Guía #7

Tema: “Estructuras de Repetición en C# [While y Dowhile]”

Objetivos

- Utilizar las estructuras de repetición para ejecutar instrucciones repetidas en una aplicación.
- Solucionar problemas con programas que integren la estructura de selección y estructuras repetitivas **While** y **Do while**.

Introducción

Una estructura repetitiva permite ejecutar una instrucción o un conjunto de instrucciones varias veces. Una ejecución repetitiva de sentencias se caracteriza por:

- La o las sentencias que se repiten.
- El test o prueba de condición antes de cada repetición, que motivará que se repitan o no las sentencias.

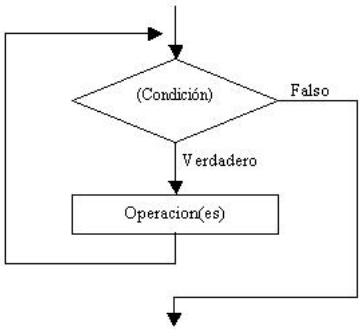
El bucle while

Repite la ejecución de una sentencia o de un grupo de sentencias mientras la condición que se evalúa (al principio del bucle) sea verdadera.

Sintaxis:

```
while(condición_de_bucle)
{
  Sentencias;
}
```

Representación gráfica de la estructura while:



No debemos confundir la representación gráfica de la estructura repetitiva while (Mientras) con la estructura condicional if (Si)

Funcionamiento:

- En primer lugar se verifica la condición, si la misma resulta verdadera se ejecutan las operaciones que indicamos por la rama del Verdadero.A la rama del verdadero la graficamos en la parte inferior de la condición. Una línea al final del

bloque de repetición la conecta con la parte superior de la estructura repetitiva.

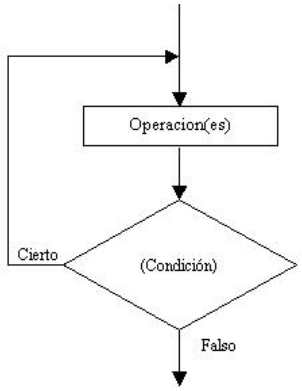
- En caso que la condición sea Falsa continúa por la rama del Falso y sale de la estructura repetitiva para continuar con la ejecución del algoritmo.
- El bloque se repite MIENTRAS la condición sea Verdadera.

Importante: Si la condición siempre retorna verdadero estamos en presencia de un ciclo repetitivo infinito. Dicha situación es un error de programación, nunca finalizará el programa.

El bucle do while

La estructura do while es otra estructura repetitiva, la cual ejecuta al menos una vez su bloque repetitivo, a diferencia del while o del for que podían no ejecutar el bloque. Esta estructura repetitiva se utiliza cuando conocemos de antemano que por lo menos una vez se ejecutará el bloque repetitivo.
La condición de la estructura está abajo del bloque a repetir, a diferencia del while o del for que está en la parte superior.

Representación gráfica de la estructura do while:



Funcionamiento:

- El bloque de operaciones se repite MIENTRAS que la condición sea Verdadera.
- Si la condición retorna Falso el ciclo se detiene. En C#, todos los ciclos repiten por verdadero y cortan por falso.
- Es importante analizar y ver que las operaciones se ejecutan como mínimo una vez.

No hay que confundir los rombos de las estructuras condicionales con los de las estructuras repetitivas do while.

Sintaxis:

```
do
{
Sentencias;
} while (condición de bucle);
```

Material y Equipo

- Guía de laboratorio No. 7.
- Computadora con Visual Studio 2013 o superior.
- Dispositivo de almacenamiento (USB).

