Guía 4

Rodrigo Estay Muñoz

Usuario de Timus: RodrigoEstay

ID de Timus: 247803OV

Problema 1:

n1 = 50, n2 = 15, n3 = 15

Problema 2:

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

/* Codigo hecho sin ayuda*/

double operacion(char op, double v1, double v2); // Declaramos la funcion.

int main(){
    double v1, v2;
    char op;
    printf("Ingrese op, v1 y v2.\n");
    scanf("%c %lf %lf", &op, &v1, &v2);
    while((op!='+' && op!='-' && op!='*' && op!='/') || (op=='/' && v2==0)){ // Validamos la entrada.
        if((op=='/' && v2==0)){
            printf("No se puede dividir por 0. Reingrese todos los valores.\n");
        }
        else{
            printf("Operacion invalida. Reingrese todos los valores.\n");
        }
        scanf("%c %lf %lf", &op, &v1, &v2);
    }
    printf("%lf\n", operacion(op,v1,v2));
    return 0;
}
```

```c
/* Observamos de cual operacion se trata y retornamos el valor esperado. */

double operacion(char op, double v1, double v2){
    if(op=='+'){
        return v1+v2;
    }
    else if(op=='-'){
        return v1-v2;
    }
    else if(op=='*'){
        return v1*v2;
    }
    else if(op=='/'){
        return v1/v2;
    }
}
```

Problema 3:

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

/* Programa hecho sin ayuda. */

long long mcm(long long n1, long long n2);

int main(){
    long long n1, n2, m;
    printf("Ingrese dos numeros:\n");
    scanf("%lld%lld", &n1, &n2);
    while(n1<=0 || n2<=0){
        printf("Entrada invalida. Porfavor compruebe que los numeros sean naturales distintos de 0.\n");
        scanf("%lld%lld", &n1, &n2);
    }
    m=mcm(n1, n2);
    printf("El minimo comun multiplo es: %lld\n", m);
}
```

```c
/* Primero hacemos que el "n1" siempre sea el mayor, luego vamos avanzando de a una unidad desde "n1",
hasta encontrar un numero que sea divisible por "n1" y "n2", luego retornamos ese numero. */

long long mcm(long long n1, long long n2){
    long long temp, i;
    if(n2>n1){
        temp=n1;
        n1=n2;
        n2=temp;
    }
    for(i=n1;i%n1!=0 || i%n2!=0;++i){}
    return i;
}
```

Problema 4:

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

/* Programa hecho sin ayuda. */

long long mcd(long long n1, long long n2);

int main(){
    long long n1, n2, m;
    printf("Ingrese dos numeros:\n");
    scanf("%lld%lld", &n1, &n2);
    while(n1==0 || n2==0){
        printf("Entrada invalida. Porfavor compruebe que los numeros sean enteros distintos de 0.\n");
        scanf("%lld%lld", &n1, &n2);
    }
    m=mcd(n1, n2);
    if(m){
        printf("El maximo comun divisor es: %lld\n", m);
    }
    else{
        printf("No tienen maximo comun divisor.\n");
    }
    return 0;
}
```

```c
/* Primero hacemos que "n1" siempre sea el menor. Luego buscamos un numero menor a "n1" tal que
cumpla que "n1" y "n2" sean divisibles por este numero, luego retornamos dicho numero.
Notar que se ocupo un while en vez de un for, ya que asi evitamos de que cuando "i" llegue
a 0 se realizen los modulos "n1%i" y "n2%i" lo cual dividiria por 0, lo cual mataria el
programa. */

