Comenzado el	viernes, 2 de noviembre de 2018, 11:27
Estado	Finalizado
Finalizado en	jueves, 15 de noviembre de 2018, 19:02
Tiempo empleado	13 días 7 horas
Calificación	Sin calificar aún

Pregunta 1

Correcta

Puntúa 15 sobre 15

Codifique la definición de una función cuyo prototipo es:

void cargarDistanciasCiudades(double distancias[N][N])

La función recibe como argumento una matriz de NxN que representa las distancias en kilómetros entre las ciudades denominadas 0, 1, ..., y N-1. Por lo tanto, el elemento en la fila i y columna j representa la distancia desde la ciudad i hasta la ciudad j. La función debe solicitar al usuario ingresar las distancias entre ciudades por teclado y almacenarlas en la matriz.

- No necesariamente la distancia entre dos ciudades debe ser la misma desde una de ellas o desde la otra, por lo que se debe ingresar la distancia tanto desde la ciudad i a la ciudad j, como también desde la ciudad j hasta la ciudad i.
- La función NO DEBE solicitar la distancia entre una ciudad y la misma ciudad. Es decir, no debe solicitar la distancia desde la ciudad i a la ciudad i, y debe asignar directamente el valor de distancia 0 a esos casos.
- En el caso que se ingrese una distancia menor o igual a cero, la función deberá indicar mediante un mensaje que el valor no es válido y pedir que se ingrese nuevamente hasta que el mismo sea mayor que cero.

Se recomienda ver el ejemplo de ejecución del programa para comprender como la función pide ingresar los valores de distancia.

Respuesta: (penalty regime: 0 %)

Reiniciar respuesta

```
void cargarDistanciasCiudades(double distancias[N][N])
{
    for(int i=0;i<N;i++)
    {
        for(int j=0;j<N;j++)
        {
            if(j==i)
            {
                distancias[i][i]=0;
            }
            else
            {
                do
                {
                cin>>distancias[i][j];
                }
                while(distancias[i][j]<=0);

            }
        }
```

	Test	Input	Expected	Got	

	Test	Input	Expected	Got	

✓	`double distancias[N][N];`
	`cargarDistanciasCiudades(distancias);`
	`for(int i=0;i<N;i++)`
	`  for(int j=0;j<N;j++)`
	`  cout<<distancias[i][j]<<endl;`
		1
		2
		3
		4
		5
		6
			0
			1
			2
			3
			0
			4
			5
			6
			0
				0
				1
				2
				3
				0
				4
				5
				6
				0
					✓

✓	`double distancias[N][N];`
	`cargarDistanciasCiudades(distancias);`
	`for(int i=0;i<N;i++)`
	`  for(int j=0;j<N;j++)`
	`  cout<<distancias[i][j]<<endl;`
		1
		-2
		2
		3
		4
		5
		6
			0
			1
			2
			3
			0
			4
			5
			6
			0
				0
				1
				2
				3
				0
				4
				5
				6
				0
					✓

✓	`double distancias[N][N];`
	`cargarDistanciasCiudades(distancias);`
	`for(int i=0;i<N;i++)`
	`  for(int j=0;j<N;j++)`
	`  cout<<distancias[i][j]<<endl;`
		1
		-2
		-2
		2
		3
		4
		5
		6
			0
			1
			2
			3
			0
			4
			5
			6
			0
				0
				1
				2
				3
				0
				4
				5
				6
				0
					✓

✓	`double distancias[N][N];`
	`cargarDistanciasCiudades(distancias);`
	`for(int i=0;i<N;i++)`
	`  for(int j=0;j<N;j++)`
	`  cout<<distancias[i][j]<<endl;`
			0	0	✓

✓	`double distancias[N][N];`
	`cargarDistanciasCiudades(distancias);`
	`for(int i=0;i<N;i++)`
	`  for(int j=0;j<N;j++)`
	`  cout<<distancias[i][j]<<endl;`
		0
		0
		1
		1
				0
				1
				1
				0
					✓

