Práctica 2: Estructuras de Control Selectivas

Objetivos

- Comprender el uso de las estructuras de control de flujo selectivas (**simple, doble, múltiple**).
- Conocer la implementación en C++ de las estructuras de control anteriores.
- Aplicar la depuración de programas para detectar y corregir errores lógicos y de ejecución.

`https://es.wikipedia.org/wiki/Depurador`

Para depurar el programa es mejor instalar la versión **`Dev-C++ 5.10 (Orwell).`**
https://sourceforge.net/projects/orwelldevcpp/files/Setup%20Releases/Dev-Cpp%205.10%20TDM-GCC%204.8.1%20Setup.exe/download

Estructuras de Control Condicionales

Las estructuras de control selectivas permiten decidir qué ejecutar y qué no ejecutar en un programa. Por ejemplo: realizar una división sólo si el divisor es distinto de cero. Permiten implementar la toma de decisiones.

***Alternativa simple*:** tenemos un bloque de sentencias que se ejecuta si se cumple una condición. En C++ se usa la instrucción **`if.`**

***Alternativa doble*:** corresponde a dos bloques de sentencias. Se ejecuta **uno u otro** dependiendo de si se cumple o no una condición. En C++ se usa la instrucción **`if-else`**.

***Alternativa múltiple*:** existen más de dos alternativas. Se ejecuta uno de los bloque dependiendo del valor de una expresión. En C++ se usa la instrucción **`switch`**.

Algunos ejemplos:

```
//Una alternativa

  if (b != 0)
   resultado = a / b;



//Dos alternativas

  if (a >= 0)
      cout << "positive o igual a cero";
  else
      cout << "negativo";
```

```
//Más de dos alternativas sentencia switch

   switch (car)
   {
       case 's':
          cout << "sobresaliente";
          break;
       case 'a' :
          cout << "aprobado";
          break;
       case 'n' :
          cout << "notable";
          break;
       default://otros casos
          cout << "no valido";
    }
```

Recuerda

1. En una sentencia con **if-else** anidados cada **else** se corresponde con el **if** precedente más cercano.
2. Si existe más de una instrucción dentro del bloque del **if** o del **else** deben ir entre llaves.
3. El selector de una sentencia **switch** debe ser de **tipo entero o compatible (caracteres o bool)**. Ni las variables ni las constantes reales pueden ser utilizadas en las etiquetas **case**. Asegura que el selector y las etiquetas **case** son del mismo tipo.
4. Si el selector toma un valor no listado en las etiquetas **case**, no se ejecutará ninguna acción; por esta causa se suele poner una etiqueta **default**.
5. Una sentencia **switch** se puede implementar mediante **if-else** anidados.

BLOQUE DE EJERCICIOS

Ejercicio 1 (depuración de código, ecuación 2º grado, ejercicio1.cpp). Abre el fichero ejercicio1.cpp. *¡Cuidado: contiene posibles errores de ejecución y lógicos!* En `Herramientas >> Opciones_del compilador >> Configuración >> Linker` activa la opción `Generar información de Debug`. Fija puntos de ruptura en las líneas 34, 38, 39 y 40. Se establecen desde el menú `Debug>>Toggle BreakPoint`. Pulsa en el menú inferior la pestaña `Depurar`.

Cuando el programa se detenga en un punto de ruptura, visualiza el contenido de las variables `discr, x1, x2`. Para ello, añade varios `watch`, uno para cada variable. Click en `Next Step`, para ejecutar instrucción a instrucción. Rellena la tabla siguiente usando el valor que se muestra en los `watch`.

- Indica el contenido de `discr, x1, x2` para a = 1, b = 6 y c = 2.
- Indica el contenido de `discr, x1, x2` para a = 1, b = 2 y c = 1.
- Indica el contenido de `discr, x1, x2` para a = 3, b = 1 y c = 2.

	a=1, b=6, c=2			a=1, b=2, c=1			a=3, b=1, c=2		
	discr	x1	x2	discr	x1	x2	discr	x1	x2
Línea 34									
Línea 38									
Línea 39									
Línea 40									

Corrige el error lógico y añade el código necesario para que en el caso de que la ecuación no tenga solución se emita el mensaje "`Error en los valores de los coeficientes, la ecuación no tiene solución.`".