long long mcd(long long n1, long long n2){
    long long temp, i;
    if(n1>n2){
        temp=n2;
        n2=n1;
        n1=temp;
    }
    i=n1;
    while(1){
        if(i==0){
            break;
        }
        if(n1%i==0 && n2%i==0){
            break;
        }
        --i;
    }
    return i;
}
```

Problema 5:

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

/* Programa hecho sin ayuda. */

int programCountinue();
double operacion(char op, double v1, double v2);
long long mcm(long long n1, long long n2);
long long mcd(long long n1, long long n2);

int main(){
    double v1, v2;
    long long n1, n2;
    char user1[100], op, c; // "user1" es la primera eleccion del usuario.
    int user2; // "user2" es la segunda eleccion del usario.
    while(1){ // Este while es para imprimir el menu denuevo si se da una entrada invalida.
        system("clear");
        user2=0; // Hacemos que sea 0, para que al entrar a algun if no se eecute algo no deseado.
        printf("Bienvenido. Seleccione la operacion a realizar:\n"
        "operacion\t- Realizar suma, resta, multiplicacion o division entre dos valores.\n"
        "mcm\t- Calcular el minimo comun multiplo de dos numeros.\n"
        "mcd\t- Calcular el maximo comun divisor de dos numeros.\n"
        "salir\t- Salir del programa.\n");
        scanf("%s", user1);

        /* Comparamos lo escrito por el usuario con cada posible opcion y ejecutamos lo que hace,
        si no volvemos a iprimir el menu. */

