	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor:	Adrian Ulises Mercado Martinez
Asignatura:	Estructura de Datos I
Grupo:	10
No de Práctica(s):	3
Integrante(s):	Castañeda Mora Carlos Sicilia Pablo Breton Arturo
No. de Equipo de cómputo empleado:	1,2 y3
No. de Lista o Brigada:	1
Semestre:	2
Fecha de entrega:	25 de febrero del 2020
Observaciones:	

CALIFICACIÓN: _____

Práctica3: Uso de los apuntadores.

Castañeda Mora Carlos
González Bretón Arturo
Sicilia Pablo

25 febrero 2020

1 Introduction

Las estructuras son colecciones de variables relacionadas bajo un nombre. Las estructuras pueden contener variables de muchos tipos diferentes de datos - a diferencia de los arreglos que contienen unicamente elementos de un mismo tipo de datos.

Las estructuras son tipos de datos derivados - estan construidas utilizando objetos de otros tipos. La palabra reservada struct indica se esta definiendo una estructura. El identificador ejemplo es el nombre de la estructura. Las variables declaradas dentro de las llaves de la definicion de estructura son los miembros de la estructura. Los miembros de la misma estructura deben tener nombres unicos mientras que dos estructuras diferentes pueden tener miembros con el mismo nombre. Cada definicion de estructura debe terminar con un punto y coma. La definicion de struct ejemplo contiene un miembro de tipo char y otro de tipo int. Los miembros de una estructura pueden ser variables de los tipos de datos basicos (int, char, float,etc) o agregados como ser arreglos y otras estructuras. Una estructura no puede contener una instancia de si misma.

2 Desarrollo

1) Archivo ,h donde declaramos la estructura de tipo matriz, además de las funciones que usará.

```
#ifndef MATRIZ_H
#define MATRIZ_H

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

typedef struct _matriz matriz;
struct _matriz{
    int row;
    int column;
    float **datos;
};

void rellenar_matriz(matriz *m);
void suma(matriz *m1,matriz *m2, matriz *mr);
void orden_matriz(matriz *m1);
void multi(matriz *m1,matriz *m2,matriz *mr);
void print(matriz *m1);
int probar_multi(matriz *m1,matriz *m2);
int probar_suma(matriz *m1,matriz *m2);
void crear(matriz *m1);
#endif
```

2)Primera parte de un archivo que contiene a las funciones que ejecuta el programa. Se muestran funciones para crear una matriz, recibir el orden de una matriz, comprobar si se pueden multiplicar o sumar y llenar de datos una matriz.

```
#include "matriz.h"

void orden_matriz(matriz *m1){
    printf("Ingrese el numero de filas de la matriz: \n");
    scanf("%d",&m1->row);
    printf("Ingrese el numero de columnas de la matriz: \n");
    scanf("%d",&m1->column);
}
int probar_suma(matriz *m1,matriz *m2){
    if((m1->row==m2->row) && (m1->column==m2->column)){
        return 5;
    }
    return 2;
}
int probar_multi(matriz *m1,matriz *m2){
    if(m1->column==m2->row){
        return 5;
    }
    return 2;
}
void crear(matriz *m1){
    int i;
    m1->datos=(float**)malloc(sizeof(float*)*m1->row);
    for(i=0;i<m1->row;i++){
        m1->datos[i]=(float*)malloc(sizeof(float)*m1->column);
    }
}
void rellenar_matriz(matriz *m){
    int i,j;
    for(i=0;i<m->row;i++){
        for(j=0;j<m->column;j++){
            printf("Ingrese el valor para el lugar [%d][%d]\n",i,j);
            scanf("%f",&m->datos[i][j]);
        }
```

