ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

ALGORITMOS Y ESTRUCTURAS DE DATOS

ASIGNATURA:	Algoritmos y Estructuras de Datos
PROFESOR:	Ing. Loarte Byron
PERÍODO ACADÉMICO:	

DEBER N° 13

TÍTULO:

Algoritmos de Búsqueda: Anchura y Profundidad, Almacenamiento de datos en archivos de texto

ESTUDIANTE

Diana Katherine Almeida Anchatuña

FECHA DE REALIZACIÓN: 14 / 01 / 20

FECHA DE ENTREGA: 21 / 01 / 20

CALIFICACIÓN OBTENIDA:

FIRMA DEL PROFESOR:

1 PROPÓSITO DE LA PRÁCTICA

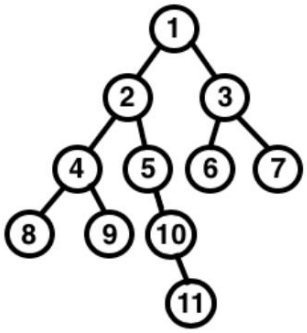
Aplicación de los algoritmos de búsqueda: anchura y profundidad para localizar un elemento en específico dentro de una estructura de datos, además verificar su existencia.

2 OBJETIVOS ESPECÍFICOS

Los Algoritmos de Búsqueda: anchura y profundidad para la resolución de problemas que tengan por objeto encontrar un elemento dentro de un grafo y completar el Taller en clase mediante los procedimientos para guardar archivos.

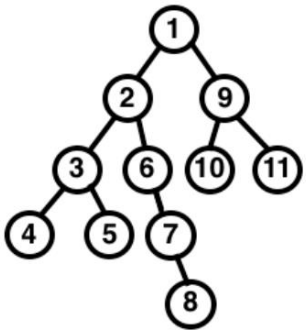
3 DESARROLLO Y RESULTADOS DE LA PRÁCTICA

1. Realizar un DFS y BFS para los siguientes grafos.



DFS = {1,2,4,8,9,5,10,11,3,6,7}

BFS = {1,2,3,4,5,6,7,8,9,10,11}



DFS = {1,2,3,4,5,6,7,8,9,10,11}

BFS = {1,2,9,3,6,10,11,4,5,7,8}

2. Terminado el ejercicio planteado en el taller, pero ahora cada resultado debe almacenarse en un archivo diferente

URL: https://github.com/AlmDiana/Algoritmos-Estructuras-Datos/tree/Taller_con_Archivos

CÓDIGO:

```
#include <iostream>
#include <stdlib.h>
#include <string>
```

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

int menu();
//Ejercicio 1
void ingresar_1(int a[], int n);
void cambiar(int a[]);
//Ejercicio 2
void ingresar_2(int a[], int n);
void burbuja (int a[], int n);
void imprimir (int a[], int n);
//Ejercicio 3
void ingresar(float depositos[]);
void buscar (float depositos[]);


int main()
{
   int opc, dim;
   string respuesta;
   int Max = 100;
   int a[6], a2[Max];

   setlocale(LC_CTYPE, "Spanish");
   do
   {
      opc = menu();
      switch (opc)
      {
      case 1:
         ingresar_1(a,6);
         cambiar(a);
         break;
      case 2:
         cout << "Ingrese la dimensión del arreglo: ";
         cin >> dim;
         Max = dim;
         ingresar_2 (a2,Max);
         burbuja(a2,Max);
         cout << "Arreglo ordenado: ";
         imprimir (a2,Max);
         break;
      case 3:
         float depositos[12];
         ingresar(depositos);
         buscar(depositos);
         break;
      case 4:
         return 0;
      }
      cout << endl << endl << "¿Desea regresar el Menú? (si/no) : ";
```

```
        cin >> respuesta;
    }
    while (respuesta == "si");
}

int menu()
{
    int op;
    do
    {
        cout << endl << "     ------------------  M E N Ú  ------------------" << endl << endl;
        cout << "1. Ejercicio 1 - Intercambiar posiciones" << endl;
        cout << "2. Ejercicio 2 - Algoritmo de ordenamiento" << endl;
        cout << "3. Ejercicio 3 - Cuenta de Ahorros virtual Banco del Pichincha" << endl;
        cout << "4. Salir" << endl;
        cout << endl << "Ingrese una opción: ";
        cin >> op;
        system ("cls");
        if (op < 0 || op > 4)
        {
            cout << endl << "ERROR! Opción no válida, ingrese nuevamente" << endl;
        }
    }
    while (op < 0 || op > 4);
    return op;
}

