UTN – FR Mar del Plata - Técnico Universitario en Programación
Programación II y Laboratorio II

Trabajo Práctico N° 2: Estructuras de Datos Estáticas y Dinámicas

1. Dada la estructura:

```
typedef struct {
      char nombre[30];
      char genero;
      int edad;
}persona;
```

a. Crear un arreglo estático de 30 elementos de esta estructura y cargarlo mediante una función.

b. Hacer una función que cuente la cantidad de un género determinado.

c. Hacer una función que copie los datos de todos los registros de un género determinado del arreglo anterior en otro arreglo del tamaño justo. Usar malloc dentro de la función y retornarlo o usar dos parámetros de tipo arreglo y crear el arreglo dinámico antes de la invocación.

2. Codificar el algoritmo de ordenamiento por selección, sobre la estructura anterior teniendo en cuenta la edad como criterio de ordenación.

3. Simular el uso de una pila de enteros a partir de la siguiente estructura:

```
typedef struct{
     int valores[100];
     int posTope; //posición de nuevo tope, lugar en donde se almacenará el nuevo valor
} Pila;
```

Implementar las siguientes funciones:

```
void apilar(Pila * p, int valor);
int desapilar(Pila * p);
int tope(Pila * p);
int pilavacia(Pila * p);
void mostrar(Pila * p);
void leer (Pila * p);
void inicPila(Pila * p);
```

4. Dadas dos pilas (como las anteriores) que se supone que tienen sus datos ordenados, generar una tercer pila que contenga los datos de las dos anteriores intercalados, de forma tal que queden ordenados.

5. Función insertar orden en un arreglo.

6. Algoritmo de ordenamiento por inserción.

7. Función eliminar un elemento de un arreglo. (ver por desplazamiento si es ordenado)

8. Sistema de notas. Se tiene que administrar un sistema para un curso con 20 alumnos que cursan 5 materias diferentes. Se deben almacenar los datos de los alumnos y las notas que obtuvieron en el examen final de cada materia.
Para ello se utilizarán las siguientes estructuras:

```
typedef struct {
  int matricula;
  char nombre[30];
} Alumno;
```

```
typedef struct {
  int codigo;
  char nombreMat[20];
} Materia ;
```

```
typedef struct {
  int matricula;
  int codigo;
  int nota;
} Nota;
```

Y las siguientes variables:

```
Alumno alus[20]; // para almacenar los datos de los 20 alumnos.
Materia mats[5]; // para almacenar los códigos y nombres de las 5 materias.
Nota notas[100]; // para almacenar todas las notas de los alumnos.
```

Las notas se almacenan en forma desordenada para todos los alumnos.

Se deben hacer las siguientes funciones:

1. Cargar el arreglo de Materia
2. Agregar un Alumno
3. Agregar una Nota, validando que exista “matrícula” y “código”. La función debe retornar 1 si la operación fue exitosa y 0 (cero) en caso contrario.
4. Hacer una función que muestre por pantalla el nombre del alumno y la lista de materias con sus respectivas notas. La función debe recibir como parámetro el nombre del alumno. *Estrategia: con el nombre del alumno se busca su matrícula en el arreglo alus, luego se recorre el arreglo notas filtrandolo por matrícula, y con el código se accede al nombre de la materia revisando el arreglo mats.*