Tribhuvan University

Bhaktapur Multiple Campus

Doodhpati-17, Bhaktapur, Nepal

Lab report of

Web-Technology

(CSC 367)

Submitted by:

Saini Thapa Magar

B.Sc. CSIT 6th semester

Roll no.: 44

Submitted to:

Ramesh Kharbuja

C# Program with Fundamental concepts

Introduction to .NET and C#

.NET is an opensource developer platform, created by Microsoft, for building many different types of applications. With .NET, we can use multiple languages, editors, and libraries to build for web, mobile, desktop, games, IoT, and more. We can write .NET apps in C#, F#, or Visual Basics. We are using C# to build our projects with .NET Framework.

C# is a general- purpose, object- oriented programming language designed for Common Language Infrastructure (CLI), which consists of the executable code and the runtime environment that allows the use of various high- level languages on different computer platforms and architectures.

Data Types:

Data type is used to represent the type of a variable. The Common Language Runtime (CLR) provides two categories of data types. They are Value Type and Reference Type. The value type stores its value in the stack and reference type stores its value in the managed heap. The value type consists of the following data types:

1) Value Type:
    - Simple Type
    - Enumeration Type
    - Structure Type
2) Reference Type:
    - Class Type
    - String Type
    - Delegate Type
    - Interface Type
    - Array Type

Simple value type includes the following:

a) Integer type:
    i) Int – System.Int32
    ii) Short – System.Int16
    iii) Byte – System.Byte
    iv) Long – System.Int64
b) Floating type:
    i) Float – System.Single
    ii) Double - System.Double
c) Decimal type - System.Decimal
d) Boolean type - System.Boolean
e) Character type - System.Char

Operators in C#

An operator is a symbol that tells the compiler to perform specific mathematical or logical manipulations. C# has rich set of built-in operators and provides the following type of operators:

- Arithmetic Operators
- Assignment Operators
- Relational Operators
- Logical Operators
- Bitwise Operators

1) **Arithmetic Operators**
   The arithmetic operators perform arithmetic operations on the given operands. The arithmetic
   operator includes:

Operator	**Description**	**Example**
+	Addition Operator	A + B
-	Subtraction Operator	A – B
*	Multiplication Operator	A * B
/	Division Operator	A / B
%	Modulo Operator (Finds remainder)	A % B
++	Increment Operator	A++ or B++
--	Decrement Operator	A—or B--

2) **Assignment operators**
   In C#, an assignment operator is used to assign a value to a variable. It is denoted by = and assigns the value of the right hand operand to a variable, a property or an indexer element given by its left hand operand. Assignment operators in C# are:
   Simple Assignment (=), Addition Assignment (+=), Subtraction Assignment (-=), Multiplication Assignment (*=), Division Assignment (/=), Modulus Assignment (%=), etc.

3) **Comparison Operators**
   Comparison operators are used to compare two values or variables. To make decisions, comparison operators are used in specific scenarios. The return value of comparison operator
   is either **true** or **false**. This data type is known as Boolean data type.

Operator	**Name**	**Example**
==	Equal to	A == B
!=	Not equal	A != B
>	Greater than	A > B

<	Less than	A < B
>=	Greater than or equal to	A >= B
<=	Less than or equal to	A <= B

4) **Logical Operators**
   Logical operators are used to determine the logic between variables or values. Like comparison
   operator, it is also of Boolean type.

Operator	**Name**	**Example**
&&	Logical AND	A < 5 && B <10
||	Logical OR	A < 5 || A < 4
!	Logical NOT	!(A<5)

5) **Bitwise operators**
   C# provides four bitwise and two bit shift operators. Bitwise and bit shift operators are used to
   perform bit level operations on integer and Boolean data.

Operator	**Name**	**Example**
~	Bitwise Complement	~A
&	Bitwise AND	A & B
|	Bitwise OR	A | B
^	Bitwise Exclusive OR (XOR)	A ^ B
<<	Bitwise Left Shift	A << 1
>>	Bitwise Right Shift	A >> 1

Flow control of Programming

Flow control refers to the order in which statements are executed in a program. It allows a program to make decisions, repeat operations, and respond to various inputs or conditions.

There are three main types of flow control in programming: selection statements, iteration statements, and jump statements.

1) Selection statements:
   These statements allow the program to make decisions based on a condition. There are two types of selection statements in C#: if-else statements and switch statements.
   a) **if-else** statements allow the program to execute different paths of code based on whether a certain condition is true or false.
      Syntax:
      if(condition)
      {statements;}

else
{statements;}

b) **switch** statements allow the program to execute different paths of code based on the value of a variable or expression.

Syntax:

```
switch(argument)
{
    Case 1:
            //Do something
            break;
    Case 2:
            //Do something
            break;
    Default:
            //Do something
            break;
}
```

2) Iteration Statements: These statements allow the program to execute a block of code repeatedly based on a certain condition. There are three types of iteration statements in C#: for loops, while loops, and do-while loops.

a) **for loops** allow the program to execute a block of code a fixed number of times.

