Práctica 3

Tipos de datos compuestos – Estructuras de datos dinámicas

Tipos de datos compuestos

1. Resuelva:
   a. Defina una estructura `rectangulo` que contenga los siguientes campos: `base` (`float`) y `altura` (`float`).
   b. Escriba una función que reciba una estructura `rectangulo` y la inicialice a partir de valores ingresados por teclado.
   c. Escriba una función que dada una estructura `rectangulo`, calcule el área.
   d. Escriba un programa que defina un arreglo de 10 rectángulos (`struct rectangulo`) y lo inicialice utilizando la función definida en el inciso anterior. Luego, informe el número de rectángulo que tiene menor área.

2. Dados los siguientes bloques de código:

```c
struct persona {
    char nombre[50];
    long int DNI;
} unaPersona;

typedef struct persona persona_t;
```

```c
typedef struct persona {
    char nombre[50];
    long int DNI;
} persona_t;
```

   ¿En qué se diferencian ambos bloques? ¿Qué define cada uno?

3. Resuelva:
   a. Defina una estructura direccion que contenga los siguientes campos: calle (arreglo de 50 caracteres), ciudad (arreglo de 30 caracteres), codigo_postal (int) y pais (arreglo de 40 caracteres).
   b. Defina una estructura `alu` que contenga los siguientes campos: `apellido` (arreglo de 50 caracteres), `nombre` (arreglo de 50 caracteres), `legajo` (arreglo de 8 caracteres), `promedio` (`float`) y `domicilio` (`struct direccion`).
      a. Renombre el tipo struct alu a alumno mediante la palabra clave typedef.
      b. Escriba una función que reciba un alumno y lo inicialice a partir de valores ingresados por teclado.
      c. Escriba un programa que defina un arreglo de 30 elementos alumno y lo inicialice utilizando la función definida en el inciso anterior. Luego, informe el nombre y apellido del alumno que tiene mejor promedio.
   c. Defina la estructura `pun3D`, la cual representa una posición en el espacio. La misma debe contener los campos `x` (`float`), `y` (`float`) y `z` (`float`). Luego:
      a. Renombre la estructura `pun3D` a `punto3D` utilizando la palabra clave `typedef`.
      b. Imprima en pantalla el tamaño del tipo `struct pun3D`. ¿Cuánto ocupa? ¿Por qué?
      c. Imprima en pantalla el tamaño del tipo `punto3D`. ¿Cuánto ocupa? ¿Es igual al de `struct pun3D`? ¿Por qué?
      d. Defina un arreglo de 4 elementos de tipo `punto3D` e imprima en pantalla el espacio ocupado por el mismo. ¿Cuánto ocupa? ¿Por qué?

4. Implemente una estructura y las funciones para implementar un mazo de 40 cartas españolas. Implemente las siguientes funciones y realice un programa para probarlas:

   a. Barajar el mazo de cartas.
   b. Sacar una carta: dado un mazo, sacar la carta del mazo y devolverla.
   c. Imprimir una carta (número/figura con su palo).
   Nota: utilice constantes (define o const) para definir los palos de las cartas, modelice las cartas y el mazo.

Estructuras de datos dinámicas

5. Responda:

   a. ¿En qué se diferencia la función **malloc** de la función **calloc**? ¿y de la función **realloc**?
   b. ¿Puede utilizarse la función **realloc** en lugar de la función **malloc**? ¿Se requiere alguna condición? ¿Qué sucede si **realloc** se invoca con el valor 0 como parámetro?
   c. ¿Qué utilidad tiene el operador **sizeof** a la hora de reservar memoria dinámicamente?

6. Analice, compile y ejecute el siguiente código. Indique imprime y porque.

```c
#include <stdio.h>
#include <stdlib.h>

void f (int * p);

int main(){

    int * ptr = NULL;
    f(ptr);

    if (ptr == NULL)
        printf("ptr es NULL\n");
    else
        printf("ptr no es NULL\n");

    return 0;
}

void f (int * p) {
    p = (int *) malloc(10*sizeof(int));
}
```

7. Escriba un programa que lea un número entero ***n*** desde teclado y luego reserve memoria en forma dinámica para un arreglo de ***n*** enteros. Inicialícelo con valores aleatorios y a continuación calcule e imprima el máximo número almacenado. Por último, libere la memoria reservada dinámicamente. *Nota: Modularice la reserva de memoria, la inicialización y el cálculo del máximo.*

8. Escriba un programa que lea un número entero ***n*** desde teclado y luego reserve memoria en forma dinámica para un arreglo de ***n*** float. Inicialícelo con valores ingresados por teclado y a continuación calcule e imprima el promedio de todos ellos. Por último, libere la memoria reservada dinámicamente. Nota: Modularice la reserva de memoria, la inicialización y el cálculo del promedio.

9. Escriba un programa que reserve en forma dinámica un arreglo de 100 caracteres. A continuación, lea 10 oraciones utilizando la función **gets** e informe para cada una de ellas la cantidad de letras minúsculas y de letras mayúsculas que la componen. Utilice el arreglo creado como variable temporal para procesar cada oración. Por último, libere la memoria reservada dinámicamente.

10. Defina la estructura de una lista enlazada de enteros. Implemente las siguientes funciones:
    a. Inicializar la lista.
    b. Eliminar todos los elementos de la lista.
    c. Agregar un elemento al principio de la lista.
    d. Agregar un elemento al final de la lista.
    e. Calcular la cantidad de elementos de la lista.
    f. Imprimir todos los elementos separados por coma.

11. Utilizando la estructura y funciones del ejercicio 9 escriba un programa que lea números enteros positivos desde teclado hasta ingresar el número 0. Los números leídos deben ser almacenados en orden ingresado por teclado. Generar una nueva lista con el orden invertido. Imprimir los elementos de cada lista junto con la cantidad de elementos de cada una. Por último, libere la memoria reservada dinámicamente.

12. Incorpore al ejercicio 10 las siguientes funciones, modifique la estructura y reimplemente las funciones según corresponda:
    a. Insertar un elemento de forma ordenada.
    b. Reimplementar la función que calcula la cantidad de elementos de forma eficiente (sin recorrer la lista). Analice y modifique todas las funciones necesarias.

13. Utilizando la estructura y funciones de los ejercicios anteriores escriba un programa que lea números enteros positivos desde teclado hasta ingresar el número 0. Los números leídos deben ser almacenados en orden ascendente en una lista enlazada. Generar 2 listas nuevas con los números pares e impares. Imprima las 3 listas junto con la cantidad de elementos. Por último libere la memoria reservada dinámicamente.

14. Utilizando la estructura y funciones de los ejercicios anteriores escriba un programa que lea números enteros desde teclado hasta ingresar el número 0. Luego, vuelva a leer otro número entero desde teclado y elimine de la lista a todos aquellos que sean múltiplos del mismo.