

LENGUAJES DE PROGRAMACIÓN

INSTRUCCIONES

Por favor, entregue esta primera hoja de enunciado junto con el examen.

Dispone de 2 horas para realizar el examen.

MATERIAL PERMITIDO: Ninguno.

Pregunta 1 (3 puntos)

Indique la veracidad o falsedad de cada una de las afirmaciones siguientes, explicando detalladamente en cada caso el motivo de su respuesta.

- A. (0.5 puntos) La máquina de von Neumann tiene dos memorias separadas, una dedicada a almacenar datos y otra dedicada a almacenar instrucciones.
- B. (0.5 puntos) La parte del programa en que una variable es visible puede ser más reducida que el ámbito de dicha variable.
- C. (0.5 puntos) Un array es un tipo de dato primitivo.
- D. (0.5 puntos) Un tipo estructura es una colección de variables del mismo tipo referenciadas bajo el mismo nombre.
- E. (0.5 puntos) En el lenguaje C++, la sentencia

$$x = i++;$$

es equivalente a las dos sentencias siguientes:

$$x = i;$$
$$i = i + 1;$$

- F. (0.5 puntos) El lenguaje C fue el primer lenguaje que introdujo la sentencia de selección con dos alternativas (**if-then-else**).

Pregunta 2 (1 punto)

Escriba la salida por consola producida al ejecutar el programa en C++ escrito a continuación.

```
#include <iostream>
#include <iomanip>

double x, y;

int main () {
    x = 7.06178;
    y = 5.62456;
    double *p;
    double a[2] = { 2.123, 6.551 };
    {
        double x = y;
        std::cout << std::scientific << std::setprecision(3)
                  << x << std::endl;
        std::cout << ::x <<std::endl;
        p = &a[0];
    }
    p[0] = p[0] + 1.0;
    std::cout << a[0] << std::endl;
    p = p + 1;
    std::cout << p[0] << std::endl;
    std::cout << *(a+1) << std::endl;
    return 0;
}
```

Pregunta 3 (1 punto)

Escriba la salida por consola producida al ejecutar el programa en C++ escrito a continuación.

```
#include <iostream>
#include <stdexcept>
#include <sstream>

int division (int i1, int i2) throw (std::invalid_argument) {
    if (!i2) {
        std::stringstream ss;
        ss << i1 << "/" << i2 << " Error";
        throw std::invalid_argument (ss.str());
    }
    return i1/i2;
}

int main() {
    for (int den=4; den>=0; den--)
        try {
            std::cout << "10/" << den << " = "
                << division(10,den) << std::endl;
        } catch ( std::invalid_argument exc ) {
            std::cout << exc.what() << std::endl;
        }
    return 0;
}
```

Pregunta 4 (2 puntos)

Escriba un programa en C++ que encuentre y muestre en la consola todos los números naturales que satisfagan la condición siguiente:

el resultado de concatenar el cuadrado del número con el cubo del número contiene todos los dígitos desde 0 hasta 9 una y solo una vez.

Veamos dos ejemplos.

Ejemplo 1 Comprobemos si $N = 19$ satisface la condición. Para ello, primeramente calculamos el cuadrado y el cubo de N :

$$19^2 = 361$$

$$19^3 = 6859$$

Seguidamente concatenamos ambos, obteniendo:

3616859

Finalmente comprobamos si el número 3616859 contiene todos los dígitos desde 0 hasta 9 una y solo una vez. La respuesta es no, ya que el dígito 6 está repetido y faltan los dígitos 0, 2, 4 y 7. El número 19 no satisface la condición, por lo que el programa no lo muestra en la consola.

Ejemplo 2 Comprobemos si $N = 69$ satisface la condición. Para ello, primeramente calculamos el cuadrado y el cubo de N :

$$69^2 = 4761$$

$$69^3 = 328509$$

Seguidamente concatenamos ambos, obteniendo:

4761328509

Finalmente comprobamos si el número 4761328509 contiene todos los dígitos desde 0 hasta 9 una y solo una vez. La respuesta es sí. El número 69 sí satisface la condición, por lo que el programa debería mostrar el número 69 en la consola.

Pregunta 5 (3 puntos)

Escriba en C++ una función que calcule el número de elementos de un vector que se encuentran fuera de un cierto intervalo, y un programa que lea números reales de un fichero de texto y muestre en la consola cuántos de estos datos están fuera en un cierto intervalo, valiéndose para ello de la función anteriormente definida. A continuación se explica todo ello con detalle.

5.a (1.5 puntos) Defina una función en C++ que acepte tres parámetros: una referencia a un vector de elementos **double** y dos variables **double**. Se muestra a continuación el prototipo de la función, la cual puede lanzar fuera de sí una excepción:

```
int numDatosFueraIntervalo(std::vector<double>&, double, double)
    throw (std::invalid_argument);
```

La función, que devuelve un valor **int**, debe realizar las acciones siguientes:

1. *Comprobar que el intervalo esté bien definido.* Los dos argumentos **double** son el extremo inferior y superior del intervalo, respectivamente. Debe satisfacerse que el extremo inferior sea menor o igual al extremo superior. Si no se satisface esta condición, la función debe lanzar una excepción.
2. *Contar los elementos del vector que están fuera del intervalo.* Para ello, debe examinar uno a uno los elementos del vector referenciado. Un elemento está fuera del intervalo si su valor es menor que el extremo inferior del intervalo o mayor que el extremo superior del intervalo.
3. *Devolver el resultado de la cuenta anterior.*

5.b (1.5 puntos) Escriba un programa en C++ que realice las acciones siguientes.

1. *Apertura para lectura de un fichero de texto.* El nombre del fichero de texto en el cual se encuentran los datos debe almacenarse en una constante global del tipo `std::string`, llamada *nombreFich*, cuyo valor debe ser "datos.txt". El programa debe abrir el fichero de texto para lectura. Si se produce error, debe mostrar un mensaje en la consola indicándolo y terminar.
2. *Declarar un vector de elementos **double** llamado `vectorDatos`.*
3. *Lectura del fichero de texto y escritura en el vector.* El fichero de texto contiene datos, que son números reales. El programa debe ir leyéndolos y añadiéndolos al final del vector `vectorDatos`.
4. *Entrada por consola de los extremos del intervalo.* El programa debe solicitar al usuario que éste introduzca por consola los extremos inferior y superior del intervalo. Ambos valores deben almacenarse en variables **double** llamadas `intervL` y `intervH`, respectivamente.

5. El programa debe invocar la función anteriormente definida, pasando como argumento la referencia al vector y los valores de los extremos del intervalo, y debe escribir en la consola el valor devuelto por la función.

El programa debe estar preparado para capturar y tratar la excepción que puede ser lanzada por la función. Si se captura una excepción, el programa debe mostrar un mensaje en la consola indicándolo.

6. Terminar.