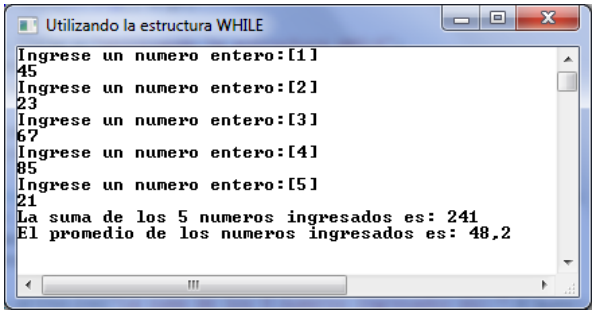
Procedimiento

Ejemplo1

Escriba un programa que lea 5 números desde teclado y presente la sumatoria de los mismos. Hacer uso de la estructura repetitiva while.

```
1    {
2    Console.Title = "Utilizando la estructura WHILE";
3    int contador = 1;
4    Double suma = 0;
5    int N;
6    Double prom;
7    while (contador <= 5)
8     {
9      Console.WriteLine("Ingrese un numero entero:[{0}] " , contador);
10     N = int.Parse(Console.ReadLine());
11     suma = suma + N;
12     contador = contador + 1;
13    }
14   Console.WriteLine("La suma de los 5 numeros ingresados es: " + suma);
15   prom = suma / 5;
16   Console.WriteLine("El promedio de los numeros ingresados es: " +
     Math.Round(prom,3));
17   Console.ReadKey();
18   }
```

Corrida del ejercicio:

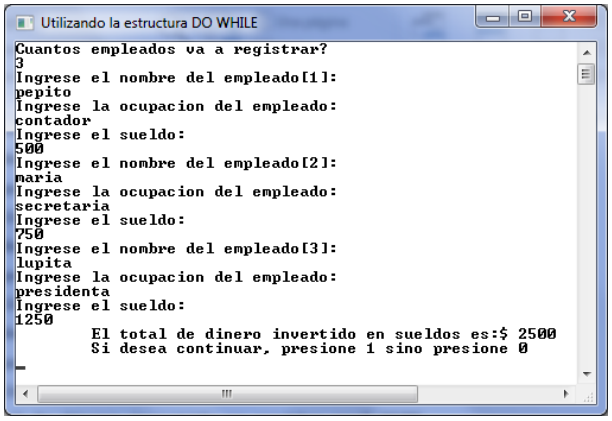


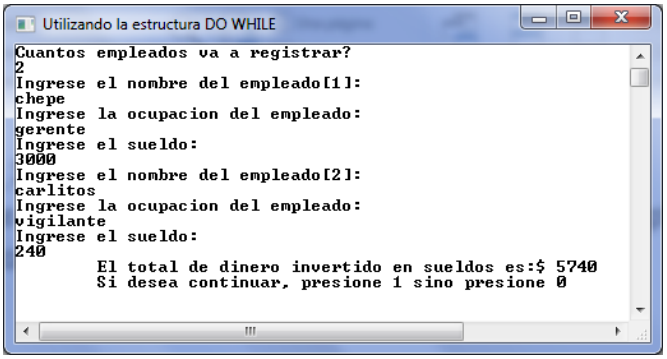
Ejemplo2

Programa que combina uso de for y do while. Esta aplicación captura desde teclado los datos de uno o varios empleados y calcula el total invertido en sueldos.

```
1    {
2     Console.Title = "Utilizando la estructura DO WHILE";
3     int op, i, cantidad;
4     String nombre, ocupacion;
5     Double sueldo;
6     Double total = 0;
7     do
8     {
9      Console.Clear();
10     Console.WriteLine("Cuantos empleados va a registrar?");
11     cantidad = int.Parse(Console.ReadLine());
12     for (i=1; i<=cantidad; i=i+1)
13      {
14        Console.WriteLine("Ingrese el nombre del empleado[{0}]: ", i);
15        nombre = Console.ReadLine();
16        Console.WriteLine("Ingrese la ocupacion del empleado: ");
17        ocupacion = Console.ReadLine();
18        Console.WriteLine("Ingrese el sueldo: ");
19        sueldo = Double.Parse(Console.ReadLine());
20        total = total + sueldo;
21      }
22    Console.WriteLine("\tEl total de dinero invertido en sueldos es:$ " + total);
23    Console.WriteLine("\tSi desea continuar, presione 1 sino presione 0");
24    op = int.Parse(Console.ReadLine());
25   } while (op == 1);
26    Console.ReadKey();
27   }
```

Corrida del ejercicio:





El numero primo

Convertir el siguiente algoritmo en código de C# que permite visualizar si un número es primo o no lo es, recordando que un número primo es todo aquel número que es divisible por sí mismo y por la unidad.

Por ejemplo:

1) 7 es primo porque solo se puede dividir entre 7 y entre 1.
2) 9 no es primo porque se puede dividir entre 9, 3 y 1, por lo tanto no es primo.
3) 13 es primo porque solo se puede dividir entre 13 y 1.
4) 27 no es primo porque se puede dividir entre 27, 9, 3, 1.

