Facultad de Informática – Universidad Complutense
1º curso de los Grados

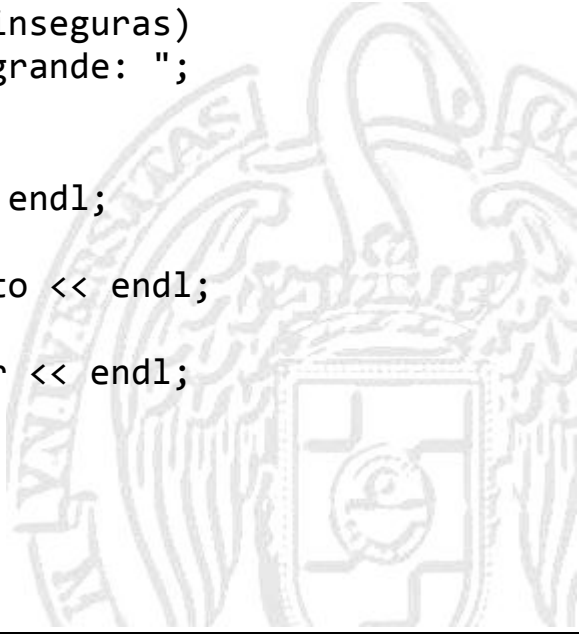
Fundamentos de la programación

Curso 2012–2013

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`, `suspenso`, `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.)

3. 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()`.

4. 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^3)	€/ m^3
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^3 que se encuentran en el intervalo. Así, si hemos consumido 750 m^3 deberíamos pagar: 100 * 0,15 + 400 * 0,20 + 250 * 0,35 = 182,50 €. Usa constantes.

5. 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 preguntará al usuario si quiere volver a calcular o terminar.

6. 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. 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.

7. 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!

8. 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).

9. 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.

10. Escribe un programa que lea del teclado diez números enteros y muestre en la pantalla el mayor de todos ellos.

11. Escribe un programa que a través de la consola consiga dos números enteros positivos y muestre su producto usando sólo sumas.

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.

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. Disponemos de cuatro variaciones del 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!*

15. ¿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 <iomanip>
using namespace std;
#include <fstream>

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;
}
```

16. Escribe un programa que genere un archivo `output.txt` en el que aparezcan *invertidos* los números enteros **positivos** 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	

17. 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.

18. Escribe un programa que lea una secuencia de caracteres introducida por el teclado (terminada en un punto) y cuente el número de caracteres anteriores a la aparición de la primera 'a'. El programa mostrará un mensaje si en la secuencia de entrada no existe ninguna 'a'.

19. 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, de forma iterativa, haga lo siguiente (en cada iteración): solicite un número entero y muestre en la pantalla su cuadrado calculado utilizando el algoritmo indicado. El programa deberá finalizar cuando se introduzca el valor `0`.

20. 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 (mayúsculas o minúsculas) y dígitos existentes entre la primera pareja de paréntesis. Si sólo aparece el paréntesis de apertura, es decir (, el recuento se realizará hasta el final de la secuencia. Hay que tener en cuenta que 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.

21. 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).

22. 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.

23. Escribe un programa que cuente el número de veces que aparece la secuencia `xy` en un archivo de texto `input26.txt` (terminado en *).

24. 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. Si el número es triangular, indicará también el entero hasta el que hay que sumar.

25. Implementa un programa que calcule el primer número natural cuyo cubo supera estrictamente otro entero N dado (N ≥ 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 $(X+1)^2 = X^2 + 2X + 1$.

$x_1 = \mathbf{1}$ $x_1^2 = \mathbf{1}$ $x_1^3 = \mathbf{1}$

$x_2 = \mathbf{2}$ $x_2^2 = x_1^2+2x_1+1 = 1+2\cdot1+1 = \mathbf{4}$ $x_2^3 = x_1^3+3x_1^2+3x_1+1 = 1+3\cdot1+3\cdot1+1 = \mathbf{8}$

$x_3 = \mathbf{3}$ $x_3^2 = x_2^2+2x_2+1 = 4+2\cdot2+1 = \mathbf{9}$ $x_3^3 = x_2^3+3x_2^2+3x_2+1 = 8+3\cdot4+3\cdot2+1 = \mathbf{27}$

$x_4 = \mathbf{4}$ $x_4^2 = x_3^2+2x_3+1 = 9+2\cdot3+1 = \mathbf{16}$ $x_4^3 = x_3^3+3x_3^2+3x_3+1 = 27+3\cdot9+3\cdot3+1 = \mathbf{64}$

...

27. Escribe un programa que genere los términos de una sucesión definida como:

$A_0 = 1$ $A_n = (-1)^n A_{n-1} + n$ para $n \geq 1$

$A_0 = 1$
$A_1 = (-1)^1 A_0 + 1 = -1 + 1 = 0$ $A_2 = (-1)^2 A_1 + 2 = 0 + 2 = 2$
$A_3 = (-1)^3 A_2 + 3 = -2 + 3 = 1$ $A_4 = (-1)^4 A_3 + 4 = 1 + 4 = 5$
$A_5 = (-1)^5 A_4 + 5 = -5 + 5 = 0$ $A_6 = (-1)^6 A_5 + 6 = 0 + 6 = 6$

El programa preguntará al usuario cuántos términos de la sucesión hay que generar.