Universidad Nacional Abierta
Vicerrectorado Académico
Área de Ingeniería
Carrera Ingeniería de Sistemas

Trabajo Práctico

Asignatura: Computación II Código: 324

Fecha de devolución: A más tardar el 17/04/2021 (Sin prórroga)

Nombre del Estudiante: Glemand A Pineda C

Cédula de Identidad: 9347512

Centro Local / Unidad de Apoyo: Táchira

Correo electrónico: glempineda@gmail.com

Teléfono celular: 04169115964

Carrera: Ing Sistemas 326

Número de originales: 1

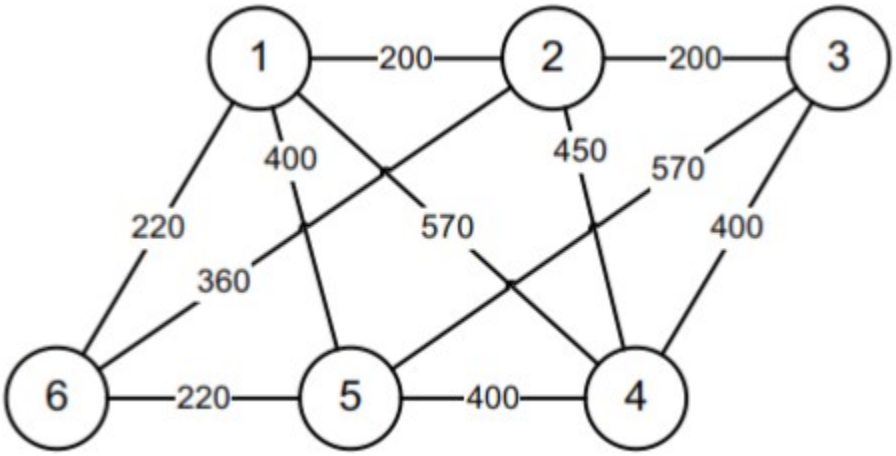
Lapso: 2021-1

RESULTADOS DE CORRECCIÓN:

OBJ N°		5	6	7
0:NL	1:L			

Objetivo 5

1- Aplicar el algoritmo de Dijkstra a la red de la figura 1, para encontrar las rutas de coste mínimo entre 1 y el resto de los nodos de la red.



OBJETIVO 6

2- Elabore un programa en C++ que realice lo siguiente:

En la empresa Supersoft S.A, se necesita un programa que permita ordenar de mayor a menor o de menor a mayor un array o arreglo con los sueldos de los N empleados de dicha empresa para ser entregados con urgencia al ministerio de trabajo. Utilice el método de la burbuja para realizar dicho ordenamiento.

En el siguiente programa (P206.cpp) de un solo cuerpo se utilizan tres funciones declaradas al inicio que son:

- Función menor que se encarga de ordenar el arreglo en forma ascendente por el método de la burbuja.
- Función mayor que se encarga de ordenar el arreglo en forma descendente por el método de la burbuja.
- Función Print que se encarga de imprimir el arreglo que estamos trabajando.

PD. Me disculpa que muchas de mis programas partes son en ingles todas mis practicas son en ese idioma.

Función Menor:

```
// Ordena en forma Ascendente
template < typename T, size_t size >
void Menor( array< T, size > &items )
{   // loop de elementos
    for ( size_t i = 0; i < items.size() - 1; ++i )
    {
        size_t indexOfSmallest = i; // Almacena el valor mas pequeno

        // Encuentra el item mas pequeno
        for ( size_t index = i + 1; index < items.size(); ++index )
            if ( items[ index ] < items[ indexOfSmallest ] )
                indexOfSmallest = index;

        // Cambia la posicion de los elementos.
        T hold = items[ i ];
        items[ i ] = items[ indexOfSmallest ];
        items[ indexOfSmallest ] = hold;
    }
}
```

Función Mayor:

```
29 // Ordena en forma descendente
30 template < typename T, size_t size >
31 void Mayor( array< T, size > &items )
32 {   // loop de los elementos
33     for ( size_t i = 0; i < items.size() - 1; ++i )
34     {
35         size_t indexOfGreater = i; // Guarda el Valor mas Grande
36 
37         // Encuentra el item mas Grande
38         for ( size_t index = i + 1; index < items.size(); ++index )
39             if ( items[ index ] > items[ indexOfGreater ] )
40                 indexOfGreater = index;
41 
42         // swap the elements at positions i and indexOfSmallest
43         T hold = items[ i ];
44         items[ i ] = items[ indexOfGreater ];
45         items[ indexOfGreater ] = hold;
46     }
47 }
```

