PROGRAMACIÓN ORIENTADA A OBJETOS
1er. PARCIAL

Practica #1
Estructuras Condicionales en C#

Daniel Vázquez de la Rosa
d_vazquez@outlook.com

Instituto Politécnico Nacional

Escuela Superior de Cómputo

Agosto 2018

Datos Personales

Crear un Documento en Word que incluya una caratula que contenga tus datos. Dicho documento será guardado como: Apellido Paterno_NombreAlumno_Practica1_POO y deberá contener toda la información de la practica creada.

Objetivos de la Práctica

- Utilizar las instrucciones de selección if e if..else para elegir una de varias acciones alternativas.
- Conocer la sintaxis de C# para las instrucciones condicionales simples, dobles y múltiples, así como también la utilidad en la programación.
- Aprender a utilizar la estructura y sintaxis del switch-case para la evaluación de condiciones múltiples.

Introducción

Visual C#

Visual Studio .NET es un entorno de programación repleto de herramientas que contiene toda la funcionalidad necesaria para la creación de proyectos de C# grandes o proyectos de C# pequeños.

C# es un lenguaje de programación que se ha diseñado para compilar diversas aplicaciones que se ejecutan en .NET Framework. C# es simple, eficaz, con seguridad de tipos y orientado a objetos; no obstante, perfectamente pueden crearse aplicaciones utilizando el paradigma de la programación estructurada. Las numerosas innovaciones con que cuenta C#, permiten desarrollar aplicaciones rápidamente y mantener la expresividad y elegancia de los lenguajes de estilo de C.

En el primer ejercicio se inicia el entorno de programación de Visual Studio .NET y se enseña a crear una aplicación en consola.

Para este curso utilizaremos la versión de Visual Studio 2010, 2012, 2013, 2015. C#, permite trabajar en modo consola y modo gráfico

En el cuerpo del programa se incluyen las variables a utilizar, asignaciones, procesos, cálculo de resultados, etc.

- **Variables**: Es el lugar (espacio de memoria) donde se almacenan los datos a utilizar y éstas pueden ser de un tipo de datos particular.
- **Constantes**: Son los datos o valores que no cambian durante la ejecución de un programa.

TIPOS DE DATOS

Los diferentes objetos de información con los que un programa trabaja se denominan datos.

Todos los datos tienen un tipo asociados con ellos que nos servirá para poder conocer con que información trabajaremos. Es decir, cuando ingresemos el sueldo de un trabajador necesitamos que este contenga decimales, o al solicitar la edad de una persona está tiene que estar con números enteros, etc. Además, la suma entre caracteres no tiene sentido.

La asignación de tipos a los datos tiene dos objetivos principales:

- Detectar errores de operaciones aritméticas en los programas.
- Determinar cómo ejecutar las operaciones.

Tipos de Datos Comunes:

Estos son los tipos de datos más utilizados en los lenguajes de programación:

- Numéricos.
- Caracteres.
- Lógicos.

TIPOS NUMÉRICOS	TIPOS CARACTER	TIPOS LÓGICOS
Dentro de estos tipos se puede hacer mención de los tipos enteros, reales, de coma flotante, Decimales y de los exponenciales.	Los tipos carácter se dividen también en caracteres ASCII, como por ejemplo: a, A, &, *, etc. El otro grupo son los strings o cadenas de caracteres, como por ejemplo: “Hola mundo”.	Los tipos lógicos solamente pueden tomar valores verdadero o falso.

OPERADORES UTILIZADOS EN LA PROGRAMACIÓN

Operadores Aritméticos

SÍMBOLO	OPERADOR
+	Suma
-	Resta
*	Multiplicación
/	División
^	Exponentes
Mod (%)	Módulo

Operadores Relacionales

En ocasiones en los programas se necesitan realizar comparaciones entre distintos valores, esto se realiza utilizando los operadores relaciones, los cuales se listan a continuación:

SÍMBOLO	OPERADOR
<	Menor que
>	Mayor que
<=	Menor o igual que
>=	Mayor o igual que
=	Asignación
==	Comparación
!=	Diferente de

Identificadores:

- Dan nombre a variables, constantes y métodos (funciones o procedimientos).
- Constan de caracteres alfanuméricos.
- C# es sensible a mayúsculas y minúsculas.
- No se pueden utilizar palabras reservadas como nombre de variables.
- Deben comenzar con letras y pueden ser seguidas de números.