Sin olvidar que el operador **MOD** se refiere al módulo de una división entera y su símbolo es **%.**

Convertir el siguiente algoritmo en su correspondiente código en C#:

```
Inicio
        Entero numero, contador, k
        Leer (numero)
        contador = 0
        k = 1
        Mientras k<= numero
                Si ((numero MOD k) == 0) Entonces
                        contador = contador + 1
                FinSi
                k = k +1
        FinMientras
        Si contador == 2 Entonces
                Escribir ("Es un numero Primo")
        Sino
                Escribir ("El numero no es primo")
        FinSi
Fin
```

El numero perfecto

Convertir el siguiente algoritmo en código de C# que permite visualizar si al digitar un número entero me diga si es perfecto o no, tomando en cuenta que:

Un número es perfecto si la suma de sus divisores menores a él es el mismo número, por ejemplo:

1) 28 = 1+2+4+7+14
2) 496 = 1+2+4+8+16+31+62+124+248
3) 8128 = 1+2+4+8+16+32+64+127+254+508+1016+2032+4064

Sin olvidar que el operador **DIV** se refiere a división entera y su respectivo símbolo es **/**

Convertir el siguiente algoritmo en su correspondiente código en C#:

```
Inicio
        Entero dato, k, suma
        Leer (dato)
        k = 1
        suma = 0
        Mientras (k <= (dato DIV 2))
                Si ((dato MOD k) == 0) Entonces
                        suma = suma + k
                FinSi
                k = k + 1
        FinMientras
        Si (suma == dato) Entonces
                Escribir ("Es numero perfecto...”)
        Sino
                Escribir ("No es perfecto...”)
        FinSi
Fin
```

Ejemplo5

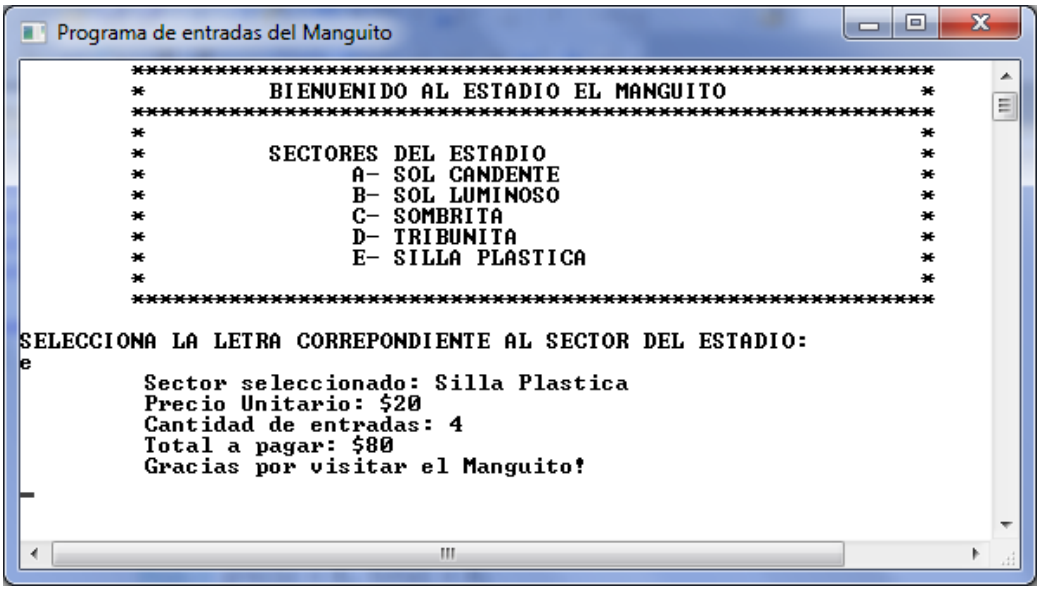
El Estadio el Manguito es uno de los estadios más importantes de Soyapango. Este estadio tiene diversos sectores. El costo de la entrada a los partidos del estadio se asigna en virtud de los sectores del estadio mediante la siguiente tabla:

Sector	Costo de la entrada
Sol Candente	$ 3
Sol Luminoso	$ 5
Sombrita	$ 8
Tribunita	$ 15
Silla Plástica	$ 20