Función Print:

```
49 //Imprime los elementos del arreglo
50 template < typename T, size_t size >
51 void print( array< T, size > &items )
52 {
53 
54     cout << "Element" << setw( 13 ) << "Value" << endl;
55     //loop para el barrido del arreglo
56     for ( size_t j = 0; j < items.size(); ++j )
57         cout << setw( 7 ) << j << setw( 13 ) << items[ j ] << endl;
58 
59     cout << endl;
60 }
61 
```

La función main: en la funcion main se llenan los arreglos con números aleatorios.

```
int main()
{

        //Definicion de Variables y Arreglos

        srand( static_cast<unsigned int>( time( 0 ) ) );
        const unsigned int N=25;
        array< double, N > n;

        // Inicializacion de arreglo a 0
```

```
        for ( size_t i = 0; i < n.size(); ++i )
                n[ i ] = 0; // set element at location i to 0


        // Imprime los Valores del arreglo
        cout << "Los Valores Iniciales del Arreglo son:" <<endl;
        print(n);

        // Asignacion de Valores de Bsf aleatorios
        for ( size_t i = 0; i < n.size(); ++i )
                n[ i ] = 100 + rand() % 10000;

        // Imprime los Valores del arreglo
        cout << "Los Valores de Sueldos del Arreglo son:" <<endl;
        print(n);

        // Selecciona el Tipo de Orden
        int orden=0;
        cout << "Desea ordenar de Mayor-menor (1) o de menor-Mayor(2):" <<endl;
        cin >> orden;

        if (orden ==1){
                cout << "Ordena de Mayor a Menor" <<endl;
                Mayor(n);

                // Imprime los Valores del arreglo
                cout << "Los Valores Ordenados del Arreglo son:" <<endl;
                print(n);
                }
        else{
                cout << "Ordena de menor a Mayor" <<endl;
                Menor(n);

                // Imprime los Valores del arreglo
                cout << "Los Valores Ordenados del Arreglo son:" <<endl;
                print(n);
                }

        return 0;


}
```

Salida del Programa:

```
gp@gp-pc:~/programs$ ./P2O6.b
Los Valores Iniciales del Arreglo son:
Element       Value
```

0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0
18	0
19	0
20	0
21	0
22	0
23	0
24	0

Los Valores de Sueldos del Arreglo son:

Element	Value
0	3569
1	9859
2	7563
3	5477
4	7272
5	2034
6	5021
7	7395
8	2423
9	9739
10	2058
11	3659
12	4979
13	138
14	2294
15	8907
16	3021
17	5265
18	4465
19	5325

```
    20        9996
    21        3007
    22        1582
    23        3673
    24        1456

Desea ordenar de Mayor-menor (1) o de menor-Mayor(2):
1
Ordena de Mayor a Menor
Los Valores Ordenados del Arreglo son:
Element       Value
     0        9996
     1        9859
     2        9739
     3        8907
     4        7563
     5        7395
     6        7272
     7        5477
     8        5325
     9        5265
    10         5021
    11         4979
    12         4465
    13         3673
    14         3659
    15         3569
    16         3021
    17         3007
    18         2423
    19         2294
    20         2058
    21         2034
    22         1582
    23         1456
    24          138
```

OBJETIVO 7

3- En una empresa se tienen almacenados desde el año 2000 hasta el 2019 en un arreglo el total de bombas de agua para vehículos pesados como se muestra en la figura 2. Haciendo uso de la BÚSQUEDA BINARIA, implemente un programa en C++ para saber si el valor de 12 se encuentra y en qué posición está dentro del arreglo.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
30	16	12	45	8	31	21	15	16	41	33	11	81	23	12	36	10	32	23	55

Figura 2

En el siguiente programa (P307.cpp) de un solo cuerpo se utilizan tres funciones declaradas al inicio que son:

- Función print que se encarga de de imprimir el arreglo que estamos trabajando.
- Función menor que se encarga de ordenar el arreglo en forma descendente por el método de la burbuja ya utilizado anteriormente.
- Función binarySearch que se encarga de realizar la busqueda binaria .

Función Print

```
10 //Imprime los elementos del arreglo
11 template < typename T, size_t size >
12 void print( array< T, size > &items )
13 {
14
15     cout << "Element" << setw( 13 ) << "Value" << endl;
16     //loop para el barrido del arreglo
17     for ( size_t j = 0; j < items.size(); ++j )
18         cout << setw( 7 ) << j << setw( 13 ) << items[ j ] << endl;
19
20     cout << endl;
21 }
```

Función Menor

```
23 // Ordena en forma Ascendente
24 template < typename T, size_t size >
25 void Menor( array< T, size > &items )
26 {   // loop de elementos
27     for ( size_t i = 0; i < items.size() - 1; ++i )
28     {
29         size_t indexOfSmallest = i; // Almacena el valor mas pequeno
30 
31         // Encuentra el item mas pequeno
32         for ( size_t index = i + 1; index < items.size(); ++index )
33             if ( items[ index ] < items[ indexOfSmallest ] )
34                 indexOfSmallest = index;
35 
36         // Cambia la posicion de los elementos.
37         T hold = items[ i ];
38         items[ i ] = items[ indexOfSmallest ];
39         items[ indexOfSmallest ] = hold;
40     }
41 }
```