Instrucciones de Entrada y Salida (E/S) en C#

Para poder mostrar en pantalla y capturar desde teclado, hacemos uso de la clase ***Console,*** que contiene los métodos para mostrar mensajes en pantalla y permite entradas desde teclado.

Cuando se desea tener acceso a los métodos hacemos uso del operador de acceso, el cual será el símbolo (.).

Estructuras de Condicionales en C#

C# cuenta con tres tipos de estructuras condicionales o lógicas.

1) La instrucción **if** realiza (selecciona) una acción si una condición es verdadera, o ignora la acción si la condición es falsa. A la instrucción **if** se le llama **instrucción de selección simple**, debido a que selecciona o ignora una acción individual.

2) La instrucción **if…else** realiza una acción si una condición es verdadera o realiza una acción distinta si la condición es falsa. A la instrucción **if...else** se le llama **instrucción de selección doble**, debido a que selecciona una de dos acciones distintas (o grupos de acciones).

3) La instrucción **switch** realiza una de varias acciones distintas, dependiendo del valor de una expresión (expresión de control). A la instrucción **switch** se le llama **instrucción de selección múltiple**, debido a que selecciona una de varias acciones distintas (o grupo de acciones).

Los operadores lógicos dentro de las estructuras producen un resultado booleano (verdadero o falso) y sus operandos son también valores lógicos. Nos permiten formular condiciones complejas a partir de condiciones simples. A continuación, se muestra una tabla con las tres compuertas lógicas básicas que nos servirán como operadores lógicos:

OPERADOR	C#	SINTAXIS	COMENTARIO
AND	&&	Exp_Lógica && Exp_Lógica	Devuelve verdaderos si se cumplen ambas condiciones.
OR	||	Exp_Lógica || Exp_Lógica	Devuelve verdaderos si se cumple al menos una de las condiciones.
NOT	!	!	Niega la condición.

Desarrollo

Ejemplo 1

Vamos a construir un programa que solicite al usuario un número y determine si dicha cantidad es par o impar, en caso de que el número sea par, el programa deberá verificar si el número está entre el rango [10 - 100].

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   namespace Ejemplo_1
8   {
9       class Par
10      {
11          static void Main(string[] args)
12          {
13              int Numero;
14              Console.Title = "VERIFICANDO NÚMEROS PARES";
15              Console.Write("\n\tIngrese un número: ");
16              Numero = int.Parse(Console.ReadLine());
17              if(Numero < 0)
18              {
19                  Console.Write("\n\tNúmero negativo...Ingrese un número positivo");
20              }
21              else if(Numero%2 == 0)
22              {
23                  Console.Write("\n\tEl número ("+Numero+") es par.");
24                  if(Numero >=10 && Numero <=100)
25                  {
26                      Console.Write("\n\tEl número("+Numero+") se encuentra en el rango [10 - 100]");
27                  }
28                  else
29                  {
30                      Console.Write("\n\tEl número(" + Numero + ") NO se encuentra en el rango [10 - 100]");
31                  }
32              }
33              else
34              {
35                  Console.Write("\n\tEl número ingresado es impar...");
36              }
37              Console.ReadKey();
38          }
39      }
40  }
```

Ejemplo 2

Se pide desarrollar un programa en C# que lea desde teclado dos números. Si el primero es mayor que el segundo se deberá mostrar la suma y la diferencia de dichas cantidades, en caso contrario se deberá mostrar el producto y la división del primero respecto al segundo.

