LENGUAJES DE PROGRAMACIÓN

Solución al examen de Junio 2023, Segunda 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) LISP es un lenguaje imperativo.

B. (0.5 puntos) En C++ se realizar la coerción entre los tipos char e int.

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

```
struct s {int i; double valor;};
```

Una variable de este tipo contiene dos variables: una de tipo **int** y otra de tipo **double**.

D. (0.5 puntos) Se dice que una función tiene recursividad de cola si la función devuelve, o bien un valor que es calculado sin necesidad de recursividad, o bien el resultado de una llamada recursiva.

E. (0.5 puntos) El tipo de dato `std::list` en C++ es una implementación de una lista doblemente enlazada.

F. (0.5 puntos) El algoritmo de la STL de C++

```
remove_copy(pBegin, pEnd, pResultBegin, val)
```

copia la secuencia origen en la secuencia destino, eliminando de la copia los elementos cuyo valor sea menor que `val`.

Solución a la Pregunta 1

A	Falso	Véase la página 58 del texto base.
B	Verdadero	Véase la página 174 del texto base.
C	Verdadero	Véase la página 232 del texto base.
D	Verdadero	Véase la página 373 del texto base.
E	Verdadero	Véase la página 489 del texto base.
F	Falso	Véase la página 556 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 main() {
   int *p1, *p2, *p3, i, m;
   int a[] = { 4, -2, 6, 7 };
   i = -10;
   m = 20;
   p1 = &i;
   p2 = a;
   p2++;
   *p2 = *p1;
   p1 = (p2 + 1);
   std::cout << *p1 << " " << *p2 << " ";
   p1 = new int;
   *p1= m;
   p3 = p2;
   std::cout << a[0] << " " << a[1] << " " << a[2] << " ";
   std::cout << *p1  << " " << *p2  << " " << *p3  << " ";
   return 0;
}
```

Solución a la Pregunta 2

La salida por consola producida al ejecutar el programa es:

```
6 -10 4 -10 6 20 -10 -10
```

PREGUNTA 3 (1 punto)

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

```
#include <iostream>
#include <vector>
#include <algorithm>

int main() {
   std::vector<int> vec(2,5);
   for (int i = 5; i < 11; i++)
      vec.push_back(i);
   for (unsigned int i = 0; i < vec.size(); i++)
      std::cout << vec.at(i) << " ";
   std::cout << std::endl;

   int n1 = std::count_if(vec.begin(), vec.end(),
      [](int n) -> bool { return !(n % 5); });
   std::cout << "n1 = " << n1 << std::endl;
   return 0;
}
```

Solución a la Pregunta 3

La salida por consola producida al ejecutar el programa es:

```
5 5 5 6 7 8 9 10
n1 = 4
```

PREGUNTA 4 (2.5 puntos)

Se considera que los puntos que pertenecen al conjunto de Mandelbrot son los números complejos c que hacen que la sucesión de números complejos:

$$\begin{aligned} z_0 &= 0 \\ z_{n+1} &= z_n^2 + c \qquad \text{con } n = 0, 1, 2, \ldots \end{aligned}$$

cumpla que $|z_n| < 2$ para todo n. Numéricamente, se puede considerar que se cumple la condición de acotación si esta condición se cumple para los N primeros elementos de la sucesión, esto es, $z_0, \ldots, z_{N-1}$.

En un fichero de texto llamado *puntos.txt* están escritos números complejos. En cada fila del fichero está escrito un número: en la primera columna su parte real y en la segunda columna su parte imaginaria. Escriba un programa en C++ que realice las acciones siguientes:

1. Declarar una constante global entera llamada `N` y asignarle el valor 1000.
2. Abrir para lectura el fichero de texto *puntos.txt* y leer su contenido, almacenándolo en una lista de números complejos llamada `lc`. Si se produce error al abrir el fichero, mostrar un mensaje en la consola indicándolo y terminar.
3. Comprobar si cada elemento de la lista `lc` pertenece al conjunto de Mandelbrot, empleando la constante global `N` en la condición de acotación. Eliminar de la lista los elementos que no pertenecen al conjunto de Mandelbrot.

   A fin de comprobar si un número complejo pertenece al conjunto de Mandelbrot, programe usted una función en C++ tal que dado un número complejo devuelva *true* si pertenece al conjunto de Mandelbrot y *false* en caso contrario. La función debe emplear la constante global `N` en la condición de acotación. Invoque esta función desde el programa. El prototipo de la función es:

   ```
   bool mandelbrot( std::complex<double> c );
   ```

4. Escribir en la consola el contenido de la lista `lc`, la cual debe contener los números del fichero que pertenecen al conjunto de Mandelbrot.
5. Terminar.

Solución a la Pregunta 4

