

LENGUAJES DE PROGRAMACIÓN

Solución al examen de Junio 2013, Primera Semana

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) En FORTRAN I es necesario declarar el tipo de las variables.
- B. (0.5 puntos) Una estructura es una colección de una o más variables del mismo tipo agrupadas bajo un nombre común.
- C. (0.5 puntos) Las variables locales existen hasta que el programa termina o hasta que son eliminadas.
- D. (0.5 puntos) El cuerpo del bucle **for** en Java se ejecuta siempre al menos una vez.
- E. (0.5 puntos) Una cola es un tipo especial de lista, en la cual los elementos se insertan por uno de los extremos de la lista y se eliminan por ese mismo extremo.
- F. (0.5 puntos) Cualquier nodo de un árbol posee un único nodo padre.

Solución a la Pregunta 1

- A Falsa Véase la página 43 del texto base.
- B Falsa Véase la página 119 del texto base.
- C Falsa Véase la página 114 del texto base.
- D Falsa Véase la página 273 del texto base.
- E Falsa Véase la página 432 del texto base.
- F Falsa Véase la página 436 del texto base.

PREGUNTA 2 (1 punto)

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

```
#include <iostream>
int x1, x2, *p1, *p2;
int main () {
    x1 = 10;
    p1 = &x1;
    std::cout << x1 << std::endl;
    std::cout << *p1 << std::endl;
    {
        int x1 = 4;
        p2 = &x1;
        ::x1 = 5;
        std::cout << *p1 << std::endl;
        std::cout << *p2 << std::endl;
        x2 = x1;
        std::cout << x2 << std::endl;
        std::cout << ::x1 << std::endl;
    }
    std::cout << x1 << std::endl;
    return 0;
}
```

Solución a la Pregunta 2

La salida por consola producida al ejecutar el programa es:

```
10
10
5
4
4
5
5
```

PREGUNTA 3 (1 punto)

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

```
#include <iostream>
#include <vector>
int main()
{
    std::vector<double> v(4,5);
    std::vector<double>::iterator iter;
    iter = v.begin()+1;
    *iter = 1;
    *(iter+1) = 2;
    v.at(0) = 10;
    *(iter+2) = -1*(*iter);
    v.at(1) = 4;
    std::cout << *iter << ", " <<
        *(iter+1) << ", " << *(iter+2) << std::endl;
    std::cout << "v = ( " << v.at(0) << ", " <<
        v.at(1) << ", " << v.at(2) << ", " <<
        v.at(3) << ")" << std::endl;
    return 0;
}
```

Solución a la Pregunta 3

La salida por consola producida al ejecutar el programa es:

```
4, 2, -1
v = ( 10, 4, 2, -1)
```

PREGUNTA 4 (2.5 puntos)

El algoritmo de Euclides permite calcular el máximo común divisor (*mcd*) de dos números enteros. Este algoritmo se describe como:

$$mcd(x, y) = \begin{cases} x & \text{si } y = 0 \\ mcd(y, x \% y) & \text{si } (y > 0) \wedge (x \geq y) \end{cases} \quad (1.1)$$

donde $x \% y$ representa el resto de la división entera de x por y :

$$x \% y = x - (\text{int}(x/y) \cdot y) \quad (1.2)$$

4.a (0.5 puntos) Programe en C++ una función que admita como parámetros dos números enteros, x e y , y devuelva el resto de la división entera de ambos, $x \% y$. Para ello, la función debe aplicar la Ecuación (1.2). El nombre de la función debe ser *resto*.

4.b (1 punto) Programe en C++ una función llamada *mcd* que admita como parámetros dos números enteros, x e y , y que devuelva el máximo común divisor. Para ello, debe aplicarse el algoritmo descrito en (1.1). El cálculo de $x \% y$ debe realizarse invocando la función *resto*, que se ha programado anteriormente.

4.c (1 punto) Escriba un programa en C++ que realice las acciones siguientes:

1. Solicite desde la consola la entrada de dos números enteros, x e y , indicando que x debe ser mayor o igual que y . Si los números introducidos por el usuario no satisfacen esta condición, entonces terminar.
2. Calcule el máximo común divisor de x e y , invocando para ello la función *mcd* que ha programado anteriormente.
3. Muestre en la consola el resultado.

Solución a la Pregunta 4

```
#include <iostream>

int resto(int x, int y)
{
    int ce = x/y;
    return x-(ce*y);
}

int mcd(int a, int b) {
    // Se asume que a >= b >= 0
    if (b==0) return a;
    return mcd(b, resto(a,b));
}

int main()
{
    int a, b;
    std::cout << "MCD"<<std::endl;
    std::cout << "a: ";
    std::cin >> a;
    std::cout << "b: ";
    std::cin >> b;
    if (a<b || b<0)
        std::cout << "ERROR: debe satisfacerse a >= b >= 0"
        << std::endl;
    else
        std::cout << "El MCD es: "<<mcd(a,b)<< std::endl;
    return 0;
}
```

PREGUNTA 5 (2.5 puntos)

Escriba en C++ un programa que lea un fichero de texto que contiene números reales y muestre en la consola el valor de la mediana. El programa debe realizar las acciones descritas a continuación.

1. *Apertura para lectura del fichero de texto.*

El nombre del fichero de texto en el cual se encuentran los números 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. *Lectura del fichero de texto.*

El fichero de texto contiene números reales. El programa debe ir leyendo los datos del fichero y almacenándolos en un vector de elementos **double** llamado `datos`.

3. *Comprobar que $N > 1$.*

Sea N el número de datos leídos del fichero. Si no se satisface $N > 1$, el programa debe mostrar un mensaje en la consola indicándolo y terminar.

4. *Ordenación creciente de los números.*

El programa debe ordenar crecientemente los elementos del vector `datos`, empleando para ello el método de la burbuja.

5. *Cálculo de la mediana.*

Sean $x_1, \dots, x_N$ los datos ordenados crecientemente. La mediana se calcula de la forma siguiente:

- Si N es impar, la mediana es el número que ocupa la posición central.
- Si N es par, la mediana es la media aritmética de los dos valores centrales.

6. *Mostrar en la consola el valor de la mediana.*

7. *Terminar.*

Solución a la Pregunta 5

```

#include <iostream>
#include <fstream>
#include <vector>
#include <string>

const std::string nombreFich = "datos.txt";

int main() {
    std::ifstream file_in(nombreFich.c_str(),std::ios::in);
    if (!file_in) {
        std::cout << "No se puede abrir el fichero"<< std::endl;
        return 1;
    }
    // Lectura del fichero
    std::vector<double> datos;
    while (!file_in.eof()) {
        double d;
        file_in >> d;
        if (!file_in.eof())
            datos.push_back(d);
    }
    // Comprobación de que N>1
    int N = datos.size();
    if (N<2) {
        std::cout << "ERROR: Datos insuficientes"<< std::endl;
        return 0;
    }
    // Ordenación creciente. Método de la burbuja
    for (int i=0; i<N; i++) {
        for (int j=N-1; j>=i+1; j--) {
            if (datos[j]<datos[j-1]){
                double aux = datos[j-1];
                datos[j-1] = datos[j];
                datos[j] = aux;
            }
        }
    }
    // Cálculo de la mediana
    double mediana;
    if (N % 2 ) {
        // N es impar
        mediana = datos[(N-1)/2];
    } else {
        // N es par
        mediana = 0.5 * ( datos[N/2-1] + datos[N/2] );
    }
    std::cout << "Mediana: " << mediana << std::endl;
    return 0;
}

```