

## Hoja de ejercicios del Tema 3

1. Conversiones de tipos: Prueba el siguiente programa en tu compilador (copia y pega). Comprueba los problemas que se generan en las conversiones inseguras.

```
#include <iostream>
using namespace std;

int main() {
    double real;
    int entero;
    short int corto;
    char caracter;

    // Conversiones "hacia arriba" (seguras)
    cout << "Introduce un carácter: ";
    cin >> caracter;
    corto = caracter;
    cout << "Entero corto: " << corto << endl;
    entero = corto;
    cout << "Entero: " << entero << endl;
    real = entero;
    cout << "Real: " << real << endl;

    // Conversiones "hacia abajo" (inseguras)
    cout << "Introduce un real muy grande: ";
    cin >> real;
    entero = real;
    cout << "Entero: " << entero << endl;
    corto = entero;
    cout << "Entero corto: " << corto << endl;
    caracter = corto;
    cout << "Carácter: " << caracter << endl;

    return 0;
}
```

2. Declara un tipo enumerado `tCalificacion` con los valores `noPresentado`, `suspense`, `aprobado`, `notable`, `sobresaliente` y `matriculaDeHonor`. Luego, declara dos variables `nota1` y `nota2`, y lee en ellas dos calificaciones numéricas

de 0 a 10 (con un decimal). Asigna a dos variables `calif1` y `calif2`, de tipo `tCalificacion`, el valor que les corresponda, de acuerdo con los valores de las correspondientes variables numéricas (0: `noPresentado`). Finalmente, muestra cada nota numérica seguida de la calificación textual que le corresponde. (Sólo mostrará las calificaciones si ambas notas están entre 0 y 10.) El programa usará las funciones adecuadas.

3. Disponemos de cuatro variaciones de un archivo de texto `entrada.txt`:

Caso 1	Caso 2	Caso 3	Caso 4
134 12.4 14256.789 27.4 100	134 12.4 c 1 2 14256.789	13.4 12 .18.2 14.7	13.4 12

Si tenemos el siguiente fragmento de código:

```
int entero1;
float float1;
double double1;
char car1, car2;
ifstream archivo;
archivo.open("entrada.txt"); // Apertura
archivo >> entero1 >> double1 >> car1;
archivo >> float1 >> car2;
...
```

Indica el valor de cada una de las variables después de ejecutar dicho fragmento de código con los cuatro ejemplos de archivo de entrada. *¡Sin ejecutar el código con un compilador!*

4. ¿Cuál será el contenido del archivo `salida.txt` después de ejecutar el siguiente programa? (Indica el resultado sin ejecutar el código en un compilador.)

```
#include <iostream>
using namespace std;
#include <fstream>
#include <iomanip>

int main() {
    ofstream archivo;
    bool llueve = false;
    int i = 35;
    double d1 = 123;
```

```

double d2 = 123.45;
char c = 'x';
float f = 3.14;

archivo.open("salida.txt"); // Creación del archivo
archivo << "Hoy llueve = " << llueve
        << boolalpha << llueve << endl;
archivo << i << right << setw(8) << i << endl;
archivo << d1 << scientific << right << setw(8) << d1 << endl;
archivo << d2 << right << fixed << setw(8)
        << setprecision(3) << d2 << endl;
archivo << c << setw(8) << left << c << endl;
archivo << f << right << setw(5) << setprecision(3)
        << f << endl;
archivo << scientific << d2 << fixed << endl;
archivo << setprecision(8) << d2 << endl;
archivo.close(); // Cierre del archivo

return 0;
}

```

5. Escribe un programa que solicite un carácter por teclado e informe de si es alfanumérico o no (letra o dígito). En caso de ser alfanumérico deberá indicar si es una letra o un dígito. En caso de ser una letra deberá informar de si es minúscula o mayúscula. Para resolver este ejercicio no se deben usar las funciones `isalnum()`, `isalpha()`, `isdigit()`, `islower()` o `isupper()`, sino que se deben definir las correspondientes funciones propias.
6. Debido a una pertinaz sequía se decidió poner en práctica un sistema de cobro de agua que penalice el consumo excesivo tal como indica la tabla siguiente:

Consumo (m <sup>3</sup> )	€/ m <sup>3</sup>
Primeros 100	0,15
De 100 a 500	0,20
De 500 a 1000	0,35
Más de 1000	0,80

Escribe un programa que lea del teclado los metros cúbicos consumidos y muestre en la pantalla el coste de agua total. Ten en cuenta que en la tabla se indica lo que hay que cobrar por los m<sup>3</sup> que se encuentran en el intervalo. Así, si hemos consumido 750 m<sup>3</sup> deberíamos pagar:  $100 * 0,15 + 400 * 0,20 + 250 * 0,35 = 182,50$  €. Usa constantes.