Syntax:

```
for(initializer; condition; iterator)
{ statements;}
```

b) **while loops** allow the program to execute a block of code while a certain condition is true.

Syntax:

```
while(condition)
{statements;}
```

c) **do-while loops** are similar to while loops, but the block of code is executed at least once, even if the condition is false.

Syntax:

```
do{
statements; //body
}while(condition)
```

d) **foreach loop** construct is a type of iteration statement in C# that allows you to iterate over a collection of items such as an array, a list, or any other object that implements the IEnumerable interface. It simplifies the process of iterating through the elements of a collection by automatically initializing a loop variable, iterating through each element of the collection, and then exiting the loop when all elements have been processed.

Syntax:

```
foreach (var item in collection)
{
   // Code to be executed for each item in the collection
}
```

3) **Jumping Statements:** These statements allow the program to transfer control to another part of the program. There are three types of jump statements in C#: break, continue, and goto.
   a) **break** statements allow the program to exit a loop early.
   b) **continue** statements allow the program to skip over the current iteration of a loop and go to the next iteration.
   c) **goto** statements allow the program to transfer control to a labeled statement in the code.

Structs and Enum

Struct : In C#, a struct is a value type that is similar to a class, but with some key differences. It is used to encapsulate small groups of related variables, which can then be passed around as a single unit. Structs are typically used for lightweight objects that do not require inheritance or other advanced features.

Syntax:

```
struct StructName
{
   // Fields and methods
}
```

Enums: An enum is a special type in C# that allows you to define a set of named constants. Each constant has an underlying integer value that can be accessed using the enum type. Enums are often used to represent sets of related values, such as days of the week or colors.

Syntax:

```
enum EnumName
{
   Constant1,
   Constant2,
   Constant3,
   // ...
}
```

Program in C#

Saini dotNET — Saini_dotl

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

namespace Saini_dotNET
{
    0 references
    internal class Program
    {
        private string model = "MUSTANG";
        1 reference
        static void Compare(string a, string b)
        {
            int x = Convert.ToInt32((string)a);
            int y = Convert.ToInt32((string)b);
            if (x > y)
                Console.WriteLine(a + " is the greatest number");
            else
                Console.WriteLine(b + " is the greatest number");
        }
        1 reference
        static void ForLooping()
        {
            for (int i = 0; i < 5; i++)
                Console.WriteLine(i + 1);
        }
        1 reference
        static void WhileLooping()
        {
            int i = 0;
            while (i != 5)
            {
                Console.WriteLine(i + 1);
                i++;
            }
        }

        1 reference
        static void SwitchCase()
        {
            string a;
            Console.WriteLine("Enter the number to denote day (1-7): ");
            a = Console.ReadLine();
            int b = Convert.ToInt32(a);
            switch (b)
            {
                case 1:
                    Console.WriteLine("Sunday");
                    break;
                case 2:
                    Console.WriteLine("Monday");
                    break;
                case 3:
                    Console.WriteLine("Tuesday");
                    break;
                case 4:
                    Console.WriteLine("Wednesday");
                    break;
                case 5:
                    Console.WriteLine("Thursday");
                    break;
                case 6:
                    Console.WriteLine("Friday");
                    break;
                case 7:
                    Console.WriteLine("Saturday");
                    break;
                default:
                    Console.WriteLine("TRY AGAIN");
                    break;
            }
        }
        1 reference
```

```
70          static void BitwiseOperator()
71          {
72              int i, j, res1, res2, res3, res4;
73              string a, b;
74              Console.WriteLine("Enter the value of a and b for bitwise operation: ");
75              a = Console.ReadLine();
76              b = Console.ReadLine();
77              i = Convert.ToInt32(a);
78              j = Convert.ToInt32(b);
79              res1 = i & j;
80              Console.WriteLine("{0} AND {1} = {2}", i, j, res1);
81              res2 = i | j;
82              Console.WriteLine("{0} OR {1} = {2}", i, j, res2);
83              res3 = i ^ j;
84              Console.WriteLine("{0} XOR {1} = {2}", i, j, res3);
85              res4 = ~i;
86              Console.WriteLine("~{0} = {1}", i, res4);
87          }
                1 reference
88          static void ShiftOperation()
89          {
90              int i, j;
91              string a, b;
92              Console.WriteLine("Enter the value of a for Arithmetic left Shift operation: ");
93              a = Console.ReadLine();
94              i = Convert.ToInt32(a);
95              Console.WriteLine("{0} << 1 = {1}", i, i << 1);
96              Console.WriteLine("Enter the value of b for Arithmetic right Shift operation: ");
97              b = Console.ReadLine();
98              j = Convert.ToInt32(b);
99              Console.WriteLine("{0} >> 1 = {1}", j, j >> 1);
100         }
                1 reference
101         static void Cont_Break()
102         {
103             Console.WriteLine("Example of Continue: ");
104             for (int i = 0; i < 10; i++)
105             { if (i % 2 == 0)
106                     continue;
107                 Console.WriteLine(i);
108             }
109             Console.WriteLine("Example of Break: ");
110             for (int i = 0; i < 10; i++)
111             {
112                 if (i > 5)
113                     break;
114                 Console.WriteLine(i);
115             }
116             Console.WriteLine("Example of GOTO: ");
117             int j = 0;
118         start:
119             if (j < 5)
120             {
121                 Console.WriteLine("j = " + j);
122                 j++;
123             goto start;
124             }
125         }
                1 reference
126         static void BooleanDataType()
127         {
128             bool YES = true;
129             Console.WriteLine("Enter true or false: ");
130             string answer = Console.ReadLine();
131             bool ans2 = Convert.ToBoolean(answer);
132             if (ans2 == YES)
133                 Console.WriteLine("True value");
134             else
135                 Console.WriteLine("False Value");
136         }
                1 reference
137         static void For_Each_Method()
138         {
139             var numbers = new List<int>() { 23, 24, 45, 76, 87 };
140             int sum = 0;
141             foreach (int num in numbers)
142             {
```

