LENGUAJES DE PROGRAMACIÓN

Solución al examen de Junio 2022, 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) La máquina de Von Neumann no soporta instrucciones como `goto` que permiten el control del flujo de programa.

B. (0.5 puntos) El lenguaje ensamblador y el lenguaje máquina son específicos para cada CPU. En consecuencia, el código de los programas escritos en estos lenguajes no es portable de una CPU a otra.

C. (0.5 puntos) Dado el tipo estructura `s1`, declarado en C++ de la siguiente manera:

```
struct s1 {int i; char nombre;};
```

Una variable de este tipo contiene dos variables, una de tipo **int** y otra de tipo **char**.

D. (0.5 puntos) Los operandos de las expresiones relacionales sólo pueden ser de tipo Booleano.

E. (0.5 puntos) El flujo del tipo `std::fstream` en C++ se emplea para leer y escribir en fichero.

F. (0.5 puntos) El error de "puntero a variable eliminada" se puede producir en punteros que contienen la dirección de una variable local.

Solución a la Pregunta 1

A	Falso	Véase la página 31 del texto base.
B	Verdadero	Véase la página 33 del texto base.
C	Verdadero	Véase la página 112 del texto base.
D	Falso	Véase la página 164 del texto base.
E	Verdadero	Véase la página 325 del texto base.
F	Verdadero	Véase la página 119 del texto base.

PREGUNTA 2 (1 punto)

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

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

int funciond(int n) throw (std::invalid_argument) {
    int i = 1;
    int count = 0;
    if (n < 0) throw std::invalid_argument("ERROR VALOR");
    while (i < n) {
        if (n % i == 0)
            count = count + 1;
        i = i + 1;
    }
    return count;
}

int main() {
    for (int i = 4; i > -2; i--) {
        try {
            std::cout << i << " ";
            std::cout << funciond(i) << std::endl;

        }
        catch (const std::invalid_argument& exc) {
            std::cout << exc.what() << std::endl;
        }
    }
    return 0;
}
```

Solución a la Pregunta 2

La salida por consola producida al ejecutar el programa es:

```
4 2
3 1
2 1
1 0
0 0
-1 ERROR VALOR
```

PREGUNTA 3 (1 punto)

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

```
#include <iostream>

int x1 = 2;
int x2 = 6;
int* p1, * p2;

int main() {
    x1 = 4;
    p1 = &x1;
    int v[] = { x1, x2, x1 * 3, 6 };
    p2 = &v[0];
    std::cout << x1   << std::endl;
    std::cout << ::x1 << std::endl;
    std::cout << *p1  << std::endl;
    std::cout << *(p2 + 1) << std::endl;
    std::cout << *(p2 + 2) << std::endl;
    {
        int x1 = 8;
        p2 = &x1;
        std::cout << *p1 << std::endl;
        std::cout << *p2 << std::endl;
        x2 = x1;
        std::cout << x2   << std::endl;
        std::cout << ::x1 << std::endl;
        x1 = 3;
    }
    std::cout << x1 << std::endl;
    return 0;
}
```

Solución a la Pregunta 3

La salida por consola producida al ejecutar el programa es:

```
4
4
4
6
12
4
8
8
4
4
```

PREGUNTA 4 (2 puntos)

Supongamos que una imagen de vídeo está descrita mediante una matriz binaria. Matrices binarias son aquellas cuyos elementos pueden valer 0 ó 1.

Se desea obtener el fondo fijo común de varias imágenes de vídeo en las que hay objetos en movimiento (por ejemplo, las imágenes captadas por una cámara fija de vigilancia). Para ello, se va a emplear el siguiente cálculo, que permite obtener la matriz binaria (**Ifondo**) que describe el fondo fijo común a tres imágenes binarias (**I1**, **I2** e **I3**):

$$\mathbf{Ifondo}_{i,j} = (\mathbf{I3}_{i,j} \text{ AND } (\mathbf{I1}_{i,j} \text{ XOR } \mathbf{I2}_{i,j})) \text{ OR } (\mathbf{I1}_{i,j} \text{ AND } \mathbf{I2}_{i,j})$$

donde i y j son los índices de fila y columna de los elementos de las matrices. Cada elemento de la matriz que describe el fondo fijo se calcula operando los elementos de las tres matrices imagen que se encuentran en su misma fila (i) y columna (j). El operador OR exclusivo (XOR) devuelve 1 si los dos operandos son diferentes y devuelve 0 si son iguales.

Ejemplo: aplicación a tres imágenes representadas mediante matrices binarias de tamaño 3×3:

I1	**I2**	**I3**		**Ifondo**
1 0 1	0 0 1	0 0 1		0 0 1
0 1 1	1 1 0	0 1 0	$\longrightarrow$	0 1 0
0 1 1	0 0 1	1 1 0		0 1 1

Escriba un programa en C++ que realice las tareas siguientes:

1. Mediante la escritura de un mensaje en la consola, solicitar al usuario que éste introduzca por consola los 9 elementos de cada una de las tres matrices **I1**, **I2** e **I3**, cada una de las cuales tiene tamaño 3×3. Almacenar los valores leídos en tres arrays bidimensionales de elementos Booleanos. Los arrays deben llamarse `I1`, `I2` e `I3`.

2. Realizando el cálculo descrito anteriormente, calcular la matriz binaria que describe la imagen de fondo y escribirla en la consola.

3. Terminar.

Solución a la Pregunta 4