Todas las pruebas superadas. ✓

Question author's solution:

```
void cargarDistanciasCiudades(double distancias[N][N])
{
    for(int i=0; i<N; i++)
        for(int j=0; j<N; j++)
        {
            if(i == j)
                distancias[i][j] = 0;
            else
            {
                cout << "Ingrese la distancia en kms desde la ciu
dad " << i << " hasta la ciudad " << j << ": ";
                // La distancia entre dos ciudades puede variar /
ser diferente segun el sentido / direccion
                cin >> distancias[i][j];
                while(distancias[i][j] <=0 )
                {
                    cout << "Error: la distancia entre dos ciudad
es debe ser mayor a 0" << endl;
                    cout << "Reingrese la distancia en kms desde
la ciudad " << i << " hasta la ciudad " << j << ": ";
                    cin >> distancias[i][j];
                }
            }
        }
}
```

Correcta

Puntos para este envío: 15/15.

Pregunta 2

Correcta

Puntúa 10 sobre 10

Codifique la definición de una función cuyo prototipo es:

void mostrarDistanciasCiudades(double distancias[N][N])

La función debe mostrar las distancias desde cada ciudad hasta las otras ciudades en el formato que se muestra en el ejemplo de ejecución abajo. No debe mostrar la distancia consigo misma que se entiende es cero.

Ejemplo (N=3)

Desde 0 hasta:
1: 1.2 kms
2: 3.4 kms
Desde 1 hasta:
0: 9 kms
2: 2.3 kms
Desde 2 hasta:
0: 7.3 kms
1: 0.4 kms

Respuesta: (penalty regime: 0 %)

Reiniciar respuesta

```
1  void mostrarDistanciasCiudades(double distancias[N][N])
2  {
3     //La función muestra las distancias desde cada ciudad hasta las o
4     //distancia consigo misma que es cero.
5     for(int i=0;i<N;i++)
6     {
7         cout<<"Desde "<< i<<" hasta: "<<endl;
8         for(int j=0;j<N;j++)
9         {
10            if(i!=j)
11            {
12            cout<<j<<": "<< distancias[i][j]<< " kms"<< endl;
13            }
14        }
15    }
16
17     return;
18 }
```

	Test	Expected	Go

	Test	Expected	Got

✓	`double distancias[N][N]={0,1,2,3,0,4,5,6,0};`
	`mostrarDistanciasCiudades(distancias);`
		`Desde 0 hasta:`
		`1: 1 kms`
		`2: 2 kms`
		`Desde 1 hasta:`
		`0: 3 kms`
		`2: 4 kms`
		`Desde 2 hasta:`
		`0: 5 kms`
		`1: 6 kms`
			`De`
			`1:`
			`2:`
			`De`
			`0:`
			`2:`
			`De`
			`0:`
			`1:`

✓	`double distancias[N][N]={0,1,1,0};`
	`mostrarDistanciasCiudades(distancias);`
		`Desde 0 hasta:`
		`1: 1 kms`
		`Desde 1 hasta:`
		`0: 1 kms`
			`De`
			`1:`
			`De`
			`0:`

Todas las pruebas superadas. ✓

Question author's solution:

```
void mostrarDistanciasCiudades(double distancias[N][N])
{
    for(int i=0; i<N; i++)
    {
        cout << "Desde " << i << " hasta: ";
        for(int j=0; j<N; j++)
            if (i != j)
                cout << endl << j << ": " << distancias[i][j] <<
" kms";
        cout << endl;
    }
}
```

Correcta

Puntos para este envío: 10/10.

Pregunta 3

Correcta

Puntúa 10 sobre 10

Codifique la definición de una función cuyo prototipo es:

int ingresarOrigenValido()

La función devuelve un valor entero que se ingresa por teclado y que corresponde a una de las N ciudades (0, 1, ..., N-1). En el caso de ingresar un valor no valido, es decir menor que cero o mayor o igual a N, se debe volver a solicitar un nuevo valor hasta que el mismo sea válido.