```cpp
#include <iostream>
#include <fstream>
#include <cmath>
#include <complex>
#include <list>

const int N = 1000;
const char nombreFich[] = "puntos.txt";

bool mandelbrot( std::complex<double> c ) {
   std::complex<double> z(0,0);
   for (int i=1; i<N; i++) {
      z = z*z + c;
      if (std::abs(z) >= 2)
         return false;
   }
   return true;
}

int main() {
   // Apertura y lectura del fichero
   std::ifstream inFich(nombreFich, std::ios::in);
   if (!inFich) {
      std::cerr << "Error al abrir el fichero\n";
      return 0;
   }
   double c_real, c_imag;
   std::list< std::complex<double> > lc;
   while ( inFich >> c_real >> c_imag ) {
      std::complex<double> z(c_real, c_imag);
      lc.push_back(z);
   }
   inFich.close();
   // Eliminar elementos
   std::list< std::complex<double> >::iterator p = lc.begin();
   while (p != lc.end()) {
      if ( mandelbrot(*p) )
         p++;
      else
         p = lc.erase(p);
   }
   // Escritura contenido lista en consola
   std::cout << "Pertenecen al conjunto de Mandelbrot:\n";
   p = lc.begin();
   while (p != lc.end()) {
      std::cout << *p << std::endl;
      p++;
   }
   return 0;
}
```

PREGUNTA 5 (2.5 puntos)

La técnica denominada de las diferencias centrales permite estimar el valor de la derivada de una función en un punto, tal como se explica a continuación.

Consideremos una función $f(x)$, que puede ser desarrollada en serie de Taylor en un entorno E de un punto x. La técnica de las diferencias centrales consiste en aproximar la derivada de la función $f(x)$ en x, mediante la expresión siguiente:

$$\frac{f(x+h) - f(x-h)}{2 \cdot h}$$

con $h > 0$, donde el intervalo $[x-h, x+h]$ está incluido en E.

5.1 (1 punto) Escriba una función en C++ con el prototipo siguiente:

```
double derf( double &x, double (*pf)(double) );
```

que devuelva la derivada de la función especificada mediante el segundo argumento, en el punto especificado mediante el primer argumento, calculada empleando la técnica de las diferencias centrales. El valor de h es una constante declarada en la función: $h = 10^{-4}$.

5.2 (0.5 puntos) Escriba una función en C++ con el prototipo siguiente:

```
double f(double x);
```

que devuelva el resultado de evaluar la expresión

$$\cos^2(2 \cdot x - 0.25)$$

donde x es el argumento de la función.

5.3 (1 punto) Escriba un programa en C++ que realice las acciones siguientes:

1. Declarar una variable llamada `vx` del tipo vector de **double** y almacenar en ella la secuencia de valores siguiente:

$$3.2 + \text{sen}\left(\frac{\pi}{180} \cdot n + 0.03\right) \qquad \text{con} \quad n = 0, 1, \ldots, N$$

donde N es una constante global de tipo entero que vale 90.

2. Aplicando el método de las diferencias centrales con el valor de h anteriormente indicado, estimar el valor de la derivada de

$$f(x) = \cos^2(2 \cdot x - 0.25)$$

en los puntos x almacenados en el vector `vx`. Para ello, debe hacerse uso de las funciones `derf` y `f` definidas anteriormente. Los valores de la derivada deben almacenarse en un vector llamado `vDerf`.

3. Escribir en la consola los puntos x almacenados en el vector `vx`, en formato fijo con 10 dígitos detrás del punto decimal, y los correspondientes valores de la derivada de la función almacenados en el vector `vDerf`, en formato científico con 6 dígitos detrás del punto decimal.

   Cada punto y la estimación de la derivada en ese punto deben escribirse en una línea, separados por un tabulador.

4. Terminar.

Solución a la Pregunta 5

```cpp
#include <iostream>
#include <iomanip>
#include <vector>
#include <cmath>

const double PI = std::acos(-1.);
const int N = 90;

double derf(double &x, double (*pf)(double)) {
   const double h = 1e-4;
   return ( (*pf)(x + h) - (*pf)(x - h) ) / (2*h);
}

double f(double x) {
   return std::pow( std::cos(2*x - 0.25), 2 );
}

int main() {
   // Declaracion vector vx y asignacion de valores
   std::vector<double> vx;
   for (int n = 0; n <= N; n++)
      vx.push_back( 3.2 + std::sin( PI/180*n + 0.03 ) );
   // Estimacion mediante diferencias centrales
   std::vector<double> vDerf(vx.size());
   for (unsigned int i = 0; i < vx.size(); i++)
      vDerf[i] = derf( vx[i], f );
   // Escritura en consola
   for (unsigned int i = 0; i < vx.size(); i++)
      std::cout << std::fixed << std::setprecision(10)
            << vx[i] << "\t"
            << std::scientific << std::setprecision(6)
            << vDerf[i] << std::endl;
   return 0;
}
```