//Funciones Ejercicio 1

void ingresar_1(int a[], int n)
{
    ofstream archivo;
    archivo.open("ElementosOpcion1.txt", ios::out);

    archivo << "Elementos del Arreglo:" << endl;
    for(int i=0; i<n; i++)
    {
        cout<<"\nIngrese el elemento "<<i+1<<": ";
        cin>>a[i];
        archivo << a[i] << endl;
    }
    archivo.close();
}
void ingresar_2(int a[], int n)
{
    ofstream archivo;
    archivo.open("ElementosOpcion2.txt", ios::out);

    archivo << "Elementos del Arreglo:" << endl;
    for(int i=0; i<n; i++)
    {
```

```
        cout<<"\nIngrese el elemento "<<i+1<<": ";
        cin>>a[i];
        archivo << a[i] << endl;
    }
    archivo.close();
}
void cambiar(int a[])
{
    ofstream archivo;
    archivo.open("IntercambiarPosiciones.txt", ios::out);

    archivo << "Intercambiar Posiciones :" << endl;
    cout<<"\nEl nuevo vector es: ";
    for(int i=5; i>=0; i--)
    {
        cout << a[i] <<" ";
        archivo << a[i] << endl;
    }
    archivo.close();
}

//Funciones Ejercicio 2

void burbuja (int a[], int n)
{
    for (int i = 0; i < n; i++)
    {
        for(int j = i+1; j <= n-1; j++)
        {
            if (a[i] > a[j])
            {
                int aux = a[i];
                a[i] = a[j];
                a[j] = aux;
            }
        }

    }
}
void imprimir(int a[], int n)
{
    ofstream archivo;
    archivo.open("ArregloOrdenado.txt", ios::out);

    archivo << "Arreglo Ordenado:" << endl;
    for (int i = 0; i < n ; i++)
    {
        cout << a[i] << "  ";
        archivo << a[i] << endl;
    }
    archivo.close();
```

```
}
//Funciones Ejercicio 3
void ingresar(float depositos[])
{
   float dinero;
   ofstream archivo;
   archivo.open("CuentaAhorros.txt", ios::out);

   archivo << "Cuenta de Ahorros Banco del Pichincha" << endl;
   cout << "Bienvenido Sr. Alberto Perez,"
        << " por favor ingrese sus depositos mensuales desde Enero hasta Diciembre: " << endl;
   archivo << "Bienvenido Sr. Alberto Perez, por favor ingrese sus depositos mensuales desde
Enero hasta Diciembre: " << endl;
   for (int i = 1; i <= 12; i++)
   {
      do
      {
         cout << i << ": ";
         cin >> dinero;
      }
      while (dinero < 0);
      depositos[i] = dinero;
      archivo << depositos[i] << endl;
   }
   archivo.close();
}

void buscar (float depositos[])
{
   bool check = false;
   float cant;
   int j;
   ofstream archivo;
   archivo.open("Busqueda.txt", ios::out);

   cout << "Por favor ingrese el deposito a buscar: ";
   archivo << "Por favor ingrese el deposito a buscar: " << endl;
   cin >> cant;
   archivo << cant << endl;
   for (j = 1; j <= 12; j++)
   {
      if (depositos[j] == cant)
      {
         cout << "Deposito fue realizado en: ";
         switch (j)
         {
         case 1:
            cout << "Enero" << endl;
            archivo << "Enero" << endl;
            break;
         case 2:
```

```
            cout << "Febrero" << endl;
            archivo << "Febrero" << endl;
            break;
        case 3:
            cout << "Marzo" << endl;
            archivo << "Marzo" << endl;
            break;
        case 4:
            cout << "Abril" << endl;
            archivo << "Abril" << endl;
            break;
        case 5:
            cout << "Mayo" << endl;
            archivo << "Mayo" << endl;
            break;
        case 6:
            cout << "Junio" << endl;
            archivo << "Junio" << endl;
            break;
        case 7:
            cout << "Julio" << endl;
            archivo << "Julio" << endl;
            break;
        case 8:
            cout << "Agosto" << endl;
            archivo << "Agosto" << endl;
            break;
        case 9:
            cout << "Septiembre" << endl;
            archivo << "Septiembre" << endl;
            break;
        case 10:
            cout << "Octubre" << endl;
            archivo << "Octubre" << endl;
            break;
        case 11:
            cout << "Noviembre" << endl;
            archivo << "Noviembre" << endl;
            break;
        case 12:
            cout << "Diciembre" << endl;
            archivo << "Diciembre" << endl;
            break;
        }
        check = true;
    }
}
if (check == false)
{
    cout << "Deposito no existente" << endl;
    archivo << "Deposito no existente" << endl;
```

```
}
archivo.close();
```