Unidad 2 - Capítulo 1 - Laboratorio 1

1. Estructuras de Decisión

Objetivos

- Utilizar estructuras de decisión con if anidados.
- Mostrar resultados a partir del ingreso de valores del usuario.
- Utilizar métodos de `System.Console y System.String`

Duración Aproximada

30 minutos

Pasos

1) Crear una nueva solución **Lab01** con un proyecto en modo consola llamado **Decision**. Ubicación: ***C:\Net\Unidad02\Capitulo01***

2) Pedir al usuario que ingrese un texto. Asignar el texto ingresado a una variable que se llame inputTexto, utilizando `Console.ReadLine().`

3) Validar que se haya ingresado texto en inputTexto, de lo contrario, mostrar mensaje y terminar la aplicación.

4) Mostrar un menú que de opciones de mostrar la frase ingresada en mayúsculas (usando `.ToUpper()` del objeto string), en minúsculas (usando `.ToLower()`), o mostrar la cantidad de caracteres que tiene (usando `Lenght`). Permitir que el usuario pueda seleccionar la opción 1, 2 o 3 solamente. Asignarla a la variable opción, de la siguiente manera:

   ```
   ConsoleKeyInfo opcion = Console.ReadKey();
   ```

   ReadKey devuelve información sobre el input del usuario. Esta información debe ser utilizada para descubrir cuál fue la opción.

5) Armar un if anidado, que muestre los resultados esperados en función de la opción. Si no se eligió ninguna de las opciones esperadas, deberá mostrar un mensaje y salir de la aplicación. Para crear las condiciones del if, utilizar `opcion.Key` para obtener la tecla presionada por el usuario y `ConsoleKey.D1, ConsoleKey.D2, y ConsoleKey.D3`, que son los números 1, 2 y 3.

   Observar que .NET provee en `ConsoleKey` cada una de las teclas, como un enumerado, lo que resulta más fácil las comparaciones, y evita confusiones entre el uso del char, del valor ASCII, etc. Siempre se compararán elementos del tipo `ConsoleKey`.

6) Ejecutar la aplicación y observar que los resultados sean los esperados.

2. Estructuras de Decisión (bis)

Objetivo

- Conocer el uso de la estructura Case

Duración Aproximada

20 minutos

Pasos

1) Repetir el ejercicio anterior, esta vez utilizando una estructura case.

3. Estructuras de iteración

Objetivo

- Uso de un array de elementos en conjunto con una estructura de iteración for.
- Crear un vector de cadenas de texto, llenarlo y luego mostrar el contenido ordenado inversamente a como se ingresó.

Duración Aproximada

20 minutos

Pasos

1) Agregar un proyecto llamado **Iteracion** a la solución **Lab01**

2) Crear un array de strings, las posiciones se determinarán por la variable `cantIteraciones`, un entero al que se le asignarán 5 posiciones. Luego, siempre se utilizara la misma variable

3) Crear un ciclo for para ingresar cada una de las posiciones.

```
for (int i = 0; i < cantIteraciones; i++)
```

4) Crear un ciclo for para mostrar por pantalla cada una de las posiciones, una por línea, comenzando desde la última que se ingresó, hasta la primera.

5) Ejecutar la aplicación.

4. Ejercicios para pensar

Objetivo

- Realizar prácticas de sintaxis del lenguaje y aplicación de los algoritmos de programación.

Pasos

Observación: Construir los siguiente ejercicios en una nueva solución llamada **Lab01.ParaPensar** que incluya los proyectos que crea necesarios a medida que los va realizando. Ubicación: ***C:\Net\Unidad02\Capitulo01***

1) Construir una aplicación que sume dos números y proporcione el resultado con el formato siguiente: **El resultado de la suma de < número uno > y < número dos > es < resultado >.**

2) Crear una aplicación que te pida un año y verifique si el año es bisiesto o no.

   Observación: Un año es bisiesto si es divisible por 4, excepto el último de cada siglo (aquel divisible por 100), salvo que éste último sea divisible por 400.
   Es decir los años que sean divisibles por 4 serán bisiestos; aunque no serán bisiestos si son divisibles entre 100 (como los años 1700, 1800, 1900 y 2100) a no ser que sean divisibles por 400 (como los años 1600, 2000 ó 2400))

3) Hacer un programa para calcular la suma de la serie de Fibonacci.

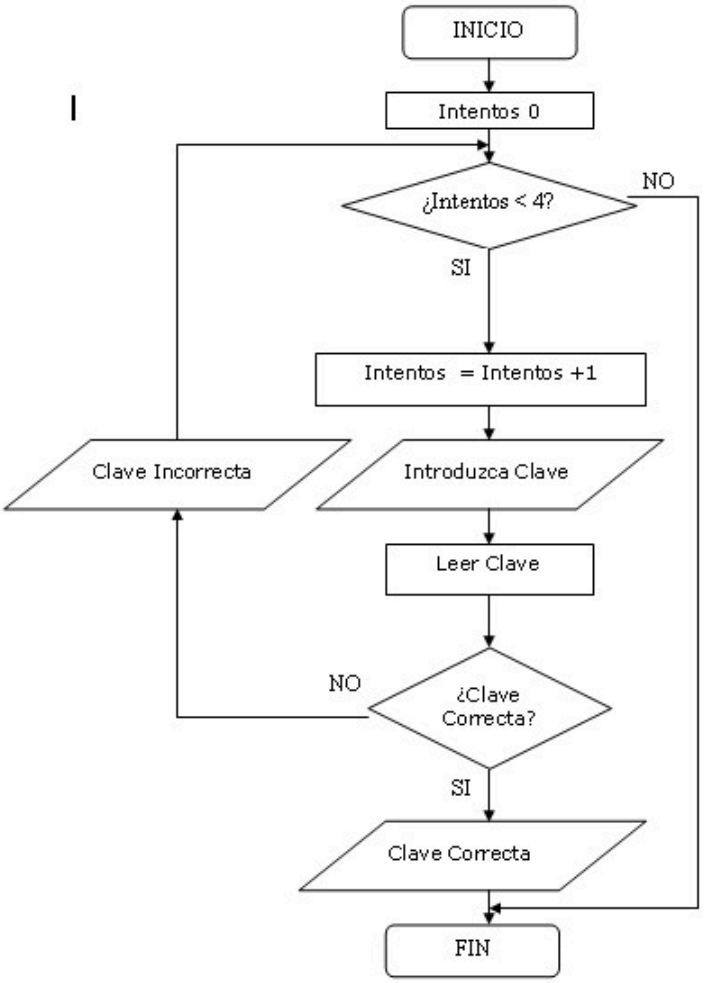
4) Construir una aplicación que proporcione todos los números pares entre el 1 y el 100.

5) Construir una aplicación que reciba el nombre de un mes del año como parámetro y proporcione su número correspondiente. Debe ser con el formato: **< Nombre del mes > + < número del mes >.**

6) Dado un número entero, que se convierta a número romano.

7) Calcular los N primeros números primos gemelos.

8) Realizar un programa basado en el siguiente algoritmo para el ingreso de una clave:



9) Crear una aplicación que pida un número de filas y respecto a estas, dibuje un triángulo como el siguiente:

```
*
***
*****
*******
*********
```