Función binarySearch

```
// Realiza la busqueda binaria
template < typename T, size_t size >
int binarySearch( const array< T, size > &items, const T& key)
{
	int low = 0; // Indice Bajo
	int high = items.size() - 1; // Indice Alto
	int middle = ( low + high + 1 ) / 2; // Elemento central
	int location = -1; // -1 si no es encontrado

	do // loop para buscar elemento
	{

		// si el elemento esta en el centro
		if ( key == items[ middle ] )
			location = middle;
		else if ( key < items[ middle ] ) // mitad es alta
			high = middle - 1; // elimina la mitad alta
		else // mitad es bajo
			low = middle + 1; // elimina la mitad baja

		middle = ( low + high + 1 ) / 2; // recalculo de la mitad
	} while ( ( low <= high ) && ( location == -1 ) );

	return location; // devuelve la ubicacion buscada

}
```

La función Main

```
int main()
{

	//Definicion de Variables y Arreglos


	const unsigned int N=20;
	array< int, N > bombas = {30,16,12,45,8,31,21,15,16,41,33,11,81,23,12,36,10,32,23,55};

	// Imprime los Valores del arreglo
	cout << "Los Valores Iniciales del Arreglo son:" <<endl;
	print(bombas);

	//Ordena el arreglo
	cout << "Ordena de Mayor a Menor" <<endl;
	Menor(bombas);

	// Imprime los Valores del arreglo Ordenado
	cout << "Los Valores Ordenados del Arreglo son:" <<endl;
	print(bombas);

	// Entrada del usuario
	cout << "Porfavor Introduzca Valor Buscado (-1 to quit): ";
	int searchKey;
	cin >> searchKey;
	cout << endl;

	// Introduce un valor a buscar , -1 para Salir
	while ( searchKey != -1 )
	{
		// Busqueda binaria para hallar el valor buscado
		int position = binarySearch( bombas, searchKey );

		// -1 se entrega si no encuentra el valor
		if ( position == -1 )
			cout << "El Entero " << searchKey << " no fue encontrado.\n";
		else
			cout << "El entero " << searchKey
			<< " fue encontrado en " << position << ".\n";

		// Entrada del usuario
		cout << "Porfavor Introduzca Valor Buscado (-1 to quit): ";
		cin >> searchKey;
		cout << endl;
	}

	return 0;
```

```
}
```

La salida del programa es:

```
gp@gp-pc:~/programs$ ./P3O7.b
Los Valores Iniciales del Arreglo son:
Element       Value
      0          30
      1          16
      2          12
      3          45
      4           8
      5          31
      6          21
      7          15
      8          16
      9          41
     10          33
     11          11
     12          81
     13          23
     14          12
     15          36
     16          10
     17          32
     18          23
     19          55

Ordena de Mayor a Menor
Los Valores Ordenados del Arreglo son:
Element       Value
      0           8
      1          10
      2          11
      3          12
      4          12
      5          15
      6          16
      7          16
      8          21
      9          23
     10          23
     11          30
     12          31
     13          32
     14          33
```

```
   15        36
   16        41
   17        45
   18        55
   19        81

Porfavor Introduzca Valor Buscado (-1 to quit): 36

El entero 36 fue encontrado en 15.
Porfavor Introduzca Valor Buscado (-1 to quit): -1

gp@gp-pc:~/programs$
```

Conclusiones

En la realización de esta actividad hemos aprendido a trabajar nuevas estructuras de datos mas complejas como los punteros , las clases, arboles y listas. Que nos permiten manejar otras situaciones mas abstractas del mundo real y de esta manera programar soluciones a diversos problemas.

Referencias

- Algorithms C++. Yang Hu. Verejava. 2020.
- C++ How to Program. Paul Deitel. 9 Edition. Pearson Editorial. 2014
- C Data ++ Plus Structures. Nell Dale. Third Edition. Jones and Bartlett Publishers. 2003
- Data Structures and Algorithms in C++. .Second Edition. Michael T. Goodrich. John Wiley & Sons, Inc. 2001.
- Effective C++ Digital Collection. Scoot Meyers. Addison-Wesley. 2012.
- Fourth Edition Data Structures and Algorithm Analysis in C++. Mark Allen Weiss. Pearson. 2014.
- Objects, Abstraction, Data Structures And Design Using C ++. Elliot B. Koffman. John Wiley & Sons, Inc . 2012.
- Programación en C,C++, Java y UML. Luis Joyanes Aguilar. 2da Edicion. Editoral McGrawHill. 2014