3)Segunda parte del archivo, se muestran las funciones para sumar matrices, multiplicar matrices e imprimir las matrices en pantalla.

```
}
void rellenar_matriz(matriz *m){
    int i,j;
    for(i=0;i<m->row;i++){
        for(j=0;j<m->column;j++){
            printf("Ingrese el valor para el lugar [%d][%d]\n",i,j);
            scanf("%f",&m->datos[i][j]);
        }
    }
}
void suma(matriz *m1,matriz *m2, matriz *mr){
    int i,j;
    for(i=0;i<m1->row;i++){
        for(j=0;j<m2->column;j++){
            mr->datos[i][j]=m1->datos[i][j]+m2->datos[i][j];
        }
    }
}
void multi(matriz *m1,matriz *m2,matriz *mr){
    int i,j,k;
    for(i=0;i<m1->row;i++){
        for(j=0;j<m1->column;j++){
            for(k=0;k<m2->row;k++){
                mr->datos[i][j]=mr->datos[i][j]+(m1->datos[i][k]*m2->datos[k][j]);
            }
        }

    }
}
void print(matriz *m1){
    int i,j,k;
    for(i=0;i<m1->row;i++){
        printf("\n");
        for(j=0;j<m1->column;j++){
            printf("%f\t",m1->datos[i][j]);
        }
    }
}
```

4) Primera parte del archivo principal, donde se eligen la opción a realizar, además se nuestra como se implementan las funciones.

```
                }
            }
            else{
                printf("No se puede sumar, el orden  no es correcto.\n");
            }
            break;
        case 2:
            for(i=0;i<2;i++){
                printf("Matriz %d",i);
                orden_matriz((m+i));
            }
            if(probar_multi(m,(m+1))){
                (m+2)->row=(m+1)->row;
                (m+2)->column=m->column;
                for(i=0;i<3;i++){
                    crear(m+i);


                }
                for(i=0;i<2;i++){
                    rellenar_matriz(m+i);
                }
                multi(m,(m+1),(m+2));
                for(i=0;i<3;i++){
                    print((m+i));
                    printf("\n");
                }
            }
            else{
                printf("No se puede sumar, el orden  no es correcto.\n");
            }
            break;
        case 3:
            return 0;
    }
}
    return 0;
}
```

5)Segunda parte del archivo principal.

```
#include "matriz.h"

int main(){
    matriz *m;
    int i,j,x;
    m=(matriz*)malloc(sizeof(matriz)*3);
    while(1){
        printf("1)Sumar\n2)Multiplicar\n");
        scanf("%d",&x);
        switch(x){
            case 1:
                for(i=0;i<2;i++){
                    printf("Matriz %d",i);
                    orden_matriz((m+i));
                }
                if(probar_suma(m,(m+1))==5){
                    (m+2)->row=m->row;
                    (m+2)->column=m->column;
                    for(i=0;i<3;i++){
                        crear(m+i);

                    }
                    for(i=0;i<2;i++){
                        rellenar_matriz(m+i);
                    }
                    suma((m),(m+1),(m+2));
                    for(i=0;i<3;i++){
                        print(m+i);
                        printf("\n");
                    }
                }
                else{
                    printf("No se puede sumar, el orden  no es correcto.\n");
                }
                break;
            case 2:
                for(i=0;i<2;i++){
```

3 Conclusion

Durante esta práctica aplicamos a un programa los tipos de datos definidos por el programador o estructuras, por lo que aprendimos que este tipo de dato nos brinda una gran facilidad para la organización sistemática de tipos de datos diferentes bajo un mismo nombre además de la sistematización de sus operaciones. Pablo

En esta práctica trabajamos con apuntadores y estreucturas para elaborar una matriz y las operaciones básicas de esta. Se puede observar la utilidad de los apuntadroes para manejar la memoria y crear matrices tan grandes como se quiera. Carlos

4 Bibliografía

https://es.wikipedia.org/wiki/N

https://concepto.de/estructuras/

https://www.fing.edu.uy/tecnoinf/mvd/cursos/prinprog/material/teo/prinprog-teorico08.pdf