```
143                 Console.WriteLine(num);
144                 sum += num;
145             }
146             Console.WriteLine("Sum is " + sum + "\n");
147
148             string[] cars = { "BMW", "Ford", "Lamborghini", "Tesla" };
149             foreach (string car in cars)
150             {
151                 Console.WriteLine(car);
152             }
153         }
                3 references
154         enum Days
155         {
156             Monday, Tuesday, Wednesday, Friday, Saturday, Sunday
157         }
                1 reference
158         static void Enumeration()
159         {
160
161             Console.WriteLine((int)Days.Monday);
162             Console.WriteLine((int)Days.Wednesday);
163             Console.WriteLine((int)Days.Saturday);
164         }
                2 references
165         struct Drama
166         {
167             public int year;
168             public string name;
169         }
                1 reference
170         static void structure()
171         {
172             Drama d=new Drama();
173             d.year = 2016;
174             d.name = "Goblin";
175             Console.WriteLine("Year = "+d.year+" Name = "+d.name);
176         }
177
                1 reference
178         static void TernaryOp()
179         {
180             int a, b;
181             string c, d;
182             c = Console.ReadLine();
183             d = Console.ReadLine();
184             a = Convert.ToInt32(c);
185             b = Convert.ToInt32(d);
186             var result = a > b ? a : b;
187             Console.WriteLine(result + " is the greatest number");
188         }
189
            0 references
190     static void Main(string[] args)
191         {
192             Console.WriteLine("Hello Saini");
193             string a, b;
194             Console.WriteLine("Input the value of a and b : ");
195             a = Console.ReadLine();
196             b = Console.ReadLine();
197             Compare(a, b);

198             ForLooping();
199             WhileLooping();
200             SwitchCase();
201             BitwiseOperator();
202             ShiftOperation();
203             Cont_Break();
204             BooleanDataType();
205             For_Each_Method();
206             structure();
207             Enumeration();
208             TernaryOp();
209             Console.ReadKey();
210         }
211     }
212 }
```

Output:

compare(a, b) : use of if/else and > operator

```
Hello Saini
Input the value of a and b :
23
32
32 is the greatest number
```

ForLooping(): use of For loop to print numbers from 1 to 5

```
1
2
3
4
5
```

WhileLooping(): use of While loop to print numbers from 1 to 5

```
1
2
3
4
5
```

SwitchCase() : use of Switch, case and break

```
Enter the number to denote day (1-7):
6
Friday
```

BitwiseOperation(): use of bitwise operator (bitwise OR, AND, XOR and NOT)

```
Enter the value of a and b for bitwise operation:
45
67
45 AND 67 = 1
45 OR 67 = 111
45 XOR 67 = 110
~45 = -46
```

ShiftOperation() : use of left and right shift operators

```
Enter the value of a for Arithmetic left Shift operation:
45
45 << 1 = 90
Enter the value of b for Arithmetic right Shift operation:
45
45 >> 1 = 22
```

Cont_Break(): use of jumping statements - continue, break and goto.

```
Example of Continue:
1
3
5
7
9
Example of Break:
0
1
2
3
4
5
Example of GOTO:
j = 0
j = 1
j = 2
j = 3
j = 4
```

BooleanDataType() : use of Boolean data type

```
Enter true or false:
True
True value
```

For_each_Method() : use of For Each Loop and +, = operators

```
23
24
45
76
87
Sum is 255

BMW
Ford
Lamborghini
Tesla
```

structure(): use of structure instance

```
Year = 2016 Name = Goblin
```

Enumeration(): use of enum types

```
0
2
4
```

TernaryOp(): use of Conditional operator(Ternary operator)

```
102
213
213 is the greatest number
```