Respuesta: (penalty regime: 0 %)

Reiniciar respuesta

```
int ingresarOrigenValido()
{
   // Devuelve un entero que se ingresa y que corresponde a una de l
   //En el caso de ingresar un valor menor que cero o mayor o igual
   //valor hasta que el mismo sea válido.
   int n;

   do
   {
       cin>>n;
   }
   while(n<0||n>=N);
   return n;
}
```

	Test	Input	Expected	Got	
✓	`cout << ingresarOrigenValido();`	`0`	`0`	`0`	✓
✓	`cout << ingresarOrigenValido();`	`-1` `0`	`0`	`0`	✓

Todas las pruebas superadas. ✓

Question author's solution:

```
int ingresarOrigenValido()
{
    int origen;
    cout << "Ingrese ciudad de origen: ";
    cin >> origen;
    while( origen < 0 || origen >= N)
    {
        cout << "La ciudad " << origen << " no es valida, ingrese
un valor valido entre 0 y " << N << ": ";
        cin >> origen;
    }
    return origen;
}
```

Correcta

Puntos para este envío: 10/10.

Pregunta 4

Correcta

Puntúa 10 sobre 10

Codifique la definición de una función cuyo prototipo es:

int destinoMasLejano (double distancias[N][N], int origen)

La función devuelve un valor entero que indica la ciudad más distante a una ciudad de origen indicada como argumento de la función. EN el caso que dos o más ciudades se encuentren a la misma distancia que la ciudad origen, la función debe devolver aquella con el identificador entero menor.

Respuesta: (penalty regime: 0 %)

Reiniciar respuesta

```
int destinoMasLejano (double distancias[N][N], int origen)
{
    //devuelve un entero que indica la ciudad más distante a una ciu
    //argumento de la función. EN el caso que dos o más ciudades se
    //distancia que la ciudad origen, la función devuelve aquella co
    double max=distancias[origen][0];
    int k;

    for(int j=0;j<N;j++)
        {
         if(max<distancias[origen][j])
          {
             max=distancias[origen][j];
             k=j;
          }
        }

    return k;
}
```

	Test	Expected	Got	
✓	`double dist[N][N]={0,1,2,0};` `cout << destinoMasLejano(dist, 0)`	1	1	✓
✓	`double dist[N][N]={0,1,2,0};` `cout << destinoMasLejano(dist, 1)`	0	2107204592	✓
✓	`double dist[N][N]={0,1,2,3,0,4,5,6,0};` `cout << destinoMasLejano(dist, 0)`	2	2	✓
✓	`double dist[N][N]={0,1,2,3,0,4,5,5,0};` `cout << destinoMasLejano(dist, 2)`	0	6295032	✓

Todas las pruebas superadas. ✓

Question author's solution:

```
int destinoMasLejano (double distancias[N][N], int origen)
{
    int destino = origen;
    double distancia = 0;
    for(int i=0; i<N; i++)
            if(distancias[origen][i] > distancia)
            {
                destino = i;
                distancia = distancias[origen][i];
            }

    return destino;
}
```

Correcta

Puntos para este envío: 10/10.

Pregunta 5

Correcta

Puntúa 15 sobre 15

Codifique la definición de una función cuyo prototipo es:

double calcularRecorrido(double distancias[N][N], int origen)

La función devuelve la suma total de kilómetros resultantes de recorrer desde una ciudad origen (indicado como argumento) y una serie de ciudades que el usuario ingresará en orden a medida que la función lo solicite. La función deberá solicitar se ingrese una próxima ciudad a recorrer hasta que el usuario ingrese un valor de ciudad inválido, es decir menor a cero o mayor o igual a N. El recorrido puede repetir ciudades y no necesariamente debe terminar en la ciudad donde se comenzó. Ver el ejemplo de ejecución.

