Facultad de Informática – Universidad Complutense

Fundamentos de la programación

Curso 2013–2014

Hoja de ejercicios del Tema 7 - practicar

1. Escribe una función `lecturaConLimites()` que reciba como parámetros dos valores de tipo `int`. La función pedirá al usuario que introduzca un número entero tantas veces como sea necesario hasta que el número suministrado se encuentre dentro del intervalo determinado por los datos de entrada (*se irá dando pistas al usuario sobre si el número está por encima o por debajo del intervalo*). La función devolverá el último número leído.

Escribe un programa principal que use dicha función para pedir al usuario un número entre 10 y 20.

Ejemplo de ejecución:

```
Introduce un numero: 5
Demasiado bajo...
Introduce un numero: 100
Demasiado alto...
Introduce un numero: 15
El numero entre 10 y 20 es 15
```

2. Escribe un programa en C++ que calcule cuántos números *perfectos* hay en un archivo de texto `datos.txt` con enteros (cada número en una línea y terminado en 0 como centinela).

- Se dice que un número (entero positivo) *n* es *perfecto*, si la suma de los divisores de *n* entre 1 y *n*-1 es igual a *n*. Por ejemplo, 6 es un número perfecto, pues sus divisores, incluyendo el 1 pero no el propio 6, son 1, 2 y 3, y sumados dan 6.

Utiliza una función que determine si un número pasado como parámetro es perfecto o no.

Ejemplo de ejecución:

```
El archivo contiene 3 numero(s) perfecto(s)
```

3. Matemáticamente se puede demostrar que todos los números perfectos son, además, triangulares. Escribe un programa en C++ que compruebe empíricamente que esto es cierto para todos los números perfectos menores de 100.000. Usa las funciones necesarias para hacer el código lo más legible posible.

Ejemplo de ejecución:

```
6 es perfecto y triangular
28 es perfecto y triangular
496 es perfecto y triangular
8128 es perfecto y triangular
Queda demostrado!
```

4. Escribe una función `inverso()` que devuelva el resultado de invertir el entero positivo que reciba. Se entiende por invertir dar la vuelta a los dígitos del número (hallar su imagen especular); así, el inverso de 3952 es 2593.

Escribe una función `capicua()` que, haciendo uso de la función `inverso()`, devuelva un valor booleano que indique si el número entero positivo que recibe es o no capicúa.

Escribe un programa principal que solicite números enteros positivos e indique si son o no capicúas. El programa solicitará números hasta que se introduzca uno negativo.

Ejemplo de ejecución:

```
Introduce un entero positivo para ver si es capicua (<0 para terminar): 8585
8585 no es capicua
Introduce un entero positivo para ver si es capicua (<0 para terminar): 8558
8558 es capicua
Introduce un entero positivo para ver si es capicua (<0 para terminar): 33
33 es capicua
Introduce un entero positivo para ver si es capicua (<0 para terminar): 56
56 no es capicua
Introduce un entero positivo para ver si es capicua (<0 para terminar):
```

5. En combinatoria, el número de variaciones de x elementos de orden y, $V_{x,y}$, ($x > 0$, $0 < y \leq x$), el número de permutaciones de x elementos, P_x, ($x>0$) y el número de combinaciones de x elementos de orden y, $C_{x,y}$, ($x > 0$, $0 < y \leq x$) se obtienen mediante las siguientes fórmulas:

$$V_{x,y} = x \cdot (x-1) \cdot (x-2) \cdot \ldots \cdot (x-y+2) \cdot (x-y+1)$$

$$P_x = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (x-2) \cdot (x-1) \cdot x$$

$$C_{x,y} = V_{x,y} \,/\, P_y$$

Implementa:

- Una función `variaciones()` que reciba dos números enteros x e y y calcule y devuelva el entero que representa $V_{x,y}$.
- Otra función `permutaciones()` que reciba un número entero x y calcule y devuelva el entero que representa P_x.
- Y otra función `combinaciones()` que reciba dos números enteros x e y y calcule y devuelva (haciendo uso de las dos funciones anteriores) el entero que representa $C_{x,y}$.

Implementa un programa principal que solicite al usuario parejas de enteros positivos x e y mientras que no sean ambos 0 y que para cada pareja compruebe que cumplen las condiciones necesarias para poder calcular $V_{x,y}$, P_x y $C_{x,y}$ ($x > 0$, $0 < y \leq x$) y si es así los calculen, haciendo uso de los subprogramas anteriores, y los muestren en la pantalla.

6. Escribe un procedimiento en C++ que tenga como entrada un número entero positivo y que escriba en la pantalla una tabla como la siguiente, en la que se ha supuesto que el argumento utilizado para llamar al procedimiento es 4.

```
1    2    3    4    10
2    4    6    8    20
3    6    9   12    30
4    8   12   16    40
```

En la última columna se muestra el resultado de sumar todos los elementos anteriores de la fila correspondiente.

Escribe la función principal en la que se solicita al usuario el número de filas de la tabla y se llama al procedimiento creado con el número de filas introducido como argumento.

Ejemplo de ejecución:

```
Filas de la tabla: 5
   1    2    3    4    5   15
   2    4    6    8   10   30
   3    6    9   12   15   45
   4    8   12   16   20   60
   5   10   15   20   25   75
```

7. Escribe un procedimiento en C++ que encuentre y muestre todos los números de tres cifras en los que la suma de los cubos de sus dígitos sea igual al propio número. Ejemplo : $153 \equiv 1^3 + 5^3 + 3^3 = 1 + 125 + 27$

Escribe la función principal para probar que el procedimiento creado funciona correctamente.

Ejemplo de ejecución:

```
153 = 1^3 + 5^3 + 3^3
370 = 3^3 + 7^3 + 0^3
371 = 3^3 + 7^3 + 1^3
407 = 4^3 + 0^3 + 7^3
```

8. Escribe un programa que solicite al usuario la base y la altura de un triángulo, calcule su área y la muestre por pantalla.

Ejemplo de ejecución:

```
Introduzca los siguientes datos del triangulo
Base: 2
Altura: 5
El area del triangulo introducido es: 5
```

9. Escribe un programa en C++ que permita manejar una lista de hasta 100 cantidades reales positivas. A continuación permitirá al usuario realizar las siguientes acciones con la lista:

- ✓ Insertar una nueva cantidad al final de la lista
- ✓ Insertar una nueva cantidad al principio de la lista
- ✓ Mostrar la lista de cantidades (una en cada línea precedida de su posición)
- ✓ Guardar en el archivo `lista.txt` la lista

Cada opción se implementará con un subprograma y habrá un menú de opciones. Habrá una función que indique si la lista está llena.