```

```c
        /* Comparamos lo escrito por el usuario con cada posible opcion y ejecutamos lo que hace,
        si no volvemos a iprimir el menu. */

        if(!strcmp(user1,"operacion")){
            while(1){ // Este while es para imprimir el submenu denuevo.
                system("clear");
                getchar();
                printf("Operacion\n\n"
                "En esta funcion debe ingresar una operacion y 2 valores cualquiera, siempre y"
                "cuando se le puedan aplicar la operacion escogida, la operacion y los valores deben"
                "estar separados por un espacio. Las operaciones aceptadas son:\n"
                "+ para sumar.\n- para restar\n* para multiplicar\n/ para dividir.\n"
                "Note que la operacion se realiza de la manera (primer valor) (operacion) (segundo valor).\n\n"
                "1- Ingresar operacion.\n"
                "2- Volver al menu principal.\n");
                scanf("%d", &user2);
                if(user2==1){
                    printf("Ingrese los valores como se especifico anteriormente.\n");
                    getchar();
                    scanf("%c %lf %lf", &op, &v1, &v2);
                    // Validamos la entrada.
                    while((op!='+' && op!='-' && op!='*' && op!='/') || (op=='/' && v2==0)){
                        if((op=='/' && v2==0)){
                            printf("No se puede dividir por 0. Reingrese todos los valores.\n");
                        }
                        else{
                            printf("Operacion invalida. Reingrese todos los valores.\n");
                        }
                        scanf("%c %lf %lf", &op, &v1, &v2);
```

```
                    scanf("%c %lf %lf", &op, &v1, &v2);
                }
                printf("%.3lf %c %.3lf = %.3lf\n", v1, op, v2, operacion(op,v1,v2));
                if(programCountinue()==1){
                    break;
                }
                else{
                    system("clear");
                    return 0;
                }
                system("clear");
                break;
            }
            else if(user2==2){
                break;
            }
            else{ // Para entrada invalida, limpiamos la pantalla y el while imprimira el submenu denuevo.
                system("clear");
            }
        }
    }
    else if(!strcmp(user1,"mcm")){
        while(1){ // Este while es para imprimir el submenu denuevo.
            system("clear");
            getchar();
            printf("Minimo comun multiplo\n\n"
            "En esta funcion debe ingresar, separados por un espacio, dos numeros naturales, "
            "excluyendo el 0, y luego se le entregara el valor del minimo comun multiplo de "
```

```
            "En esta funcion debe ingresar, separados por un espacio, dos numeros naturales, "
            "excluyendo el 0, y luego se le entregara el valor del minimo comun multiplo de "
            "los numeros ingresados.\n\n"
            "1- Ingresar operacion.\n"
            "2- Volver al menu principal.\n");
            scanf("%d", &user2);
            if(user2==1){
                printf("Ingrese los valores como se especifico anteriormente.\n");
                scanf("%lld%lld", &n1, &n2);
                // Validamos la entrada.
                while(n1<=0 || n2<=0){
                    printf("Entrada invalida. Porfavor compruebe que los numeros sean naturales distintos de 0
                    scanf("%lld%lld", &n1, &n2);
                }
                printf("El minimo comun multiplo de %lld y %lld es %lld\n", n1, n2, mcm(n1,n2));
                if(programCountinue()==1){
                    break;
                }
                else{
                    system("clear");
                    return 0;
                }
                system("clear");
                break;
            }
            else if(user2==2){
                break;
            }
            else{ // Para entrada invalida, limpiamos la pantalla y el while imprimira el submenu denuevo.
                system("clear");
            }
```

```
                }
            }
        }
        else if(!strcmp(user1,"mcd")){
            while(1){ // Este while es para imprimir el submenu denuevo.
                system("clear");
                getchar();
                printf("Maximo comun divisor\n\n"
                "En esta funcion debe ingresar, separados por un espacio, dos numeros enteros"
                "distintos de 0, y luego se le entregara el valor del maximo comun divisor de los numeros "
                "ingresados.\n\n"
                "1- Ingresar operacion.\n"
                "2- Volver al menu principal.\n");
                scanf("%d", &user2);
                if(user2==1){
                    printf("Ingrese los valores como se especifico anteriormente.\n");
                    scanf("%lld%lld", &n1, &n2);
                    // Validamos la entrada.
                    while(n1<=0 || n2<=0){
                        printf("Entrada invalida. Porfavor compruebe que los numeros sean enteros distintos de 0.\n")
                        scanf("%lld%lld", &n1, &n2);
                    }
                    printf("El maximo comun divisor de %lld y %lld es %lld\n", n1, n2, mcd(n1,n2));
                    if(programCountinue()==1){
                        break;
                    }
                    else{
                        system("clear");
                        return 0;
                    }
                    system("clear");
```

```
                    system("clear");
                    break;
                }
                else if(user2==2){
                    break;
                }
                else{ // Para entrada invalida, limpiamos la pantalla y el while imprimira el submenu denuevo.
                    system("clear");
                }
            }
        }
        // Si se escoje salir, salimos del while, lo cual nos lleva al return 0;
        else if(!strcmp(user1,"salir")){
            system("clear");
            return 0;
        }
        else{ // Para entrada invalida, limpiamos la pantalla y el while imprimira el submenu denuevo.
            system("clear");
        }
    }
}

/* Funcion nueva creada con el objetivo de comprobar si el usuario desea seguir utilizando el
programa despues de terminar de usar algunas de sus funciones. */

int programCountinue(){
    int des;
    printf("\nDesea continuar usando el programa?\n1- Si\n2- No\n");
    while(1){
        scanf("%d", &des);
        if(des==1 || des==2){
```

```
171         if(des==1 || des==2){
172             return des;
173         }
174         else{
175             printf("Entrada invalida, intente denuevo...\n");
176         }
177     }
178 }
179 
180 /* Estas son las funciones explicadas previamente. */
181 
182 double operacion(char op, double v1, double v2){
183     if(op=='+'){
184         return v1+v2;
185     }
186     else if(op=='-'){
187         return v1-v2;
188     }
189     else if(op=='*'){
190         return v1*v2;
191     }
192     else if(op=='/'){
193         return v1/v2;
194     }
195 }
196 
197 long long mcm(long long n1, long long n2){
198     long long temp, i;
199     if(n2>n1){
200         temp=n1;
201         n1=n2;
```

```
201         n1=n2;
202         n2=temp;
203     }
204     for(i=n1;i%n1!=0 || i%n2!=0;++i){}
205     return i;
206 }
207 
208 long long mcd(long long n1, long long n2){
209     long long temp, i;
210     if(n1>n2){
211         temp=n2;
212         n2=n1;
213         n1=temp;
214     }
215     i=n1;
216     while(1){
217         if(i==0){
218             break;
219         }
220         if(n1%i==0 && n2%i==0){
221             break;
222         }
223         --i;
224     }
225     return i;
226 }
```