7. Modifica el Ejercicio 19 de la hoja del Tema 2 de forma que utilice una instrucción `switch` en lugar de instrucciones `if`. Además, el programa permitirá al usuario volver a calcular o terminar.

8. Escribe un programa que lea caracteres del teclado hasta que el usuario introduzca un \*. Los caracteres se solicitan e introducen uno a uno, cada uno en una línea (seguidos de Intro). El programa debe contar el número de dígitos, letras mayúsculas y letras minúsculas introducidas y, cuando haya finalizado la introducción de datos, mostrar cuántos caracteres de cada tipo había. El programa usará las funciones desarrolladas en el Ejercicio 5.
9. Modifica el programa anterior de modo que los caracteres se introduzcan todos en la misma línea acabados en salto de línea (Intro) (no en asterisco). ¡Y que cuente también los espacios en blanco!
10. Escribe un programa que lea un número entero positivo del teclado y muestre la suma de sus dígitos. Por ejemplo, si el entero es 932, mostrará 14 ( $9 + 3 + 2$ ). El programa usará una función que calcule la suma de los dígitos de un entero.
11. Modifica el programa anterior de forma que si la suma de los dígitos es mayor que 9, repita el proceso sobre la propia suma, hasta obtener un valor entre 1 y 9. Ése es el que se conoce como *dígito mágico* del número introducido y será calculado con una función del programa.
12. Escribe un programa que lea un número entero positivo del teclado y lo muestre *invertido* en la pantalla. Se entiende por *invertir* el dar la vuelta a los dígitos que componen el número (*su imagen especular*). Esto es, el inverso de 3952 es 2593. El programa utilizará una función que devuelva una cadena con el nº invertido.
13. Escribe un programa que *invierta* cada número entero positivo que se introduzca por teclado. El programa actuará de forma cíclica, finalizando la ejecución cuando se introduzca un número negativo o el cero.
14. Escribe un programa que genere un archivo `output.txt` en el que aparezcan *invertidos* los números enteros estrictamente **positivos** ( $>0$ ) que haya en otro archivo `input.txt`. Cada línea de `input.txt` contendrá un número entero y terminará en una línea con un 0 (centinela).
- | input.txt | output.txt |
|-----------|------------|
| 1234      | 4321       |
| 56        | 65         |
| 1000      | 0001       |
| -987      | 12345      |
| 54321     |            |
| 0         |            |
15. Escribe un programa que lea los números de un archivo `datos.txt` (cada línea contiene un número real positivo) y muestre en la pantalla el mayor de todos ellos. El archivo termina con un 0 como centinela.

16. El cuadrado de un número entero es igual a la suma de tantos números impares consecutivos (desde la unidad) como unidades tiene el número. Es decir,  $3^2$  es igual a  $1+3+5$  (3 impares) y  $5^2$  es igual a  $1+3+5+7+9$  (5 impares).

Implementa un programa que solicite enteros al usuario y muestre en la pantalla su cuadrado calculado con el algoritmo indicado. El programa deberá finalizar cuando se introduzca el valor 0. Usará una función para calcular los cuadrados.

17. Implementa un programa que lea del teclado secuencias de caracteres terminadas en punto y que, para cada secuencia, cuente y muestre en la pantalla el número de blancos, letras y dígitos que hay entre la primera pareja de paréntesis. Si sólo hay un paréntesis de apertura, el recuento seguirá hasta el final de la secuencia. Puede no haber ninguna pareja de paréntesis. El programa solicitará secuencias hasta que se introduzca una línea con sólo el punto.