Se pide desarrollar un programa que permita seleccionar un sector del estadio, ingresar la cantidad de entradas solicitadas y calcular el total a pagar por las entradas.

```
1     static void Main(string[] args)
2      {
3      Console.Title = "Programa de entradas del Manguito";
4      int cantidad = 0;
5      String op;
6      Double precio = 0, total = 0;
7      Console.WriteLine("Hey que ondas aqui estas en el estadio El Manguito");
8      Console.WriteLine("Ingresa la cantidad de entradas requeridas: ");
9      cantidad = int.Parse(Console.ReadLine());
10     Console.Clear();
11     if (cantidad < 1)
12     {
13     Console.WriteLine("\t ERROR: La cantidad de entradas debe ser mayor que 1");
14     Console.ReadKey();
15     Environment.Exit(0);
16     }
17     Console.Clear();
18     Console.WriteLine("\t*************************************************************");
19     Console.WriteLine("\t*         BIENVENIDO AL ESTADIO EL MANGUITO              *");
20     Console.WriteLine("\t*************************************************************");
21     Console.WriteLine("\t*                                                           *");
22     Console.WriteLine("\t*         SECTORES DEL ESTADIO                              *");
23     Console.WriteLine("\t*               A- SOL CANDENTE                             *");
24     Console.WriteLine("\t*               B- SOL LUMINOSO                             *");
25     Console.WriteLine("\t*               C- SOMBRITA                                 *");
26     Console.WriteLine("\t*               D- TRIBUNITA                                *");
27     Console.WriteLine("\t*               E- SILLA PLASTICA                           *");
28     Console.WriteLine("\t*                                                           *");
29     Console.WriteLine("\t*************************************************************");
30     Console.WriteLine("");
31     Console.WriteLine("SELECCIONA LA LETRA CORREPONDIENTE AL SECTOR DEL ESTADIO: ");
32     op = Console.ReadLine();
33     switch(op)
34     {
35     case "A": case "a":
36     Console.WriteLine("\t Sector seleccionado: Sol Candente");
37     precio = 3;
38     break;
39     case "B": case "b":
40     Console.WriteLine("\t Sector seleccionado: Sol Luminoso");
41     precio = 5;
42     break;
43     case "C": case "c":
44     Console.WriteLine("\t Sector seleccionado: Sombrita");
45     precio = 8;
46     break;
47     case "D": case "d":
```

```
48      Console.WriteLine("\t Sector seleccionado: Tribunita");
49      precio = 15;
50      break;
51      case "E": case "e":
52      Console.WriteLine("\t Sector seleccionado: Silla Plastica");
53      precio = 20;
54      break;
55      default:
56      Console.WriteLine("\t ERROR: El sector seleccionado no existe");
57      Console.ReadKey();
58      Environment.Exit(0);
59      break;
60      }
61      total = precio * cantidad;
62      Console.WriteLine("\t Precio Unitario: $" + precio);
63      Console.WriteLine("\t Cantidad de entradas: " + cantidad);
64      Console.WriteLine("\t Total a pagar: $" + total);
65      Console.WriteLine("\t Gracias por visitar el Manguito!");
66      Console.ReadKey();
67     }
```



NOTA:

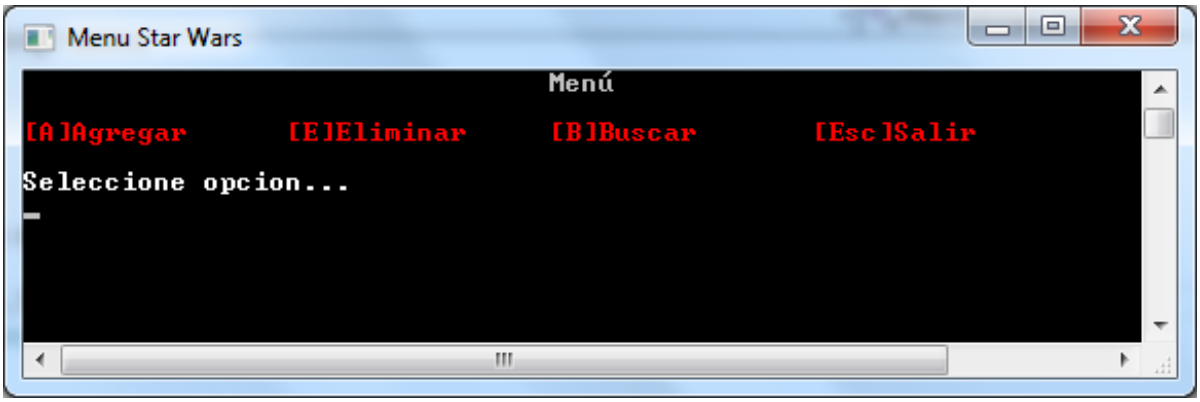
Después de haber utilizado este sistema de entradas, se le contrata a su persona para que posea una opción de SALIR, sin olvidar que, utilizando alguna estructura repetitiva se pueda obtener que el menú de opciones siempre se encuentre activo hasta que el usuario decida salir del programa.