Problema 6:

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

/* Programa hecho sin ayuda. */

void ordenParcial(int* v1, int n, int cant);

/* En main solo esta hecho para probar si la funcion funciona como se espera. */

int main(){
    int n;
    scanf("%d", &n);
    int v1[n], i, cant;
    for(i=0;i<n;++i){
        scanf("%d", &v1[i]);
    }
    scanf("%d", &cant);
    for(i=0;i<n;++i){
        printf("%d ", v1[i]);
    }
    printf("\n");
    ordenParcial(v1, n, cant);
    for(i=0;i<n;++i){
        printf("%d ", v1[i]);
    }
    printf("\n");
    return 0;
}
```

```c

/* Primero ordenamos el vector de mayor a menor, luego para los primeros "cant" valores del vector,
inviertiremos sus posiciones, asi haciendo que los primeros "cant" valores sean los mayores del arreglo
y al invertirlos conseguimos que esten ordenados en orden ascendiente (de menor a mayor). */

void ordenParcial(int* v1, int n, int cant){
    int i, j, temp;
    for(i=0;i<n-1;++i){
        for(j=i;j<n;++j){
            if(v1[i]<v1[j]){
                temp=v1[j];
                v1[j]=v1[i];
                v1[i]=temp;
            }
        }
    }
    for(i=0;i<cant/2;++i){
        temp=v1[i];
        v1[i]=v1[cant-i-1];
        v1[cant-i-1]=temp;
    }
}
```

Problema 7:

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

/* El ayudante ma ayudo con la existencia de limts.h y el paso de matrices entre
funciones. */

int determinante(int n, int m1[n][n]);

int main(){ // este main es solo para probar la funcion.
    int n;
    scanf("%d", &n);
    int m1[n][n], i, j, det;
    for(i=0;i<n;++i){
        for(j=0;j<n;++j){
            scanf("%d", &m1[i][j]);
        }
    }
    det=determinante(n,m1);
    printf("%d\n", det);
    return 0;
}
```

```c
/* Calculamos las matrices para cuando n={2,3} con el metodo del triangulo, y cuando n=4
lo haceos por submatrices, en caso que "n" no es ninguno de estos valores, entonces hacemos
que retorne INT_MIN la cual es una constante definida por limits.h que tiene el valor minimo
que puede tomar int. */

int determinante(int n, int m1[n][n]){
    int det;
    if(n==2){
        det=m1[0][0]*m1[1][1]-m1[1][0]*m1[0][1];
    }
    else if(n==3){
        det=m1[0][0]*m1[1][1]*m1[2][2]+m1[0][1]*m1[1][2]*m1[2][0]
           +m1[0][2]*m1[1][0]*m1[2][1]-m1[2][0]*m1[1][1]*m1[0][2]
           -m1[2][1]*m1[1][2]*m1[0][0]-m1[2][2]*m1[1][0]*m1[0][1];
    }
    else if(n==4){
        int subDet1, subDet2, subDet3, subDet4;
        subDet1=m1[1][1]*m1[2][2]*m1[3][3]+m1[1][2]*m1[2][3]*m1[3][1]
               +m1[1][3]*m1[2][1]*m1[3][2]-m1[1][3]*m1[2][2]*m1[3][1]
               -m1[1][1]*m1[2][3]*m1[3][2]-m1[1][2]*m1[2][1]*m1[3][3];
        subDet2=m1[1][0]*m1[2][2]*m1[3][3]+m1[1][2]*m1[2][3]*m1[3][0]
               +m1[1][3]*m1[2][0]*m1[3][2]-m1[1][3]*m1[2][2]*m1[3][0]
               -m1[1][0]*m1[2][3]*m1[3][2]-m1[1][2]*m1[2][0]*m1[3][3];
        subDet3=m1[1][0]*m1[2][1]*m1[3][3]+m1[1][1]*m1[2][3]*m1[3][0]
               +m1[1][3]*m1[2][0]*m1[3][1]-m1[1][3]*m1[2][1]*m1[3][0]
               -m1[1][1]*m1[2][0]*m1[3][3]-m1[1][0]*m1[2][3]*m1[3][1];
        subDet4=m1[1][0]*m1[2][1]*m1[3][2]+m1[1][1]*m1[2][2]*m1[3][0]
               +m1[1][2]*m1[2][0]*m1[3][1]-m1[1][2]*m1[2][1]*m1[3][0]
               -m1[1][1]*m1[2][0]*m1[3][2]-m1[1][0]*m1[2][2]*m1[3][1];
        det=m1[0][0]*subDet1-m1[0][1]*subDet2+m1[0][2]*subDet3-m1[0][3]*subDet4;
```

```
52                  -m1[1][1]*m1[2][0]*m1[3][2]-m1[1][0]*m1[2][2]*m1[3][1];
53          det=m1[0][0]*subDet1-m1[0][1]*subDet2+m1[0][2]*subDet3-m1[0][3]*subDet4;
54      }
55      else{
56          det=INT_MIN;
57      }
58      return det;
59  }
```