Un ejemplo de ejecución del programa sería el siguiente:

```
Entrada:      Esto es (una prueba) de secuencia de entrada.↵
Salida:       Blancos: 1      Letras: 9      Dígitos: 0
Entrada:      Esto es la prueba de secuencia (de entrada 2.↵
Salida:       Blancos: 2      Letras: 9      Dígitos: 1
Entrada:      Esto es la prueba de secuencia de entrada 3.↵
Salida:       Blancos: 0      Letras: 0      Dígitos: 0
Entrada:      .↵
Fin del programa
```

Recuerda que con `cin.sync()`; puedes descartar el salto de línea (y lo que pueda haber después del punto), una vez leído el punto final de cada texto.

18. Desarrolla un programa que determine si una secuencia de enteros introducidos por el teclado y terminada en 0 (centinela) es creciente o no (cada elemento es mayor o igual que el anterior).
19. Implementa un programa que lea un número entero N y pida al usuario que averigüe su raíz cuadrada, preguntando tantas veces como sea necesario hasta que el usuario acierte. Se da por buena una respuesta cuando la distancia (error) entre N y el número introducido elevado al cuadrado sea menor que  $10^{-4}$ . Tras cada respuesta del usuario, el programa indicará si el número es mayor o menor que la raíz cuadrada buscada.
20. Escribe un programa que cuente el número de veces que aparece la secuencia xy en un archivo de texto `input20.txt` (terminado en \*).
21. Escribe un programa que indique si es *triangular* cada número entero que haya en un archivo `enteros.txt` (terminado en 0 como centinela). Se dice que un número entero es triangular si es igual a la suma de varios enteros positivos consecutivos empezando desde 1. El programa usará una función que indique si un entero es triangular.

22. Implementa un programa que pida al usuario las calificaciones de los 10 estudiantes de una clase y los guarde en un array (declara el tipo adecuado; se admiten decimales). Después, mostrará esas calificaciones (cada una en una línea), seguida de la media del curso y del número de aprobados y suspensos. Se usará una función para calcular la nota media y otra para el número de suspensos.
23. Modifica el programa anterior para que se puedan guardar hasta 100 calificaciones. El programa pedirá calificaciones al usuario hasta que éste introduzca una nota negativa (o se llene el array). El programa, además, mostrará la nota máxima llamando a una función que se encargue de encontrarla.
24. Escribe un programa en C++ que lea de un archivo `palabras.txt` (con xxx como centinela) una lista de hasta 50 palabras (cadenas) y luego muestre al usuario un menú con estas opciones: mostrar la lista de palabras (una por línea precedida de su posición), localizar una palabra (indicará su posición), contar las apariciones de una palabra, localizar la primera palabra de más de una longitud dada, desplazar la lista de palabras (cada palabra pasa a la siguiente posición del array; la última a la primera posición) y salir. El programa repetirá el menú hasta que se elija la opción para salir y usará las funciones adecuadas.
25. Implementa un programa que calcule el primer número natural cuyo cubo supera estrictamente otro entero N dado ( $N \geq 0$ ). El programa mostrará la secuencia de números recorrida.
26. Resuelve el problema anterior sin usar multiplicaciones (ni `pow()`). Para ello, debes emplear la relación  $(X+1)^3 = X^3 + 3X^2 + 3X + 1$ . Observa que de esta forma el cubo de cada número natural se puede calcular con sumas a partir del cubo y del cuadrado del número natural anterior. Para calcular los cuadrados sin multiplicaciones debes usar la relación análoga, es decir  $(X+1)^2 = X^2 + 2X + 1$ .

$$x_1 = 1 \quad x_1^2 = 1$$

$$x_1^3 = 1$$

$$x_2 = 2 \quad x_2^2 = x_1^2 + 2x_1 + 1 = 1 + 2 \cdot 1 + 1 = 4 \quad x_2^3 = x_1^3 + 3x_1^2 + 3x_1 + 1 = 1 + 3 \cdot 1 + 3 \cdot 1 + 1 = 8$$

$$x_3 = 3 \quad x_3^2 = x_2^2 + 2x_2 + 1 = 4 + 2 \cdot 2 + 1 = 9 \quad x_3^3 = x_2^3 + 3x_2^2 + 3x_2 + 1 = 8 + 3 \cdot 4 + 3 \cdot 2 + 1 = 27$$

$$x_4 = 4 \quad x_4^2 = x_3^2 + 2x_3 + 1 = 9 + 2 \cdot 3 + 1 = 16 \quad x_4^3 = x_3^3 + 3x_3^2 + 3x_3 + 1 = 27 + 3 \cdot 9 + 3 \cdot 3 + 1 = 64$$

...