Ejemplo6

Star Wars:

Digitar este código y ejecutarlo para saber lo que realiza.

```
{
ConsoleKeyInfo op;
do
 {
  Console.Clear(); //Limpiar la pantalla
  Console.WriteLine("\t\t\t\tMenú \n");
  Console.ForegroundColor = ConsoleColor.Red;
  Console.Write("[A]Agregar\t");
  Console.Write("[E]Eliminar\t");
  Console.Write("[B]Buscar\t");
  Console.Write("[Esc]Salir\t\n\n");
  Console.ForegroundColor = ConsoleColor.White;
  Console.WriteLine("Seleccione opcion...");
  op = Console.ReadKey(true);//Que no muestre la tecla señalada
  //métodos son acciones, las propiedades son valores
  switch (op.Key)
   {
     case ConsoleKey.A:
     Console.WriteLine("Ud seleccionó la opción Agregar");
     Console.Write("Presione una tecla para continuar...");
     Console.ReadKey();
     break;
     case ConsoleKey.E:
     Console.WriteLine("Ud seleccionó la opción Eliminar");
     Console.Write("Presione una tecla para continuar...");
     Console.ReadKey();
     break;
     case ConsoleKey.B:
     Console.WriteLine("Ud seleccionó la opción Buscar");
     Console.Write("Presione una tecla para continuar...");
     Console.ReadKey();
     break;
     case ConsoleKey.Escape:
     Console.WriteLine("Chao");
     Console.ReadKey();
     break;
   }
 } while (op.Key != ConsoleKey.Escape);
```



Análisis de Resultados

1. Desarrolle un programa que presente el siguiente menú:
    a) Seno de x.
    b) Coseno de x.
    c) Tangente de x.
    d) Raiz cuadrada de x
    e) Potencia de Y^x
    f) Verificar si un número es par o impar.
    g) Salir del programa.

    Consideraciones:
    - El menú tiene que estar activo hasta que el usuario decida salir del programa.
    - Cada vez que se seleccione una opción deberá realizar una limpieza de pantalla para más orden en la aplicación.
    - Se recomienda la utilización del switch case y la estructura repetitiva do while.
2. Que pida un número entero del rango entre 1 y 15 y muestre en pantalla el mismo número de asteriscos.
3. Que pida dos números y muestre todos los números pares que van desde el primero al segundo. Se debe controlar que los valores son correctos.

Investigación Complementaria

1. Se ingresan las temperaturas de cada dia de la semana, determinar e informar:
    - Promedio de temperatura semanal
    - El día mas frio y el más caluroso
    - Porcentaje de temperaturas bajo cero

2. Hacer un programa para una minitienda, el programa deberá tener un menú: Hacer una venta y Cerrar el dia.

   En Hacer una venta, pedirá, la cantidad y el precio unitario del producto, luego preguntara si desea ingresar otro producto a la venta. Cuando ya haya ingresado todos los productos, mostrara el total a pagar y solicitara el monto con que cancela el cliente, luego muestra el vuewlto a devolver, hace una pausa y luego vuelve al menú principal.

   La opción Cerrar el dia, muestra la cantidad de productos total vendido y el total de ventas del dia, hace una pausa y se cierra.

3. Supóngase que en una reciente elección hubo cuatro candidatos (con identificadores 1,2,3 y 4). Usted habrá de encontrar, mediante un programa, el número de votos correspondiente a cada individuo y el porcentaje que obtuvo respecto al total de los votantes. El usuario tecleara los votos de manera desorganizada, tal y como se obtuvieron en la elección, el final de datos está representado por un cero.

   Observe, como ejemplo, la siguiente lista:

   1 3 2 3 4 2 3 4 4 1 2 1 2 4 0

   Donde 1 representa un voto para el candidato1, 3 un voto para el candidato3 y así sucesivamente.

Bibliografía

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