Lenguaje C
Nombre: Pichardo Rico Cristian Eduardo **Computación I** **Grupo: 9080**

Práctica 4

Matrices I

Ejercicio 1.Escribe, compile y ejecute los siguientes programas.

Programa 1:

El programa 1 es un acercamiento inicial a las matrices en el lenguaje C. Lo que hace este programa es imprimir el valor de cada una de las celdas de la matriz que hemos ingresado.

Primeramente, abrimos el *main* y declaramos la variable entera *i* y la definimos como "*0*", y también declaramos la matriz de un renglón y ocho columnas (*columna[8]*). Posteriormente, definimos el valor de cada una de las ocho columnas, siendo la primera "*columna[0]=0*", la segunda "*columna[1]=1*", la tercera "*columna[2]=2*" y así sucesivamente hasta llegar a la octava columna definida como "*columna[7]=7*". Después, simplemente ocupamos el ciclo *for* para imprimir el valor de cada una de las columnas mediante este formato: "*El valor de la celda %u es %u*" donde *%u* se utiliza para imprimir un *entero sin signo*, y en donde la primera *%u* imprime el número de la columna y el segundo *%u* imprime el valor definido a esa columna.

El ciclo *for* se ha especificado mediante la condición que la variable entera *i* debe ser menor a 8, en un principio esta variable esta definida como "*i=0*" pero al empezar el ciclo *for* la variable queda definida como "*i=1*" y esta incrementará su valor en uno (es decir, se le sumará 1) cada vez que el ciclo vuelva a repetirse y se detendrá cuando ya no se cumpla la condición del ciclo (es decir, cuando *i* ya no sea menor que 8).

Finalmente cerramos el ciclo *for* y cerramos el programa con *return 0. Así*, el código queda dela siguiente manera:

```
//El siguiente programa muestra el llenado deuna matriz mediante la asignacion
//directa. Todas las variables que componen la matriz tienen el mismo nombre "semana"
//pro diferente indice (el valor del argumento i=0.1.2...7)

#include <stdio.h>

int main(void){

     int semana[8];
     int i=0;

     semana[0]=0;
     semana[1]=1;
     semana[2]=2;
     semana[3]=3;
     semana[4]=4;
     semana[5]=5;
     semana[6]=6;
     semana[7]=7;

     for(i=1;i<8;i++){
       printf("El valor de la celda %u es %u\n",i,semana[i]);
     }

     return 0;
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir el numero de la columna de la matriz y el valor asociada a ella. En este caso, todo queda perfecto para que se imprima el valor de todas las columnas.

```
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes   matriz.c  Plantillas  Vídeos
Desktop    examples.desktop  Matriz1.c  Música    Público
ubuntu@ubuntu:~$ gcc -o matriz matriz.c
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes  Matriz1.c  Música      Público
Desktop    examples.desktop  matriz    matriz.c   Plantillas  Vídeos
ubuntu@ubuntu:~$ ./matriz
El valor de la celda 1 es 1
El valor de la celda 2 es 2
El valor de la celda 3 es 3
El valor de la celda 4 es 4
El valor de la celda 5 es 5
El valor de la celda 6 es 6
El valor de la celda 7 es 7
ubuntu@ubuntu:~$
```

Programa 2:
El código de este programa se escribe de manera similar al anterior, unicamente que ahora declaramos la matriz de un renglon y ocho columnas como *char* y definimos cada columna de la matriz como un carácter. En este caso, definimos el valor de cada una de las ocho columnas como "*columna[0]='a'*" a la primer columna, la segunda como "*columna[1]='b'*", la tercera "*columna[2]='c'*" y así sucesivamente hasta llegar a la octava columna definida como "*columna[7]='h'*". También se modificó la función *printf* para imprimir el valor de cada columna, ya que estos valores al ser de tipo *char* se necesitan especificar con *%c* (que imprime caracteres).

Un hecho a destacar es que para asignar un número a una variable se colocan los números directamente (ejemplo, *int i=12*), sin embargo, al momento de asignar caracteres de hacemediante el uso de las comillas simples (ejemplo, *char arg='a'*).

El código quedó de la siguiente manera:

```c
//Este programa hacelo mismo pero ahora con caracteres.