Ejercicio 2 (depuración de código, función x^3, ejercicio2.cpp). Abre el fichero ejercicio2.cpp. *¡Cuidado: contiene errores!* Fija puntos de ruptura en las líneas 35 y 37. Visualiza el contenido de las variables `x, res`, usando `watch`. Rellena la tabla siguiente.

	x = 2	x = 45	x = -33
Línea 35	res = ?	res = ?	res = ?
Línea 37	res = ?	res = ?	res = ?

¿Cómo se explican estos resultados? ¿Cómo se denomina este fenómeno?

Añade el código necesario (**sin modificar el tipo de dato de las variables**) para que en el caso de que se introduzcan valores para la variable x que el programa no está preparado para procesar se informe con el siguiente mensaje de error "`Error en el valor de x. El programa no está diseñado para procesar x^3.`"

Ejercicio 3 (DNI, ejercicio3.cpp). Implementa un programa en C++ que tome como dato de entrada por teclado el Documento Nacional de Identidad (un número entero positivo) de una persona e imprima su letra por pantalla. El DNI consta de 8 dígitos y de una letra. La letra se obtiene a partir de los dígitos del siguiente modo:

1) Calcular el resto de dividir el DNI entre 23. Será un número entre 0 y 22;
2) Selecciona la letra asociada al DNI, según el valor obtenido en el resto, en la tabla:

0	1	2	3	4	5	6	7	8	9	10	11
T	R	W	A	G	M	Y	F	P	D	X	B

12	13	14	15	16	17	18	19	20	21	22
N	J	Z	S	Q	V	H	L	C	K	E

Indica el diagrama de flujo. Documenta el programa usando los comandos de Doxygen y genera la documentación.

Ejercicio 4 (Estaciones, ejercicio4.cpp). Escribe un programa en C++ que lea por teclado un día del año comprendido entre 1 y 365 (ambos inclusive) e indique como salida por pantalla si es primavera, verano, otoño o invierno. Asume que se trata de un año no bisiesto. Por ejemplo, si el usuario introduce el día 35, se debe imprimir por pantalla:

```
El dia 35 es INVIERNO.
```

Indica el diagrama de flujo. Documenta el programa usando los comandos de Doxygen.

Ejercicio 5 (gasolinera, ejercicio5.cpp) Implementa un programa de "selfservice" para gestionar una gasolinera. Primero, se introduce por teclado la cantidad de litros (valor entero positivo) de combustible que se desea. Segundo, se mostrará un menú de opciones para elegir el tipo de combustible y se leerá la opción seleccionada.

```
Tipo de combustible:
   a) – Gasolina 95 a 1,48 €/l
   b) – Gasolina 98 a 1,64 €/l
   c) – Gasoleo A a 1,35 €/l
```

Finalmente, se mostrará un mensaje con la cantidad de litros, la opción elegida y el importe. Por ejemplo:

```
  Ha puesto 10 litros de Gasolina 98. Pase por caja y pague 16,40 €.
```

Indica el pseudocódigo. Documenta el programa usando los comandos de Doxygen.

IMPORTANTE:

- Todos los programas deben seguir la Guía de Estilo de C++ e incluir los comandos de Doxygen (pdf en Aula Virtual) para generar la documentación.
- Subir a Aula Virtual todos los archivos .cpp (de uno en uno) sin comprimir (enunciado y adicionales). Sólo los sube un miembro de la pareja.
- Fecha de entrega: durante la sesión de vuestro grupo de lab.
- Realizar una foto del pseudocódigo y/o el diagrama de flujo. Subir el archivo junto con los ficheros cpp.