FUNDAMENTOS DE LA PROGRAMACIÓN
1º Grado en Ingeniería Informática
- Febrero 2017 -
Dpto. Tecnologías de la Información
Escuela Técnica Superior de Ingeniería(Universidad de Huelva)

Puntuación máxima de este examen: 5 puntos.

Problema n. 1.- Este problema pretende evaluar las habilidades del alumno para **implementar métodos dentro de una class**, en concreto debe de realizar los métodos **cargarDatosColegio, buscar y mismo** de la clase colegio:

```
typedef char cad[20];
struct alumno {
        cad nombre;                                  //Nombre del alumno
        cad asigalumno[12];
                          //Nombre de las asignaturas en que está matriculado
        int n;                              //Número de asignaturas matriculadas
};
struct profesor {
        cad nombre;                                  //Nombre del profesor
        cad asigprofesor[5];
                           //Nombre de las asignaturas que imparte el profesor
        int n;                              //Número de asignaturas que imparte
};
class colegio{
        cad nombre;                                  //Nombre del colegio
        cad direccion;                               //Dirección del colegio
        alumno alumnos[200];                 //alumnos inscritos en el colegio
        int nalumn;                                  //Número de alumnos inscritos
        profesor profesores[21];                     //Profesores del colegio
        int nprof;                                   //Número de profesores
 public:
        colegio()  { nalumn=0; nprof=0;}
        void cargarDatosColegio();                              (0.9 Puntos)
        /* Pedirá por teclado el nombre y dirección del colegio, pedirá cuantos
        alumnos se van a inscribir, para cada alumno se pedirá el nombre, de
        cuantas asignaturas se va a matricular y el nombre de cada una de esas
        asignaturas. Pedirá por teclado cuantos profesores hay en el centro, y
        para cada uno de ellos pedirá su nombre, cuantas asignaturas imparte y el
        nombre de cada una de esas asignaturas. Almacenando todos estos
        valores en los atributos*/

        alumno buscar(cad nom);                                 (0.9 Puntos)
        /* Recibiendo en el parámetro nom el nombre de un alumno, lo buscará
        en la tabla alumnos, devolviendo un struct alumno (nombre,
        asignaturas…) con toda la información de ese alumno en caso de que
        exista, si no existe devolverá un struct alumno con el nombre "NO HAY
        ALUMNO"    */

        bool mismo(colegio c);                                  (0.7 Puntos)
        /* Devolverá true si el nombre y la dirección del colegio c coinciden con el
        nombre y la dirección respectivamente del objeto al que se llama, en caso
        contrario devolverá false*/
        };
```

Problema n. 2.- Este problema pretende evaluar las habilidades del alumno para **generar programas que utilicen código (clases y funciones genéricas) ya implementados**.

Para ello disponemos de:

- La clase **Trabajador** que almacena los datos de un trabajador
- La clase **Obra** que almacena la información de los trabajadores que trabajan en una determinada obra.
- La función genérica **LocalizarTrabajador** que como su identificador indica, localiza un trabajador dentro de la plantilla de trabajadores.

Con estos recursos, el alumno deberá implementar dentro de la función **main** el código necesario para resolver un problema determinado que se explica dentro del propio main más adelante.

Comenzamos explicando ambas clases cuyos métodos consideramos implementados y que se pueden utilizar si nos hace falta.

- La clase Trabajador almacena los datos de un trabajador.

```
typedef char Cadena[100];.
class Trabajador  {                          //Almacena los datos de un trabajador.
     Cadena Nombre;                                  //Nombre del trabajador.
     Cadena Banco;                          //Banco donde tiene su cuenta corriente.
     float Sueldo;                          //Contiene el sueldo que gana cada mes.
     int Dni;                                        //Contiene el dni del trabajador.
public:
    //Devuelve el nombre del trabajador en el parámetro pNombre.
    void getNombre(Cadena pNombre);

    //Devuelve el nombre del banco donde tiene la cuenta corriente
    void getBanco(Cadena pBanco);

   float getSueldo();                     //Devuelve el sueldo que gana cada mes
   int getDni();                                  //Devuelve el Dni del trabajador
};
```

- La clase Obra almacena los datos de una obra, y los Dni de sus trabajadores.

```
class Obra {
     Cadena Direccion;                                    //Dirección de la obra.
     int Trabajadores[100];                  //Dni de los trabajadores de la obra.
     int NTrabajadores;                      //Número de trabajadores en la obra.
public:
     //Devuelve la dirección de la obra en el parámetro pDireccion.
     void getDireccion(Cadena pDireccion);

     int getNTrabajadores();                 //Devuelve el número de trabajadores

     //Devuelve el Dni del trabajador situado en la posición Pos
     void getDniTrabajador(int Pos, int &Dni);

     //Devuelve la posición del trabajador dentro del vector Trabajadores,
     //cuyo dni coincide con el pasado por parámetro. -1 si no lo encuentra.
     int BuscarTrabajador(int Dni);
};
```

- Función Genérica que devuelve la posición dentro del vector *pPlantilla* de aquel trabajador cuyo *dni* coincide con el pasado por parámetro. El número de posiciones del vector ocupadas con datos correctos lo indica el parámetro *pTamaPlantilla*. En caso de no localizar ningún trabajador con ese dni la función devuelve -1.

```
int LocalizarTrabajador(Trabajador pPlantilla[500], int pTamaPlantilla, int dni);
```

- Programa principal que tiene definido un vector de trabajadores de la empresa y un vector de obras que la empresa tiene contratadas por particulares.

```
int main() {
   Trabajador Plantilla[500];                    //Trabajadores de la empresa.
   int TamaPlantilla;                  //Número de trabajadores contratados.
   Obra Trabajos[200];                                  //Obras contratadas.
   int NTrabajos;                              //Número de obras contratadas.
    …
 /*  Código a Implementar                                        (2.5 puntos)
```

Implemente un código que solicite por teclado la dirección de una obra y muestre por pantalla los nombres y salario de todos los trabajadores que están en dicha obra. Además al final del listado se debe mostrar la suma total de los salarios de los trabajadores de esa obra.

```
  */
  return 0;
}
```