Tecnológico de Monterrey

Programación de estructuras de datos y algoritmos fundamentales

Act 2.3 - Actividad Integral de Conceptos Básicos y Algoritmos Fundamentales (Evidencia Competencia 2)

Javier Corona Del Río A01023063

Kevin López Cano A01028138

ADT de la clase:

```
class ConexionesComputadora
{
private:
  string IP;
  string Nombre;
  stack <string> ConexionesEntrantes;
  queue <string> ConexionesSalientes;

public:
  //Constructor por default
  ConexionesComputadora()
  {

  }
  //Constructor con parametros
  ConexionesComputadora(string IP_, string Nombre_, stack <string>
ConexionesEntrantes_, queue <string> ConexionesSalientes_)
  {
     setIP(IP_);
     setNombre(Nombre_);
     ConexionesEntrantes = ConexionesEntrantes_;
     ConexionesSalientes = ConexionesSalientes_;
  }
  //Destructor
  ~ConexionesComputadora()
  {

  }
  //Setters
  void setIP(string IP_)
  {
     IP = IP_;
```

```
}

void setNombre(string Nombre_)
{
    Nombre = Nombre_;
}
//Getters
string getIP()
{
    return IP;
}

string getNombre()
{
    return Nombre;
}

stack <string> getStack()
{
    return ConexionesEntrantes;
}

queue <string> getQueue()
{
    return ConexionesSalientes;
}
//Funcion print para imprimir el objeto.
void print()
{
    cout << "Ahora el IP del usuario es: " << getIP() << endl;
    cout << endl;
    cout << "El hostmail del usuario es: " << getNombre() << endl;
    cout << endl;
}
```

```
};
```

Nuestro ADT Se conforma de una clase llamada ConexionesComputadora cuyos objetos tienen como atributos dos string: IP, Nombre y dos objetos uno tipo stack y otro tipo queue finalmente creamos getters y setters para poder acceder tanto a los atributos como a los objetos stack y queue, los cuales eran privados, además le agregamos una función print que nos permite imprimir el IP y el nombre del objeto.

Justificación de nuestras estructuras de datos lineales:

Elegimos ConexionesEntrantes como una estructura de datos lineal tipo stack ya que esta es la que sigue el principio de LIFO(Last in First out).
Elegimos ConexionesSalientes como una estructura de datos lineal tipo stack ya que esta es la que sigue el principio de FIFO(First in First out).

¿Qué hace el código?

El usuario da un número entre 1 y 150 el cual será usado para completar la dirección IP que se utilizará, esto se realiza completando la IP estándar de la compañía con la ip del usuario.

Luego se procede a asignar conexionesEntrantes y conexionesSalientes dependiendo si es un IP origen o destino. Finalmente, con este IP que generamos buscamos a qué usuario pertenece y creamos un objeto tipo ConexionComputadoras asignando el IP el nombre de usuario y los dos objetos conexionesEntrantes y conexionesSalientes.

Para resolver las preguntas, primero revisamos el último elemento de conexionEntrante el cual es un stack y decimos cual es y si es una conexión interna o externa.

Por otro lado, para saber las conexiones entrantes y salientes imprimimos el size de conexionesEntrantes y conexionesSalientes es decir, un stack y un queue respectivamente.

Finalmente para la pregunta extra evaluamos si hay 3 elementos consecutivos iguales en conexionesEntrantes, es decir, en nuestro stack.

Reflexión:

En definitiva, las estructuras de datos lineales son muy importantes en la implementación de algoritmos computacionales ya que la memoria de las computadoras de por sí está organizada de forma lineal además dichas estructuras facilitan el acceso a los datos debido a que cada elemento está unido a su anterior y a su siguiente adyacente.