#include <stdio.h>

int main(void){

     char semana[8];
     int i=0;

     semana[0]='a';
     semana[1]='b';
     semana[2]='c';
     semana[3]='d';
     semana[4]='e';
     semana[5]='f';
     semana[6]='g';
     semana[7]='h';

     for(i=1;i<8;i++){
       printf("El valor de la celda %u es %c\n",i,semana[i]);
     }

     return 0;
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir el numero de la columna de la matriz y el valor asociada a ella, que en este caso es un carácter.

```
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes   Música      Público
Desktop    examples.desktop  matriz1.c  Plantillas  Vídeos
ubuntu@ubuntu:~$ gcc -o matriz matriz1.c
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes   matriz1.c  Plantillas  Vídeos
Desktop    examples.desktop  matriz     Música     Público
ubuntu@ubuntu:~$ ./matriz
El valor de la celda 1 es b
El valor de la celda 2 es c
El valor de la celda 3 es d
El valor de la celda 4 es e
El valor de la celda 5 es f
El valor de la celda 6 es g
El valor de la celda 7 es h
ubuntu@ubuntu:~$
```

Programa 3:
Este es un programa sencillo donde se utilizan las matrices para sacar la distancia de un objeto en sus ejes coordenados (es decir, su distancia en x y en y) a partir de las velocidades que tiene en los ejes coordenados y el tiempo que recorrió durante un trayecto determinado.

Primeramente, abrimos el *main* y declaramos las variables. Primero declaramos la matriz de un renglón y dos columnas "*velocidad[2]*" y definimos sus columnas como "*velocidad[2]={4.55, 6.73}*" es decir, que la columna *velocidad[0]=4,55* y la columna *velocidad[1]=6,73.* Después declaramos las variables flotantes *distanciax* y *distanciay* y ambas las definimos como 0. También declaramos la variable flotante *tiempo* y la definimos como "*tiempo=3.00*" y declaramos la matriz de un renglón y dos columnas "*coordenada[2]={'x', 'y'}*", es decir, la columna 1 de esta matriz queda definida como *coordenada[0]='x'* y la columna 2 queda definida como *coordenada[1]='y'*.

Después definimos que la variable "*distanciax*" es igual al producto "*velocidad[0]*tiempo*", es decir, la primer columna de la matriz *velocidad[2]* multiplicada por la variable *tiempo*. De igual manera, la variable "*distanciay*" se definió como el producto entre la segunda columna dela matriz *velocidad[2]* y la variable *tiempo*.

Por ultimo, se utilizo la función *printf* para imprimir la coordenada y la distancia en esa coordenada. El código quedo dela siguiente manera:

```c
//Llenado deuna matriz con dato, una sola vez

#include <stdio.h>

int main(void){

     float velocidad[2]={4.55,6.73};
     float distanciax=0.00;
     float distanciay=0.00;
     float tiempo=3.00; //media horas
     char coordenada[2]={'x','y'};

     distanciax=velocidad[0]*tiempo;
     distanciay=velocidad[1]*tiempo;

     printf("La distancia en %c es %f\n",coordenada[0], distanciax);
     printf("\nLa distancia en %c es %f\n",coordenada[1], distanciay);

     return(0);
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir las frases "*La distancia en x es %f*" donde x es eje coordenado x y %f es la distancia en la coordenada x. Y la frase "*La distancia en y es %f*" donde y es eje coordenado y y %f es la distancia en la coordenada y.

Programa 4:
Este programa sirve para multiplicar un vector por un escalar.
Primero abrimos el *main* y declaramos las variables. En esta ocasión declaramos la matriz de un renglón y dos columnas "*vector[2]={0,0}*" la cual será nuestro vector, también declaramos la matriz de un renglón y dos columnas "*campo[2]={0,0}*" que usaremos para guardar las variables que salen como resultado del producto del vector con un escalar. Y declaramos la variable flotante "*escalar=0,000*" que será nuestra variable que usaremos como escalar.

Una vez declaradas y definidas, utilizamos la función *puts* para imprimir una cadena de caracteres (muy similar a la función printf) que pida introducir el valor de la componente x del vector y con la función *scanf* guardamos el numero introducido en la variable *vector[0]* (La cual es la primera columna de nuestra matriz *vector[2]*) y hacemos lo mismo con la componente y del vector (usamos puts para imprimir una frase que pida el valor de la componente y del vector y luego usamos *scanf* para guardar ese numero en la variable *vector[1]* que es la segunda columna de nuestra matriz *vector[2]*).

Después utilizamos la función *printf* para observar el vector que se ha introducido y nuevamente usamos *puts* para pedir, ahora, el factor de estiramiento o el numero escalar con el que se va a efectuar la multiplicación por escalar y nuevamente usamos *scanf* para guardar el numero introducido en la variable *escalar*.

Ahora, efectuamos las operaciones dela siguiente manera:
Definimos la primera columna de campo*[2]* como "campo*[0]=vector[0]*escalar*" y, similarmente, definimos la segunda columna de campo*[2]* como "campo*[1]=vector[1]*escalar*".

Por ultimo, usando la función *printf* imprimimos la frase "El vector estirado en %g es [%g,%g]", donde el primer %g es la variable *escalar* y los otros dos %g son campo*[0]* y campo*[1]* respectivamente.

Por ultimo, usamos *puts* para imprimir en pantalla "Oprime return para salir"y usamos la función *getchar()* para capturar ese enter y pasar a *return* 0 *d*onde termina el código.

El código de este programa quedo dela siguiente manera:

