit. It also includes pictures (sometimes known as resource files) and a number of controlled modules.

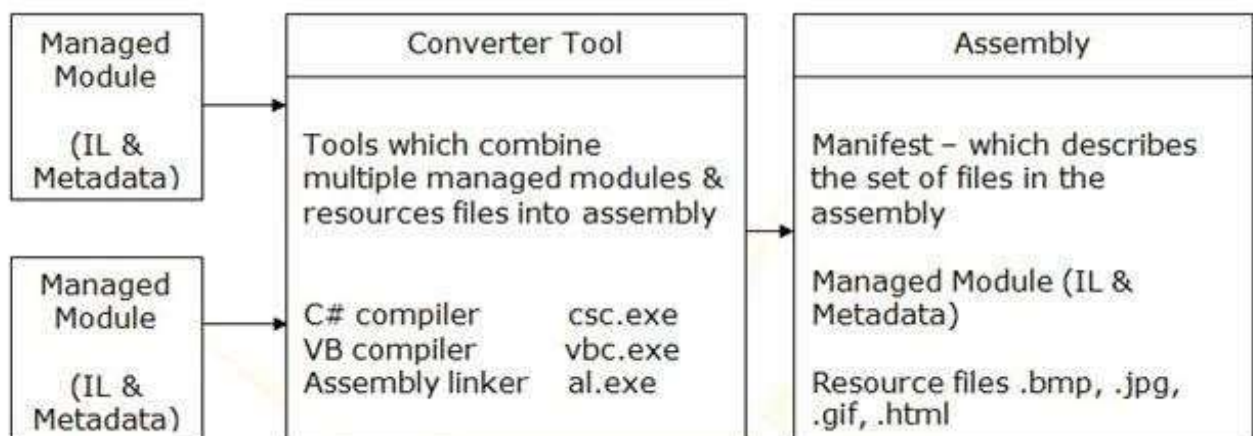


Figure 1-38: Combining newly created managed module

❖ Loading the CLR:

- Each software is stored on a hard drive. Each file should be loaded into RAM from a hard disk, converting magnetic address space to electric address space. This is achievable with the Loader's assistance.

❖ Executing the assembly in & by CLR:

- The acronym CLR stands for Common Language Runtime. It's the brains behind the .NET framework, and it's what allows a program to run on a computer.
- Garbage collection, code verification, memory management, code access security, and IL to Native translation are some of the features provided by CLR.
- PE Header, CLR Header, Metadata, and Address Table values are all visible.

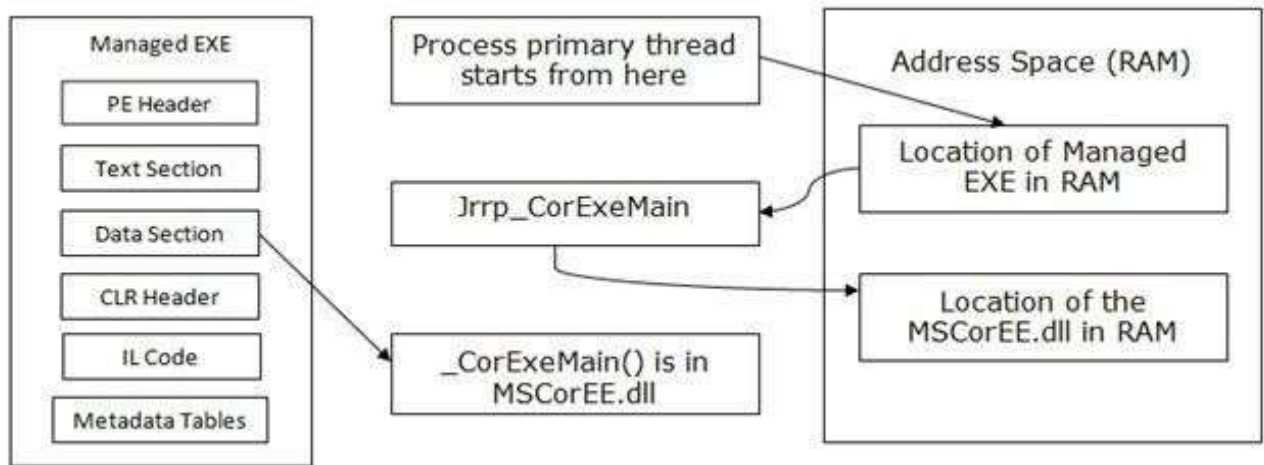


Figure 1-39: Executing the assembly in & by CLR

1.3 Examine the implementation of an algorithm in a suitable language.
Evaluate the relationship between the written algorithm and the code variant (D1)

CRITICAL EVALUATION

CONCLUSION

After completing this report, I understood and grasped how to build a basic algorithm, the process to execute a program, the life cycle of an algorithm, how to draw a flow chart to explain How does the algorithm work?

Programming is really amazing! thanks to it I understand how the devices around us work. This is very interesting and fantastic. Thankyou my Mentor!

REFERENCES

1. hnglobal.ighernationals.com (2021). Cloud Computing, Computing & Digital Technologies Resource Library [online]. Available: <https://hnglobal.ighernationals.com/subjects/computing/resource-library>. [Accessed 18 June 2021].