Universidad de Granada
Departamento de Ciencias de la Computación
e Inteligencia Artificial

Fundamentos de Programación (2015/16)
1º Grado en Ingeniería Informática (Prácticas)
15 de Septiembre de 2016

DECSAI

Normas para la realización del examen: **Duración: 1.5 horas**

- **Debe disponer de un documento oficial que acredite su identidad a disposición del profesor.**
- **Crear una carpeta en el escritorio llamada EXAMEN y copiar en ella el fichero `examen.cpp`. Modificar los datos que aparecen en la cabecera y escribir la solución en ese fichero.**
- **La entrega de la práctica se hará durante el periodo de tiempo en el que se realiza el examen, y desde los ordenadores instalados en el aula. Para efectuar la entrega se usará la plataforma `decsai` y se copiará ÚNICAMENTE el fichero `examen.cpp`.**

◁ Ejercicio 1 ▷ Búsqueda por Interpolación [3 puntos]

La **Búsqueda por Interpolación** del valor *buscado* en un vector *v* entre las posiciones *izda* y *dcha* es un método avanzado de búsqueda que recuerda a la *búsqueda binaria* porque:

- requiere que el vector en el que se va a realizar la búsqueda está ordenado, y
- en cada consulta sin éxito se descarta una parte del vector para la siguiente búsqueda.

La diferencia fundamental con la búsqueda binaria es la manera en que se calcula el elemento del vector que sirve de referencia en cada consulta (que ocupa la posición *pos*). Ya no es el que ocupa la posición central del subvector en el que se efectúa la búsqueda (el delimitado únicamente por *izda* y *dcha*), sino que depende también del contenido de esas casillas, de manera que *pos* será más cercana a *dcha* si *buscado* es más cercano a *v[dcha]* y más cercana a *izda* si *buscado* es más cercano a *v[izda]*.

En definitiva, se cumple la relación:

$$\frac{\texttt{pos} - \texttt{izda}}{\texttt{dcha} - \texttt{izda}} = \frac{\texttt{buscado} - \texttt{v[izda]}}{\texttt{v[dcha]} - \texttt{v[izda]}}$$

Las tareas a realizar en este examen son las siguientes:

1. Ampliar la clase `SecuenciaEnteros` con el método:

   ```
   int BusquedaInterpolacion (int buscado, int izda, int dcha);
   ```

2. Completar el programa esbozado en la función `main` de `examen.cpp` para que lea un número entero (*buscado*) y los extremos entre los que realizará la búsqueda (*izda* y *dcha*).
   Realizar cuantas comprobaciones y ajustes crea oportuno antes de llamar a la función `BusquedaInterpolacion`.
3. Llamar a la función `BusquedaInterpolacion` y mostrar el resultado.
4. Preguntar si se desea realizar otra búsqueda. En caso afirmativo, repetir los pasos anteriores. En caso negativo indicar el número de búsquedas realizadas y el número (y porcentaje) de éxitos y fracasos.

Debe completar el contenido de `examen.cpp`, que puede descargarse desde decsai. El fichero que proporcionamos contiene código que le permite crear una secuencia de enteros desde un array y mostrar su contenido. El contenido de `examen.cpp` es el siguiente:

```
/*****************************************************************************/
// FUNDAMENTOS DE PROGRAMACIÓN
// GRADO EN INGENIERÍA INFORMÁTICA
//
// CURSO 2015-2016
// DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA ARTIFICIAL
//
// Examen Septiembre 2016
// Examen práctico
//
// Búsqueda por interpolación
//
// APELLIDOS:
// NOMBRE:
// GRUPO:
//
// ORDENADOR:
//
/*****************************************************************************/

#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;


////////////////////////////////////////////////////////////////////////////////

class SecuenciaEnteros
{

private:

    static const int TAMANIO = 100; // Número de casillas disponibles
    int vector_privado[TAMANIO];

    //PRE: 0 <= total_utilizados <= TAMANIO
    int total_utilizados; // Número de casillas ocupadas

public:

    /*************************************************************************/
    // Constructor sin argumentos

    SecuenciaEnteros (void)
    {
        total_utilizados = 0;
    }

    /*************************************************************************/
    // Métodos de lectura (Get) de los datos individuales

    // Devuelve el número de casillas ocupadas
    int TotalUtilizados (void)
    {
        return (total_utilizados);
    }

    // Devuelve el número de casillas disponibles
    int Capacidad (void)
    {
        return (TAMANIO);
    }

    /*************************************************************************/
    // Añade un elemento al vector.
    // Recibe: nuevo, el elemento que se va a añadir.
    //
    // PRE: total_utilizados < TAMANIO
    //  La adición se realiza si hay espacio para el nuevo elemento.
    //  El nuevo elemento se coloca al final del vector.
    //  Si no hay espacio, no se hace nada.
```

```
    void Aniade (int nuevo)
    {
        if (total_utilizados < TAMANIO) {
            vector_privado[total_utilizados] = nuevo;
            total_utilizados++;
        }
    }

    /**************************************************************************/
    // Devuelve el elemento de la casilla "indice"
    //
    // PRE: 0 <= indice < total_utilizados

    int Elemento (int indice)
    {
        return vector_privado[indice];
    }

    /**************************************************************************/
};

////////////////////////////////////////////////////////////////////////////////


/**************************************************************************/
/**************************************************************************/

int main (void)
{
    // Array del que se tomarán los datos para formar la secuencia

    const int NUM_DATOS = 36;
    int datos[]={1,3,5,7,9,10,11,12,13,15,17,19,20,21,23,25,27,29,30,
                 33,35,37,38,39,41,43,56,47,49,51,53,55,57,58,59,60};


    // Formar la secuencia v

    SecuenciaEnteros v;

    for (int i=0; i< NUM_DATOS; i++)
        v.Aniade(datos[i]);


    // Mostrar el contenido de v

    for (int i=0; i< v.TotalUtilizados(); i++)
        cout << "Elemento num. " << setw(3) << i
             << ": " << setw(3) << v.Elemento(i) << endl;

    cout << endl;

    // EL EXAMEN EMPIEZA AQUÍ //



    return (0);
}
```