```c
//Obtendremos unvector estirado

#include <stdio.h>

int main(void){

     float vector[2]={0,0};
     float escalar=0.000;
     float campo[2]={0.000,0.000};

     puts("Escribe el componente x del vector: ");
     scanf("%f", &vector[0]);
     puts("Escribe el componente y del vector: ");
     scanf("%f", &vector[1]);

     printf("El vector introducido es [%g,%g]\n\n",vector[0],vector[1]);

     puts("Escribe, ahora, el factor de estiramiento: ");
     scanf("%f", &escalar);

     campo[0]=escalar*vector[0];
     campo[1]=escalar*vector[1];

     printf("\nEl vector estirado en %g unidades es [%g, %g]\n\n", escalar, campo[0], campo[1]);

     puts("Oprime return para salir: ");
     getchar();

     return(0);
}
```

Compilamos y ejecutamos el programa y lo que hace es pedir las componentes *x* y *y* de un vector, lo imprime en pantalla y luego pide el factor de estiramiento (para hacer el producto por escalar) y después imprime "*El vector estirado en %g es [%g,%g]*", donde el primer %g es la variable *escalar* y los otros dos %g son campo*[0]* y campo*[1]* respectivamente que son las componentes del vector estirado. Luego se imprime "*Presione return para salir*" y se espera a que introduzcamos *return* para terminar el programa.

```
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes   Música      Público
Desktop    examples.desktop  matriz3.c  Plantillas  Vídeos
ubuntu@ubuntu:~$ gcc -o matriz matriz3.c
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes  matriz3.c  Plantillas  Vídeos
Desktop    examples.desktop  matriz    Música     Público
ubuntu@ubuntu:~$ ./matriz
Escribe el componente x del vector:
2
Escribe el componente y del vector:
2
El vector introducido es [2,2]

Escribe, ahora, el factor de estiramiento:
5

El vector estirado en 5 unidades es [10, 10]

Oprime return para salir:
ubuntu@ubuntu:~$
```

Programa 5.

Este ejercicio es cortito, consiste en un programa que imprime todos los valores de todas las columnas de una matriz de un renglón y de n columnas, donde n es natural, en nuestro caso sera para una matriz de 7 columnas.

Primero abrimos el *main* y declaramos la variable entera "*j=0*" y la matriz "*semana[7]={'d','l','m','i','j','v','s'}*". Después creamos un ciclo *for* donde defina a *j* como 1 y que se ejecute cuando la variable j sea menor que 7 y que en cada vuelta j incremente uno. Este ciclo va a imprimir el numero de la columna correspondiente junto con su valor.

Terminamos el código y este quedo así:

```c
//Programa paramostrar el llenado porteclado de una matriz

#include <stdio.h>

int main(void){

      char semana[7]={'d','l','m','i','j','v','s'}; //matriz a llenar
      int j=0; //contador

      for(j=0;j<7;j++){
        printf("Celda no. %u=%c\n",j,semana[j]);
       }



      return(0);
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir el numero de columna de la matriz junto con su valor correspondiente.

Programa 6.
Este ejercicio es aún más cortito, consiste en un programa que imprime todos los valores de todas las columnas de una matriz de un renglón y de n columnas, donde n es natural pero en forma de cadena por lo que la cadena debe determinar en carácter nulo, es decir, en cero. En nuestro caso sera para una matriz de 8 columnas.