Respuesta: (penalty regime: 0 %)

Reiniciar respuesta

```
double calcularRecorrido(double distancias[N][N], int origen)
{
    double sum=0.0;
    int i, j;
    i=origen;
     cin>>j;
            while(j>=0 && j<N)
                {
                 sum+=distancias[i][j];
                 i=j;
                 cin>>j;
                }

    return sum;
}
```

	Test	Input	Expected	Got
✓	`double dist[N][N]={0,1.25,2,0};` `cout << calcularRecorrido(dist, 0)`	1 -1	1.25	1.25
✓	`double dist[N][N]={0,1,2.73,0};` `cout << calcularRecorrido(dist, 1)`	0 2.73	2	2.73
✓	`double dist[N][N]={0,1.3,2,3,0,4.2,5,6,0};` `cout << calcularRecorrido(dist, 0)`	1 2 -1	5.5	5.5
✓	`double dist[N][N]={0,1,2,3.3,0,4,5,6.1,0};` `cout << calcularRecorrido(dist, 2)`	1 0 3	9.4	9.4

Todas las pruebas superadas. ✓

Question author's solution:

```
double calcularRecorrido(double distancias[N][N], int origen)
{
    int proxima;

    double suma = 0;
    int anterior = origen;

    cout << "Recorridos 0kms. Ingrese proxima ciudad a visitar de
sde la ciudad " << anterior << ": ";
    cin >> proxima;
    while (proxima >= 0 && proxima < N)
    {
        suma += distancias[anterior][proxima];
        anterior = proxima;
        cout << "Recorridos " << suma << "kms. Ingrese proxima ci
udad a visitar desde la ciudad " << anterior << ": ";
        cin >> proxima;
    }

    return suma;
}
```

Correcta

Puntos para este envío: 15/15.

Pregunta 6

Finalizado

Puntúa como 40

Realizar un programa en C++ que incluya las siguientes funciones:

1) void cargarDistanciasCiudades(double distancias[N][N])

2) void mostrarDistanciasCiudades(double distancias[N][N])

3) int ingresarOrigenValido()

4) int destinoMasLejano (double distancias[N][N], int origen)

5) double calcularRecorrido(double distancias[N][N], int origen)

EJEMPLO DE EJECUCION PARA N=3

Ingrese la distancia en kms desde la ciudad 0 hasta la ciudad 1: 1.2

Ingrese la distancia en kms desde la ciudad 0 hasta la ciudad 2: 3.4

Ingrese la distancia en kms desde la ciudad 1 hasta la ciudad 0: 0

Error: la distancia entre dos ciudades debe ser mayor a 0

Reingrese la distancia en kms desde la ciudad 1 hasta la ciudad 0: -1

Error: la distancia entre dos ciudades debe ser mayor a 0

Reingrese la distancia en kms desde la ciudad 1 hasta la ciudad 0: 9

Ingrese la distancia en kms desde la ciudad 1 hasta la ciudad 2: 2.3

Ingrese la distancia en kms desde la ciudad 2 hasta la ciudad 0: 7.3

Ingrese la distancia en kms desde la ciudad 2 hasta la ciudad 1: 0.4

Desde 0 hasta:

1: 1.2 kms

2: 3.4 kms

Desde 1 hasta:

0: 9 kms

2: 2.3 kms

Desde 2 hasta:

0: 7.3 kms

1: 0.4 kms

Ingrese ciudad de origen: 3

La ciudad 3 no es valida, ingrese un valor valido entre 0 y 3: -1

La ciudad -1 no es valida, ingrese un valor valido entre 0 y 3: 2

La ciudad mas lejana desde este origen es: 0

Recorridos 0kms. Ingrese proxima ciudad a visitar desde la ciudad 2: 0

Recorridos 7.3kms. Ingrese proxima ciudad a visitar desde la ciudad 0: 1

Recorridos 8.5kms. Ingrese proxima ciudad a visitar desde la ciudad 1: 2

Recorridos 10.8kms. Ingrese proxima ciudad a visitar desde la ciudad 2: 0

Recorridos 18.1kms. Ingrese proxima ciudad a visitar desde la ciudad 0: -1

La distancia total del recorrido es : 18.1kms

```
#include <iostream>
using namespace std;

const int N=3;

 void cargarDistanciasCiudades(double distancias[N][N])
{
   for(int i=0;i<N;i++)
   {
      for(int j=0;j<N;j++)
      {
         if(j==i)
         {
            distancias[i][i]=0;
         }
```

◄ Ejemplo EA - 9 Ir a... Ejemplo EA - 11 (CR) ►