Problema 8:

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

// Programa hecho sin ayuda

int caracteresComunes(char* cadena1, char* cadena2);

int main(){ // main es para probar la funcion.
    char cadena1[1000], cadena2[1000];
    printf("Introduzca dos palabras separadas por un espacio:\n");
    scanf("%s%s", cadena1, cadena2);
    printf("%d caracteres comunes\n", caracteresComunes(cadena1,cadena2));
    return 0;
}
```

```c
int caracteresComunes(char* cadena1, char* cadena2){
    char comunes[260]; // En esta cadena se almacenaran los caracteres comunes encontrados.
    int i, j, k, contComunes=0, esNuevo;
    int n1=strlen(cadena1), n2=strlen(cadena2);
    // Para cada caracter de la cadena1 buscamos si tiene un igual en la cadena2.
    for(i=0;i<n1;++i){
        for(j=0;j<n2;++j){
            // Si se encuentra un caracter nuevo, revisamos si es que se ha encontrado antes.
            if(*(cadena1+i)==*(cadena2+j)){
                esNuevo=1;
                for(k=0;k<contComunes;++k){
                    if(*(cadena1+i)==*(comunes+k)){
                        esNuevo=0;
                        break;
                    }
                }
                // Si nunca se ha encontrado ese caracter, significa que es nuevo, asi que lo almacenamos
                // y aumentamos el contador de caracteres comunes.
                if(esNuevo){
                    *(comunes+contComunes)=*(cadena1+i);
                    ++contComunes;
                }
                break;
            }
        }
    }
    return contComunes; // Retornamos el contador de caracteres comunes.
}
```

Problema 9:

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

// Programa hecho sin ayuda.

char palindromo(char* cadena1);

int main(){ // main es para revisar el funcionamiento de la funcion.
    char cadena1[1000];
    printf("Ingrese la palabra a ser analizada:\n");
    scanf("%s", cadena1);
    if(palindromo(cadena1)){
        printf("Es palindromo.\n");
    }
    else{
        printf("No es palindromo.\n");
    }
    return 0;
}
```

```c
char palindromo(char* cadena1){
    int n=strlen(cadena1), i; // Primero obtenemos el largo de la palabra.
    /* Luego revisamos para la primera mitad de los caracteres de la cadena si son iguales a los
    caracteres del otro lado de la palabra, si no lo son altiro devolvemos el 0, pero si lo son
    se va a terminar el ciclo y se devolvera un 1. */
    for(i=0;i<n/2;++i){
        if(*(cadena1+i)!=*(cadena1+(n-1-i))){
            return 0;
        }
    }
    return 1;
}
```