Primero abrimos el *main* y declaramos la variable entera "*i=0*" y la matriz "*semana[8]={'d','l','m','i','j','v','s',0}*", nótese que esta matriz termina en carácter nulo. Después, mediante la función *printf* hacemos que se imprima en pantalla la palabra "*Cadena*" seguida de la cadena del valor de todas las columnas de la matriz (en nuestro caso: dlmijvs).
El código del programa quedo dela siguiente manera:

```c
//Esta es otra forma de inicializar una matriz de caracteres
//pero para que forme una cadena.
//Ahora la cadena debe de terminar con el caracter nulo.

#include <stdio.h>

int main(void){

     char semana[8]={'d','l','m','i','j','v','s',0}; //matriz a llenar
     int i=0; //contador

     printf("Cadena %s\n",semana);

     return(0);
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir la palabra "*Cadena*" seguida de todos los valores de todas las columnas (por orden) de la matriz.

Programa 7.

Para este programa primero abrimos el *main* y declaramos las variables enteras *i,j,k y* la matriz "*semana[7]={'l','m','i','j','v','s','d'}*" ademas del puntero "*dia*" de tipo *char.* Después iniciamos un ciclo *for* definiendo "*j=0*" y bajo la condición de que *j* sea menor que 7 y que cada vez que se repita el ciclo, se le sumará 1 a j. Este ciclo va a imprimir "*El dia es: "* seguido de la letra correspondiente a la matriz. Como es un ciclo se repetirá hasta que "*j=8"* cuando se detendrá y *getchar()* esperará a que o introduzcamos un carácter o que presionemos enter. Después de ello igualamos el puntero dia con la matriz *semana[7]* y lo imprimimos en pantalla mediante *printf*. Después hay un *getchar()* que nuevamente esperara a que introduzcamos un carácter o presionemos enter y después hay otro ciclo *for*, Este ciclo define "*k=1*" y se ejecutara siempre que *k* sea menor a 8 y cada vez que el ciclo se repita se le sumara 1 a la variable k. Este ciclo imprimirá "*La dirección %u es %u*" donde la primer %u es el correspondiente a k+1 y la segunda %u a dia+1.

Al final el código queda dela siguiente manera:

```c
#include <stdio.h>

int main(void){

     char semana[7]={'l','m','i','j','v','s','d'};
     int i, j, k;
     char dia; //declaramos el puntero dia

     for(j=0;j<7;j++){
       printf("El dia es %c\n", semana[j]);
      }

     //Ahora se va a guardar en el puntero dia la direccion de la matriz semana
     dia=semana[7];

     //Note que en la direcciòn enterior no colocamos eloperador deindicaciòn "&"
     //Vamos a imprimir el contenido de dia
     printf("El puntero dia contiene la direccion: %u\n", dia);
     getchar();

     //Ahora vamos a imprimir el resto dedirecciones correspondientes a
     //los elementos siguientes.
     for(k=1;k<8;k++){
       printf("La direccion %u es %u\n",k+1,(dia+1));
      }

     printf("Cadena %s\n",semana);

     return(0);
     getchar();
}
```

Compilamos y ejecutamos el programa y lo que hace es imprimir “El dia es l” luego “El dia es m”, despues “El dia es i” y asi seguidamente hasta acabar con todas las columnas de la matriz. Después se espera a que demos enter e imprime “*El puntero tiene la direccion: 0*”, luego vuelve a esperar a quedemos enter e imprime “La direccion 2 es 1” seguido de “La direccion 3 es 1” y asi seguidamente hasta “La direccion 8 es 1”. Luego imprime la cadena de caracteres que surge de la matriz inicial.

```
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes   Música      Público
Desktop    examples.desktop  matriz6.c  Plantillas  Vídeos
ubuntu@ubuntu:~$ gcc -o matriz matriz6.c
ubuntu@ubuntu:~$ ls
Descargas  Documentos        Imágenes  matriz6.c  Plantillas  Vídeos
Desktop    examples.desktop  matriz    Música     Público
ubuntu@ubuntu:~$ ./matriz
El dia es l
El dia es m
El dia es i
El dia es j
El dia es v
El dia es s
El dia es d
El puntero dia contiene la direccion: 0

