**EXERCISE 1**

**BASIC PROGRAMS IN C#**

**Objective:**

To develop C# programs to using control statements, arrays and methods

**Questions(Complete All the Questions):**

1. Create a program to check whether a given year is a leap year or not by using methods
2. Create a program to generate all prime numbers between the two given numbers using method.
3. To write a program in C# for an interactive calculator using switch case, method and menu options.
4. Create a program in C# for a simple calculator with params for multiple inputs using methods
5. Develop a program to perform the following banking operations using method with **ref** and **out** parameters.
6. Create a program in C# to get a set of regno using arrays, (ii) display the array and (iii) search the array if a regno is there not.

**Description:**

**Control Statements:**

**if** and **else** Statement

* **if** statement to specify a block of C# code to be executed if a condition is **true**.
* **else** statement to specify a block of code to be executed if the condition is **false**.

**Example:** C# Program to check the given number is odd or even.

using System;

class Program {

static void Main(string[] args) {

Console.WriteLine("Enter the number: ");

int num = Convert.ToInt32(Console.ReadLine());

if(num%2 == 0) {

Console.WriteLine("Even");

}

else{

Console.WriteLine("Odd");

}

Console.ReadKey();

}

}

**Switch Statements**

* The **switch** statement to select one of many code blocks to be executed.
* **Example:** Simple Calculator Program

using System;

class Program {

static void Main(string[] args) {

Console.Write("Enter the Number1: ");

int num1 = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the Number2: ");

int num2 = Convert.ToInt32(Console.ReadLine());

Console.Write("Enter the Operator(+,-,\*,/): ");

int op = Console.Read();

switch (op)

{

case '+':

Console.WriteLine("Sum: " + (num1+num2));

break;

case '-':

Console.WriteLine("Difference: " + (num1 - num2));

break;

case '\*':

Console.WriteLine("Product: " + (num1 \* num2));

break;

case '/':

Console.WriteLine("Division: " + (num1 / num2));

break;

default:

Console.WriteLine("Invalid Operator");

break;

}

Console.ReadKey();

}

}

**for Loop**

* + **for loop** is used to repeat the execution of a logic or set of statements.
  + When you know exactly how many times you want to loop through a block of code, use the **for loop.**

**Example: To find factorial of a given number**

using System;

class Program {

static void Main(string[] args) {

Console.Write("Enter the Number: ");

int num = Convert.ToInt32(Console.ReadLine());

int fact = 1;

for (int i = 1; i <= num; i++)

{

fact \*= i;

}

Console.WriteLine("Factorial: "+ fact);

Console.ReadKey();

}

}

**while Loop**

* The **while** loop loops through a block of code as long as a specified condition is true.
* **Example:** Count Number of Digits in an Integer using while loop

using System;

class Program {

static void Main(string[] args) {

Console.Write("Enter the Number: ");

int num = Convert.ToInt32(Console.ReadLine());

int count = 0;

while (num != 0)

{

num /= 10;

++count;

}

Console.WriteLine("Number of digits: " + count);

Console.ReadKey();

}

}

**Create an Array**

To declare an array, define the variable type with square brackets:

Example: To store set of numbers

int[] nums = { 1, 2, 3, 4, 5, 6 };

**Create an Empty Array:-**

You can create an empty array using new operator and the values can be stored dynamically.

Example: To create an empty to store 5 numbers as given below

int[] nums = new int[5];

Here, the new operator creates an array object and allocate memory for five integer numbers.

**To Access all the elements of an array**

We can access all the elements of an array using for loop

using System;

class Program {

static void Main(string[] args) {

int[] nums = { 11, 21, 13, 40, 15, 60 };

for(int i = 0; i < nums.Length; i++)

{

Console.WriteLine(nums[i]);

}

Console.ReadKey();

}

}

**The foreach Loop in arrays**

There is also a foreach loop, which is used exclusively to loop through elements in an **array**:

using System;

class Program {

static void Main(string[] args) {

int[] nums = { 11, 21, 13, 40, 15, 60 };

foreach(int i in nums){

Console.WriteLine(i);

}

Console.ReadKey();

}

}

**C# Methods:** A method is a block of code which only runs when it is called. we can pass data, known as parameters, into a method. Methods are used to perform certain actions, and they are

also known as functions.

**Syntax:**

class Program{

static void MyMethod() {

// code to be executed

}

}

**C# Call By Value:** In C#, value-type parameters are that pass a copy of original value to the

function rather than reference. It does not modify the original value. A change made in passed

value does not alter the actual value.

**Syntax**:

public void add(int a, int b){

int r = a + b;

}

**C# Call By Reference**: C# provides a ref keyword to pass argument as reference-type. It passes

reference of arguments to the function rather than copy of original value. The changes in passed

values are permanent and modify the original variable value.

**Syntax:**

public void add(ref int a, ref int b){

a = a+b;

}

add(ref a, ref b) // method call by reference.

**C# Params:** In C#, params is a keyword which is used to specify a parameter that takes variable

number of arguments. It is useful when we don't know the number of arguments prior. Only one

params keyword is allowed and no additional parameter is permitted after params keyword in a

function declaration.

Syntax:

public static int Add(params int[] ListNumbers)

{

//statements;

}

**C# out Parameter:**

C# provides out keyword to pass arguments as out-type. It is like reference-type, except that it

does not require variable to initialize before passing. We must use out keyword to pass argument

as out-type. It is useful when we want a function to return multiple values. We use this keyword

for getting data in a unidirectional manner.

**Syntax:**

public static void MyMethod(out int a, out int b)

{

a = 5;

b = 2;

}