Problema 10:

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

void modificarMatriz(int n, int m, int mat1[n][m]);

/* main es para probar la funcion. */

int main(){
    int n, m, i, j;
    printf("De que dimensiones es su matriz? (nxm)\n");
    scanf("%d %d", &n, &m);
    int mat[n][m];
    printf("Ingrese los %d valores\n", n*m);
    for(i=0;i<n;++i){
        for(j=0;j<m;++j){
            scanf("%d", &mat[i][j]);
        }
    }
    printf("Cambiando valores...\n");
    modificarMatriz(n,m,mat);
    for(i=0;i<n;++i){
        for(j=0;j<m;++j){
            printf("%d\t", mat[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

```c
/* Simplemente recorremos la matriz cambiando los 1 por -1. */

void modificarMatriz(int n, int m, int mat1[n][m]){
    int i, j;
    for(i=0;i<n;++i){
        for(j=0;j<m;++j){
            if(mat1[i][j]==1){
                mat1[i][j]=-1;
            }
        }
    }
}
```

Problema 11:

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

// Programa hecho sin ayuda.

int longitudPalabraMasLarga(int n, char crucigrama[n][n]);

int main(){ // Main para revisar la funcion de la funcion.
    int n, i, j;
    printf("Tamaño?\n");
    scanf("%d", &n);
    getchar();
    printf("Ingrese los %d valores\n", n*n);
    char mat[n][n];
    for(i=0;i<n;++i){ // Escaneamos los caracteres ignarando los saltos de linea.
        for(j=0;j<n;++j){
            scanf("%c", &mat[i][j]);
            if(mat[i][j]=='\n'){
                --j;
                continue;
            }
        }
    }
    for(i=0;i<n;++i){
        for(j=0;j<n;++j){
            printf("%c", mat[i][j]);
        }
        printf("\n");
    }
    printf("la palabra mas larga es de %d letras.\n", longitudPalabraMasLarga(n,mat));
    return 0;
}
```

```c
int longitudPalabraMasLarga(int n, char crucigrama[n][n]){
    int i, j, cont1, cont2, maxLen=0; // Asumimos que la palabra mas larga es de 0 caracteres.
    // Estos ciclos revisan a la vez el largo de las palabra tantos horizontales como verticales.
    for(i=0;i<n;++i){
        // Reiniciamos los contadores.
        cont1=0;
        cont2=0;
        for(j=0;j<n;++j){
            // Revisa el largo de la palabra horizontal al encontrarse con un - y si es mas largo
            // que el maximo largo registrado, se guarda como el nuevo largo maximo. Luego se reinicia
            // el contador horizontal.
            if(crucigrama[i][j]=='-'){
                if(cont1>maxLen){
                    maxLen=cont1;
                }
                cont1=0;
            }
            else{
                ++cont1; // Para cada caracter distinto de - sumamos el contador horizontal.
            }
            // Revisa el largo de la palabra vertical al encontrarse con un - y si es mas largo
            // que el maximo largo registrado, se guarda como el nuevo largo maximo. Luego se reinicia
            // el contador vertical.
            if(crucigrama[j][i]=='-'){
                if(cont2>maxLen){
                    maxLen=cont2;
                }
                cont2=0;
            }
```

```
            }
            else{
                ++cont2; // Para cada caracter distinto de - sumamos el contador horizontal.
            }
        }
        // Revisamos el largo de las palabras, ya que al salir del ciclo se trata de las palabras
        // que llegaban hasta el final de la matriz, actualizamos el largo maximo si es necesario.
        if(cont1>maxLen){
            maxLen=cont1;
        }
        if(cont2>maxLen){
            maxLen=cont2;
        }
    }
    return maxLen;
}
```