```
#include <iostream>

const int N = 3; // Tamaño de la matriz imagen: NxN

int main() {
  // Entrada por consola de I1, I2 e I3
  bool I1[N][N], I2[N][N], I3[N][N];
  std::cout << "Introduzca la matriz I1:\n";
  for (int i=0; i<N; i++)
    for (int j=0; j<N; j++)
      std::cin >> I1[i][j];
  std::cout << "Introduzca la matriz I2:\n";
  for (int i=0; i<N; i++)
    for (int j=0; j<N; j++)
      std::cin >> I2[i][j];
  std::cout << "Introduzca la matriz I3:\n";
  for (int i=0; i<N; i++)
    for (int j=0; j<N; j++)
      std::cin >> I3[i][j];
  // Cálculo y escritura en consola de Ifondo
  for (int i=0; i<N; i++) {
    for (int j=0; j<N; j++) {
      bool Ifondo_i_j = ( I3[i][j] && ( I1[i][j] != I2[i][j] ) ) ||
                        ( I1[i][j] && I2[i][j] );
      std::cout << Ifondo_i_j << "  ";
    }
    std::cout << "\n";
  }
    return 0;
}
```

PREGUNTA 5 (3 puntos)

Un fichero de texto llamado *circulos.txt* almacena las coordenadas cartesianas (X, Y) del centro, la longitud del radio y el color de cada uno de los círculos que forman parte de un conjunto de círculos. Cada círculo está descrito en una fila del fichero. En las dos primeras columnas están escritas las coordenadas X e Y del centro del círculo. En la tercera columna está escrita la longitud del radio del círculo. La cuarta columna contiene una palabra que describe el color del círculo. Por ejemplo, la línea del fichero mostrada a continuación describe un círculo que está centrado en el punto del plano con coordenadas $(1.0, 2.5)$, tiene radio 0.3 y color azul.

```
1.0  2.5  0.3  azul
```

5.1 (1 punto) Escriba una función en C++ llamada `getCirculos` que tiene el prototipo siguiente:

```
std::list<sCirc> getCirculos( std::string color,
                              std::string nombreFichero )
                        throw (std::invalid_argument);
```

La función devuelve una lista de elementos del tipo estructura `sCirc`, definido de la forma siguiente:

```
struct sCirc {
    double centroX, centroY, radio;
};
```

Los elementos de la lista contienen las coordenadas X e Y del centro, y la longitud del radio de cada uno de los círculos cuyo color es el especificado mediante el primer argumento de la función, de los escritos en el fichero cuyo nombre es indicado mediante el segundo argumento de la función.

Si se produce error al abrir el fichero de texto para lectura, la función debe lanzar una excepción.

5.2 (1 punto) Escriba una función en C++ llamada `checkCirculos` que tiene el prototipo siguiente:

```
std::list<bool> checkCirculos(
                    std::list<sCirc> &lCirculos,
                    double radioUmbral );
```

El primer argumento es una referencia a una lista de elementos del tipo estructura `sCirc`. La función devuelve una lista de elementos Booleanos con el mismo número de elementos que la lista argumento. El elemento de la lista devuelta tiene el valor *true* si y sólo si el radio del círculo representado por el correspondiente elemento de la lista argumento tiene un valor menor que el segundo argumento de la función: `radioUmbral`.

5.3 (1 punto) Escriba un programa en C++ que declare una constante global de tipo **double** llamada `RADIO` y le asigne el valor 10.5.

El programa debe, invocando las dos funciones definidas anteriormente, obtener todos los círculos del fichero *circulos.txt* que son de color rojo y además tienen un radio menor que la constante `RADIO`. A continuación, el programa debe escribir en la consola las coordenadas X e Y del centro y el valor del radio de cada uno de estos círculos, todo ello en formato científico con cuatro dígitos decimales.

Si se produce error al abrir el fichero para lectura, la función `getCirculos` lanza una excepción. El programa principal debe capturarla, mostrar un mensaje en la consola indicando el error y terminar.

Solución a la Pregunta 5

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include <list>
#include <string>
#include <stdexcept>

const double RADIO = 10.5;
const std::string nombreFich = "circulos.txt";

struct sCirc {
  double centroX, centroY, radio;
};


```

```
16
17 std::list<sCirc> getCirculos(std::string color,
18                               std::string nombreFichero)
19                       throw (std::invalid_argument) {
20   std::ifstream inFich(nombreFichero.c_str(), std::ios::in);
21   if (!inFich)
22     throw std::invalid_argument("Error al abrir el fichero");
23   std::list<sCirc> lCirc;
24   sCirc circ;
25   std::string colorCirc;
26   while (inFich >> circ.centroX >> circ.centroY >> circ.radio >>
      colorCirc)
27     if (color == colorCirc)
28       lCirc.push_back(circ);
29   inFich.close();
30   return lCirc;
31 }
32
33 std::list<bool> checkCirculos(std::list<sCirc> &lCirculos,
34                               double radioUmbral) {
35   std::list<bool> lRes;
36   std::list<sCirc>::iterator p = lCirculos.begin();
37   while (p != lCirculos.end()) {
38     lRes.push_back(p->radio < radioUmbral);
39     p++;
40   }
41   return lRes;
42 }
43
44 int main() {
45   std::list<sCirc> lCircRojo;
46   try {
47     lCircRojo = getCirculos("rojo", nombreFich);
48   } catch (std::invalid_argument &exc) {
49     std::cout << exc.what() << std::endl;
50   }
51   std::list<bool> esRadioMenorRADIO = checkCirculos(lCircRojo, RADIO);
52   std::list<sCirc>::iterator pCirc = lCircRojo.begin();
53   std::list<bool>::iterator pCondRadio = esRadioMenorRADIO.begin();
54   while (pCirc != lCircRojo.end()) {
55     if (*pCondRadio)
56       std::cout << std::scientific << std::setprecision(4)
57           << pCirc->centroX << "\t" << pCirc->centroY << "\t"
58           << pCirc->radio << std::endl;
59     pCirc++;
60     pCondRadio++;
61   }
62   return 0;
63 }
```