```
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6
7   namespace Ejemplo_2
8   {
9       class Program
10      {
11          static void Main(string[] args)
12          {
13              Double x, y, z, w;
14              Console.Title = "CONDICIONALES";
15              Console.WriteLine("\tPrimer Número: ");
16              x = Double.Parse(Console.ReadLine());
17              Console.WriteLine("\tSegundo Número: ");
18              y = Double.Parse(Console.ReadLine());
19              if(x > y)
20              {
21                  z = x + y;
22                  w = x - y;
23                  Console.WriteLine("\tMOSTRANDO RESULTADOS:");
24                  Console.WriteLine("\tSUMA: "+z);
25                  Console.WriteLine("\tDIFERENCIA: "+w);
26              }
27              else
28              {
29                  z = x * y;
30                  if (y == 0)
31                  {
32                      Console.WriteLine("\tNo se puede dividir entre CERO");
33                  }
34                   else
35                   {
36                       w = x / y;
37                       Console.WriteLine("\tMOSTRANDO RESULTADOS:");
38                       Console.WriteLine("\tPRODUCTO: " + z);
39                       Console.WriteLine("\tDIVISIÓN: " + w);
40                   }
41               }
42               Console.ReadKey();
43          }
44      }
45  }
```

Ejemplo 3

Se necesita un programa que muestre un menú con las siguientes opciones:

1. Suma.
2. Resta.
3. Multiplicación.
4. División.
5. Raíz Cuadrada.
6. Exponenciación.
7. Salir del programa.

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

namespace Ejemplo_3
{
    class Opciones
    {
        static void Main(string[] args)
        {
            int op;
            Double x, y, z;
            Console.WriteLine("\tPrimera cantidad: ");
            x = Double.Parse(Console.ReadLine());
            Console.WriteLine("\tSegunda cantidad: ");
            y = Double.Parse(Console.ReadLine());
            Console.Title = "SWITCH CASE";
            Console.Write("\n\tOPERACIONES MATEMÁTICAS");
            Console.Write("\n\t1. SUMAR");
            Console.Write("\n\t2. RESTAR");
            Console.Write("\n\t3. MULTIPLICAR");
            Console.Write("\n\t4. DIVIDIR");
            Console.Write("\n\t5. RAIZ CUADRADA");
            Console.Write("\n\t6. POTENCIACIÓN");
            Console.Write("\n\t7. SALIR DEL PROGRAMA");
            Console.Write("\n\n\tDigite su opción:  ");
            op = int.Parse(Console.ReadLine());
            switch (op)
            {
                case 1:
                    z = x + y;
                    Console.WriteLine("\tEl resultado de la suma es: "+z);
                    break;

                case 2:
                    z = x- y;
                    Console.WriteLine("\tEl resultado de la resta es: " + z);
                    break;

                case 3:
                    z = x * y;
                    Console.WriteLine("\tEl resultado de la multiplicación es: " + z);
                    break;

                case 4:
```

```
48                      if (y == 0)
49                      {
50                          Console.WriteLine("\tDivisión Inválida...");
51                      }
52                      else
53                      {
54                          z = x / y;
55                          Console.WriteLine("\tEl resultado de la división es: " + z);
56                      }
57                      break;
58
59                  case 5:
60                      z = Math.Sqrt(x);
61                      Console.WriteLine("\tLa raíz cuadrada del primer número es: "+z);
62                      break;
63
64                  case 6:
65                      z = Math.Pow(y,2);
66                      Console.WriteLine("\tEl cuadrado el segundo número es: "+z);
67                      break;
68
69                  case 7:
70                      Environment.Exit(0);
71                      break;
72
73                  default:
74                      Console.WriteLine("\tOpción no definida...intente de nuevo...");
75                      break;
76              }
77              Console.ReadKey();
78          }
79      }
80  }
```

NOTA: Siempre hay que utilizar las validaciones para que las operaciones matemáticas tengan el resultado correcto.

Responda

Una vez que has hecho estos ejemplos, deberás de contestar o realizar los siguientes ejercicios

1. Creé un programa que pida 3 números al usuario y determine cuál de ellos es el mayor.
2. Se requiere calcular la edad de un individuo; para ello se va a tener como entrada dos fechas en el formato día (1 a 31), mes (1 a 12) y un año (entero de cuatro dígitos), correspondientes a la fecha de nacimiento y la fecha actual, respectivamente. Escriba un programa que calcule y muestre la edad del individuo. Si es la fecha de un bebé (menos de un año de edad), la edad se debe dar en meses y días; en caso contrario, la edad se calculará en años.
3. Las raíces reales de la expresión $ax^2 + bx + c = 0$. Desarrollar un programa para encontrar las raíces de la ecuación cuadrática.

Esta Práctica se deberá de entregar a más tardar el día lunes 20/08/2018 antes de las 10:00 PM, toda práctica que no sea enviada dentro de este horario será evaluada con cero.

Bibliografía

- Deitel, Harvey M. y Paul J. Deitel, Cómo Programar en C#, Segunda Edición, México, 2007.