Transformación de algoritmos recursivos en iterativos

Objetivos

- Realizar la transformación de algoritmos recursivos a iterativos utilizando el esquema general de transformación de algoritmos.
- Implementar algoritmos en C++.
- Gestionar pilas en un programa utilizando la librería stack.
- Trabajar en grupo.

Actividades

1.- Implementa la versión iterativa del siguiente algoritmo y realiza un programa que pida por teclado dos números y muestre el resultado de aplicar el algoritmo iterativo implementado.

```
función programa1(x:entero, y:entero):entero
   si x≤1
      devolver y
   si no
      devolver programa1(x-1, y+2) - x + y
   fsi
ffunción
```

2.- Implementa la versión iterativa del siguiente algoritmo, aplicando el esquema general de transformación apropiado, y realiza un programa que pida por teclado dos números y muestre el resultado de aplicar el algoritmo iterativo implementado. En el caso de que y sea un número negativo el programa imprimirá el texto Error.

```
función programa2(x:entero, y:natural U {0}):entero
   z:entero
   si y=0
      z ← 1
   si no
      z ← programa2(x,y/2)
      z ← z * z
      si y mod 2 = 1
         z ← z * x
      fsi
   fsi
   devolver z
ffunción
```

3.- Implementa la versión iterativa del siguiente algoritmo y realiza un programa que pida por teclado dos números y muestre el resultado de aplicar el algoritmo iterativo implementado.

```
función programa3(a:entero, b:entero):entero
   si a < 3
      devolver a + b
   si no
      devolver programa3(a/3, b−2) + a
   fsi
ffunción
```

Anexo: La clase pila en C++

En C++ existe una librería llamada stack que tiene implementada la clase pila y funciones para trabajar con ellas. Las funciones que incorpora esta librería son las siguientes:

Función	Descripción
push()	Introduce un elemento en la cima de la pila.
pop()	Elimina el elemento de la cima de la pila.
top()	Devuelve el elemento que se encuentra en la cima de la pila.
empty()	Verdadero si la pila está vacía. En caso contrario devuelve falso.
size()	Número de elementos de la pila.

Para declarar una variable del tipo pila se utiliza la instrucción

```
stack <tipodatos> nombrepila;
```

siendo tipodatos el tipo de datos de los datos que se guardan en la pila: int, float…

En el siguiente ejemplo se muestra cómo utilizar una variable del tipo stack y la llamada a las funciones de la librería. Este programa declara una variable llamada pila del tipo stack y almacena en ella 5 números enteros que se introducen por teclado. A continuación, muestra el tamaño de la pila y finalmente imprime todos los elementos que se han apilado hasta dejar la pila vacía.

```
#include <iostream>
#include <stack>
#include <stdlib.h>

using namespace std;

int main()
{
  stack <int> pila;
  int dato,i;


  cout << "Apilando datos" << endl;
  cout << "--------------" << endl;

  for (i=1; i<=5; i++){
    cout << "Introduce dato: ";
    cin >> dato;
    pila.push(dato);
  }

  cout << endl;
  cout << "Num. elementos de la pila: ";
  cout << pila.size() << endl << endl;

  cout << "Desapilando datos" << endl;
  cout << "-----------------" << endl;
  while (! pila.empty() )
  {
    dato = pila.top();
    cout << dato << endl;
    pila.pop();
  }

  cout << endl;
  system("pause");
  return 0;
}
```

Modo de entrega

La práctica se realizará en equipos de **dos o tres alumnos** (excepcionalmente de forma individual) y se entregarán los siguientes ficheros con los nombres que se indican.

Archivo comprimido:	practica4.zip
Contenido del archivo:	p4_1.cpp, p4_2.cpp, p4_3.cpp Cada fichero contiene el código fuente de las actividades 1, 2 y 3 y los nombres de los miembros del equipo, así como los nombres de los usuarios dentro del corrector online (aluXX, donde XX es el número).

Todos los componentes del equipo entregarán el archivo practica4.zip en la tarea llamada “Práctica 4: Transformación de algoritmos recursivos en iterativos”, dentro del acceso identificado de la página web de la asignatura. Además, también se entregarán los ficheros .cpp en la plataforma de corrección online.

Fecha fin de entrega: Domingo, 29 de marzo de 2020 a las 23:59.

Evaluación

La puntuación de cada actividad de la práctica es:

Actividad 1: 0,6 puntos
Actividad 2: 0,6 puntos
Actividad 3: 0,8 puntos

A continuación, se indica el sistema de evaluación:

Opción A:

Cada miembro del equipo debe entregar tanto la práctica en la tarea de la web de la asignatura como los programas aceptados en el corrector online (http://atlas.umh.es/mooshak). Se evaluará cogiendo al azar la práctica de uno de los miembros del equipo, de forma que dicha práctica será la que se corrija. La calificación será la suma del total de la puntuación de cada ejercicio correcto.

Opción B: 0 puntos

- El alumno/a:
  - No entrega la práctica en la tarea.
  - Los programas no son aceptados en el corrector online.
- El programa no realiza lo que se pide.
- Se detecta copia con otras prácticas (en la tarea y/o en el corrector online): La nota será un 0 en esta práctica para todas las prácticas implicadas.

Sobre la opción de evaluación continua y control de asistencia:

Esta práctica será impartida como una sesión virtual. Durante las sesiones virtuales de prácticas, no se controlará la asistencia mediante hoja de firmas o equivalente. Se considerará que un alumno ha asistido a esta sesión si entrega el material a la tarea antes de que venza su plazo de entrega. De esta forma, se recuerda a los alumnos que deberán cumplir el mínimo de asistencia exigido en cada una de las modalidades ofrecidas en la asignatura.