LAPORAN PRAKTIKUM

STRUKTUR DATA DAN ALGORITME

MODUL 9

**DOUBLE LINKED LIST**

Logo

Description automatically generated

**Disusun Oleh :**

Nama : Fatkhurrohman Purnomo

NIM : 21102125

**Dosen Pengampu**

Ipam Fuaddina Adam, S.T., M.Kom.

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**PURWOKERTO**

**2022**

# Dasar Teori

**Definisi dan konsep**

Graph (Graf) adalah kumpulan dari simpul dan busur yang secara matematis dinyatakan sebagai:

G = (V, E)

G = Graph

V = Vertex/Node/Simpul/Titik

E = Edge/Busur/Ruas/Garis

Contoh: Suatu graph G(E,V) dengan elemen-elemen sebagai berikut:

V mengandung 4 simpul: A, B, C, D

E mengandung 5 busur:

e1 = (A,B) e2 = (B,C) e3 = (A,D)

e4 = (C,D) e5 = (A,C)

Dapat digambarkan:

Diagram

Description automatically generated

**Jenis-Jenis Graph**

Graph Berarah (Directed) dan Tak Berarah (Undirected)

A picture containing text, map, watch

Description automatically generated

**Graph berarah (directed graph):**

Urutan simpul mempunyai arti. Misal busur AB adalah e1 sedangkan busur BA adalah e8.

**Graph tak berarah (undirected graph):**

Urutan simpul dalam sebuah busur tidak diperhatikan. Misal busur e1 dapat disebut busur AB atau BA.

A picture containing text, map, watch, bicycle

Description automatically generated

**Graph Berbobot (Weighted Graph)**

Jika setiap busur mempunyai nilai yang menyatakan hubungan antara 2 buah simpul, maka busur tersebut dinyatakan memiliki bobot.

**Representasi Graph**

**Representasi dengan Matriks**

****

**Representasi dengan linked list**

****

Yang perlu diperhatikan dalam membuat representasi graph dalam bentuk linked list adalah membedakan antara simpul vertex dengan simpul edge. Simpul vertex menyatakan simpul atau vertex dan simpul edge menyatakan busur (hubungan antar simbol). Struktur keduanya bisa sama bisa juga berbeda tergantung kebutuhan, namun biasanya disamakan. Yang membedakan antara simpul vertex dengan simpul edge nantinya adalah anggapan terhadap simpul tersebut juga fungsinya masing-masing.

Diagram

Description automatically generated

Ref:

Modul 9: GRAPH

[Graf dan Pohon Dalam Algoritma Pemrograman C++ | KURNIA'S TYPING (kurniatyping.blogspot.com)](http://kurniatyping.blogspot.com/2019/05/graf-dan-pohon-dalam-algoritma.html)

[GRAF DAN POHON DALAM PEMROGRAMAN C++ ~ MILDA HAYATI](http://mildahayati.blogspot.com/2019/06/graf-dan-pohon-dalam-pemrograman-c.html)

# Guided

## Program Menampilkan Data Graph

|  |
| --- |
| #include <iostream>  using namespace std;  ///PROGRAM MENAMPILKAN DATA GRAPH  //Deklarasi Node/Simpul  string simpul[7] = {"Jakarta",                      "Bandung",                      "Bekasi",                      "Tasikmalaya",                      "Semarang",                      "Purwokerto",                      "Yogyakarta"};  //Deklarasi Edge/Busur  int busur[7][7] = { {0,7,8,0,0,0,0},                      {0,0,5,0,0,15,0},                      {0,6,0,0,5,0,0},                      {0,5,0,0,0,4,0},                      {23,0,0,10,0,0,8},                      {0,0,0,0,7,0,3},                      {0,0,0,0,9,4,0}};  //Tampil Data Graph  void tampilGraph()  {      for (int baris = 0; baris < 7; baris++){          cout << " " << simpul[baris] << ": ";          for (int kolom = 0; kolom < 7; kolom++){              if (busur[baris][kolom] != 0){                  cout << " " << simpul[kolom] << "(" << busur[baris][kolom] << ")" << " ";              }          }      cout << endl;      }  }  int main()  {      tampilGraph();      return 0;  } |

**Deskripsi:**

Yang pertama adalah membuat array simpul untuk nama tempat, dan array busur untuk graph. Lalu membuat fungsi untuk menampilkan graph.

