

Prácticas – MTP I 2004/2005

*Dpto. Ingeniería en Informática
Prof. David Manuel Arenas González*

1. Escribir un programa que indique si un número leído por teclado es par o impar.
2. Escribir un programa para comprobar si una ecuación de la forma $Ax^2+Bx+C = 0$ tiene raíces reales, y a si es así, encuentre dichas raíces.
3. Escribir un programa, que dados dos enteros que representan una fecha (día, mes) e indicando si el año es bisiesto o no muestre por pantalla si la fecha es correcta o no.
4. Escribir un programa que calcule el factorial de un número.
5. Dado un entero $n \geq 0$, escribir un programa que calcule la suma de los n primeros múltiplos de 3.
6. Realizar el diseño y programar un algoritmo que calcule el siguiente sumatorio:

$$y=(1+1!)+(2+2!)+\dots+(N+N!)$$

siendo N un numero entero introducido por teclado y denotándose por $x!$ el factorial de x .

7. Escribir un programa para jugar a adivinar un número (entre 1 y 100) generado por dicho programa de manera aleatoria. Por cada intento de adivinar un número, el programa da una de las siguientes respuestas:

Fallo: el número a adivinar es menor que el tuyo

Fallo: el número a adivinar es mayor que el tuyo

Correcto:

El programa termina cuando el número es adivinado y muestra el número de intentos. Para generar el numero a adivinar se utilizar las funciones `int random(int x)` y `void randomize()` de la librería `<stdlib.h>`.

8. Escribir un programa que calcule el producto de dos enteros sin utilizar el operador `*`.

9. Implementa el algoritmo de ordenación por selección para ordenar un array de 10 números enteros que se piden por la consola.

10. Dados dos matrices de enteros de tamaño 4x4, implementa un programa que de como resultado el número de casillas iguales. Es decir el número de coincidencias de valores en la misma fila y columna.

11. Implementa un algoritmo que multiplique dos matrices 3x3 de números enteros y muestre el resultado por pantalla.

12. Diseñar un algoritmo que lea una cadena de caracteres en minúsculas de tamaño máximo 10 caracteres y muestre por pantalla la inversa en mayúsculas.

13. Leer una secuencia de nombres (máximo 10 caracteres por nombre) y almacenarlos en un array de tamaño máximo 20. No se deben almacenar nombre repetidos, cuando se inserte uno se mostrará por pantalla un error indicando que se ha intentado insertar un nombre que ya esta en la lista. Los nombres se introducen por teclado separados por INTRO. La secuencia termina con el valor '#'. Posteriormente se deben visualizar dichos nombres junto con su tamaño e indicando el orden en el que se introdujeron.

14. Ampliar el ejercicio anterior dando la posibilidad de borrar al final de la lista uno o varios nombres. Para borrarlo se debe poder elegir entre pasar la posición del nombre o la cadena de caracteres. Los nombres se borran de uno en uno con la posibilidad de ir borrando varios hasta que el usuario quiera. Se tiene que mostrar un mensaje de error si se quiere borrar por nombre una entrada que no esta en la tabla.

15. Implementa un algoritmo que tome caracteres por pantalla terminado en \n (INTRO) y muestre como resultado el numero total de vocales minúsculas y el número de vocales de cada tipo. Nota: Utilizar la función `char cin.get()`.

16. Implementar una estructura COMPLEJO que conste de dos miembros parteR (parte real) y parteI (parte imaginaria). Además impleméntense las operaciones de suma, resta, producto utilizando la estructura anteriormente creada. OJO los miembros de los complejos son números reales.

17. Implementar el ejercicio 16 utilizando punteros a las estructuras COMPLEJO. OJO es necesario reservar memoria para las estructuras con `new` y es obligatorio liberarla cuando no se necesite el espacio con `delete`.

18. Implementar un algoritmo que copie el contenido de un archivo de texto (*origen.txt*) que esta ubicado en el directorio del archivo ejecutable creado por el proyecto (carpeta \Windows\Debug_Build), a otro archivo (*destino.txt*) que esta en la misma ruta. El programa debe tener dos opciones:

1. Copiar el archivo borrando todo el contenido que tiene destino, es decir hacer una copia exacta.
2. Copiar el archivo origen añadiendo el contenido al final del archivo destino.

NOTAS:

- El archivo como mucho puede tener líneas de longitud 20 caracteres.
- No olvidar cerrar los flujos abiertos cuando no vayan a ser utilizados mas. OJO abrir y cerrar flujos innecesariamente son operaciones costosas en tiempo, por tanto hay que utilizarlas las veces estrictamente necesarias.
- Controlar el éxito de la apertura de un archivo tanto para lectura como para escritura y mostrar un error en caso de fallo.
- En este programa puede ser de gran ayuda el uso del método *stream.open*("ruta de localización",[opciones]). Stream es una variable del tipo *ofstream* o *ifstream*. Los [] indican que puede o no haber opciones dependiendo del tipo de archivo y la forma en que se quiere abrir.