La direccion 2 es 1
La direccion 3 es 1
La direccion 4 es 1
La direccion 5 es 1
La direccion 6 es 1
La direccion 7 es 1
La direccion 8 es 1
Cadena lmijvsd
ubuntu@ubuntu:~$
```

Matrices II

Ejercicio 1. Escriba el programa y compilelo.

Este programa utiliza la función "*pow*" de la librería "*math*" y saca cuadrados de números enteros. Para hacer el código del programa, ademas de importar la librería "*stdio.h*" también debemos importa la librería "*math.h*".

Bien, empezamos abriendo nuestro *main* y definimos las variables enteras "*dato*" y "*result*" y las definimos, en principio, como 0. Después, mediante la función *printf* pedimos que se introduzca un número entero y mediante *scanf* recogemos el valor ingresado por el teclado y lo guardamos en la variable *dato*. Después definimos la variable *result* como "*result=pow(dato,2)*", nótese que aquí ocupamos la función *pow* la cual tiene la siguiente sintaxis:

pow(entero, exponente)

por lo que, en nuestro código *pow* queda de la siguiente forma:

pow(dato,2)

es decir, que a al valor que tiene la variable *dato* se elevará al exponente 2 (al cuadrado).
Después mediante *printf* imprimimos en pantalla el resultado y cerramos el código con *return 0* y cerrando las llaves del *main.*

El código queda así:

```c
//Este codigo saca el cuadrado de un numero con la funcion pow

#include <stdio.h>
#include <math.h>

int main(void){

     int dato=0;
     int result=0;

     printf("Introduce un entero: ");
     scanf("%i",&dato);
     result=pow(dato,2);

     printf("El cuadrado de %i es %i.\n",dato, result);

     return(0);

}
```

Compilamos y ejecutamos el programa y lo que hace es pedirte que introduzcas un número entero y después imprime la frase "El cuadrado de x es y" donde "*x*" es el numero entero que nosotros ingresamos y "*y*" es el cuadrado de ese número introducido. En mi ejemplo lo hice con los números 5, 8 y 25. y tal y como se mencionó antes, el programa nos imprime el cuadrado de dichos números respectivamente.

```
ubuntu@ubuntu:~$ ls
A1.c       Desktop     examples.desktop  Música      Público
Descargas  Documentos  Imágenes          Plantillas  Vídeos
ubuntu@ubuntu:~$ gcc -o A A1.c -lm
ubuntu@ubuntu:~$ ls
A     Descargas  Documentos        Imágenes  Plantillas  Vídeos
A1.c  Desktop    examples.desktop  Música    Público
ubuntu@ubuntu:~$ ./A
Introduce un entero: 5
El cuadrado de 5 es 25.
ubuntu@ubuntu:~$ ./A
Introduce un entero: 8
El cuadrado de 8 es 64.
ubuntu@ubuntu:~$ ./A
Introduce un entero: 25
El cuadrado de 25 es 625.
ubuntu@ubuntu:~$
```

Como una observación debo mencionar que por defecto no se enlaza la biblioteca *math.h* a la hora de compilar elprogramay hacer el ejecutable final con *gcc*. Para solucionarlo se debe enlazar dicha biblioteca explícitamente mediante la opción *-lm* bajo la siguiente sintaxis:

gcc *nombre_archivo.c* -lm

En este caso, utilizamos la siguiente sintaxis porque utilice la opción *-o* para cambiarle el nombre determinado del ejecutable final:

gcc -o *nombre_nuevo nombre_archivo.c* -lm

Ejercicio 2. Haga los programas y corralos. (Suma de matrices)
Este ejercicio no me sale, lo copie igual que el que tiene el profesor y no sale, me sale que suma_mat y imprime_mat no están definidas.

El código:

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

void suma_mat(int a[2][2], int b[2][2], int c[2][2]);
void imprime_mat(int C[2][2]);

int main(int argc, char *argv[]){
     int A[2][2]={
       {1,2},
       {3,4}
                 };
    int B[2][2]={
       {2,2},
       {2,2}
                 };

    int C[2][2];

    printf("Hola mundo\n");
    //Sumando matrices

    printf("La matriz A es:\n");
    imprime_mat(A);

    printf("La matriz B es:\n");
    imprime_mat(B);

    //impresion de la suma
    printf("La suma de las matrices es:\n");
    suma_mat(A,B,C);
    imprime_mat(C);

    return EXIT_SUCCESS;

     return(0);

}
```

Y el error que me sale en la terminal:

```
ubuntu@ubuntu:~$ gcc -o A A2.c -lm
/tmp/ccHhseWu.o: En la función `main':
A2.c:(.text+0x76): referencia a `imprime_mat' sin definir
A2.c:(.text+0x8e): referencia a `imprime_mat' sin definir
A2.c:(.text+0xb1): referencia a `suma_mat' sin definir
A2.c:(.text+0xbd): referencia a `imprime_mat' sin definir
collect2: error: ld returned 1 exit status
ubuntu@ubuntu:~$
```