**Output:**

Text

Description automatically generated

## Program Representasi Graph Dengan Matriks Berdasarkan Input User

|  |
| --- |
| #include <iostream>  using namespace std;  ///PROGRAM REPRESENTASI GRAPH DENGAN MATRIKS BERDASARKAN INPUT USER  //Deklarasi  int jumlahSimpul = 7;  string \*dataSimpul; //digunakan untuk membuat array dinamis  int \*\*dataBusur; //digunakan untuk membuat array dinamis dua dimensi  bool cekMatriks = false;  //Buat Matriks Graph  void buatMatriks()  {      dataSimpul = new string[jumlahSimpul]; //membuat array string dinamis      //Membuat simpul dengan jumlah yang dimasukkan user      dataBusur = new int\*[jumlahSimpul]; //membuat pointer array integer dinamis      dataBusur[0] = new int[jumlahSimpul \* jumlahSimpul]; //menginisialisasi array busur indeks-0 dengan array integer dinamis berindeks pointer        for (int i = 1; i < jumlahSimpul; i++){          dataBusur[i] = dataBusur[i-1] + jumlahSimpul;      }      //Membuat matriks 2 dimensi dengan ukuran sesuai masukan user      //Proses input nama simpul      cout << " Silahkan masukkan nama simpul" <<endl; //input nama simpul sebanyak jumlah simpul      for (int i = 0; i < jumlahSimpul; i++){          cout << " Simpul " <<i+1<< " : ";          cin >> dataSimpul[i];      }      //Proses input nilai busur (bobot)      cout << " Silahkan masukkan bobot antar simpul" << endl; //input bobot busur antar simpul      for (int baris = 0; baris < jumlahSimpul; baris++){          for (int kolom = 0; kolom < jumlahSimpul; kolom++){              cout << " " << dataSimpul[baris] << " --> " << dataSimpul[kolom] << " : "; //input bobot sebagai baris dan kolom              cin >> dataBusur[baris][kolom];          }      }        cout << endl;      //Inisialisasi Pengecekan Matriks      cekMatriks = true; //atur keberadaan matriks menjadi ada/true  }  //Tampil Matriks Graph  void tampilMatriks()  {      if (cekMatriks){ //jika matriks ada          cout << " ";          for (int i = 0; i < jumlahSimpul; i++){ //membuat kolom nama simpul              cout << dataSimpul[i] << " ";          }          cout << endl;            for (int baris = 0; baris < jumlahSimpul; baris++){ //membuat kolom bobot              cout << " " << dataSimpul[baris] << " "; //membuat baris nama simpul              for (int kolom = 0; kolom < jumlahSimpul; kolom++){ //membuat baris bobot                  cout << dataBusur[baris][kolom] << " ";              }          cout << endl;          }      }      else{ //jika matriks belum ada          cout << " Tidak ada matriks" << endl;      }  }  int main()  {      cout << " Silahkan masukkan jumlah simpul: "; //input jumlah simpul      cin >> jumlahSimpul;      buatMatriks();      tampilMatriks();      return 0;  } |

**Dekripsi:**

Yang pertama adalah melakukan Deklarasi variabel, Lalu membuat fungsi untuk membuat matriks dengan input dari user dan fungsi untuk menampilkan graph.

**Output:**

Text

Description automatically generated

## Program Representasi Graph dengan Linked List berdasarkan Input User

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  ///PROGRAM REPRESENTASI GRAPH DENGAN LINKED LIST  //Deklarasi Global  int jumlahSimpul = 7;  string \*dataSimpul; //digunakan untuk membuat array dinamis  int \*\*dataBusur; //digunakan untuk membuat array dinamis dua dimensi  //Deklarasi Linked List  struct Graph  {      string data;      Graph \*kanan;      Graph \*kiri;  };  Graph \*simpul, \*busur, \*awal, \*akhir, \*\*alamat, \*bantu, \*bantu2;  //Inisialisasi  void init()  {      awal = NULL;      akhir = NULL;  }  //isEmpty  int isEmpty()  {      if (awal == NULL && akhir == NULL)      return 1; //true      else      return 0; //false  }  //Buat Matriks  void buatMatriks()  {      dataSimpul = new string[jumlahSimpul];      dataBusur = new int\*[jumlahSimpul];      dataBusur[0] = new int[jumlahSimpul \* jumlahSimpul]; //inisialisasi ukuran elemen matriks      for (int i = 1; i < jumlahSimpul; i++){          dataBusur[i] = dataBusur[i-1] + jumlahSimpul;      }      //Membuat matriks 2 dimensi dengan ukuran sesuai masukan user      //Proses input nama simpul      cout << " Silahkan masukkan nama kota" <<endl; //input nama simpul sebanyak jumlah simpul      for (int i = 0; i < jumlahSimpul; i++){          cout << " Kota " <<i+1<< " : ";          cin >> dataSimpul[i];      }        //Proses input nilai busur (bobot)      cout << " Silahkan masukkan bobot antar kota" << endl; //input bobot busur antar simpul      for (int m = 0; m < jumlahSimpul; m++){          for (int n = 0; n < jumlahSimpul; n++){              cout << " " << dataSimpul[m] << " --> " << dataSimpul[n] << " : "; //input bobot sebagai baris dan kolom              cin >> dataBusur[m][n];          }      }  }  //Buat Simpul Graph  void buatSimpulGraph()  {      alamat = new Graph\*[jumlahSimpul]; //membuat pointer alamat sebanyak jumlah simpul      buatMatriks();      for (int i=0; i<jumlahSimpul; i++){          if (isEmpty() == 1){              simpul = new Graph;              simpul->data = dataSimpul[i];              simpul->kanan = NULL;              simpul->kiri = NULL;              awal = simpul;              akhir = simpul;              alamat[i] = awal;          }          else{              simpul = new Graph;              simpul->data = dataSimpul[i];              akhir->kiri = simpul;              akhir = simpul;              simpul->kiri = NULL;              simpul->kanan = NULL;              alamat[i] = akhir;          }      }      bantu = awal;      for (int m=0; m<jumlahSimpul; m++){          bantu2 = bantu;          for (int n=0; n<jumlahSimpul; n++){              if (dataBusur[m][n] != 0){                  simpul = new Graph;                  simpul->data = to\_string(dataBusur[m][n]);                  bantu2->kanan = simpul;                  simpul->kiri = alamat[n];                  simpul->kanan = NULL;                  bantu2 = simpul;              }          }          bantu = bantu->kiri;      }  }  //Tampil  void tampilGraph()  {      if (isEmpty() == 0){          bantu = awal;          while (bantu != NULL){              cout << ends << bantu->data << ": ";              bantu2 = bantu->kanan;              while (bantu2 != NULL){                  cout << ends << bantu2->kiri->data << ": " << bantu2->data << " ";                  bantu2 = bantu2->kanan;              }              cout << endl;              bantu = bantu->kiri;          }      }      else{      cout << " Graph masih kosong!" << endl;      }  }  int main()  {      init();      cout << " Silahkan masukkan jumlah kota: ";      cin >> jumlahSimpul;      buatSimpulGraph();      tampilGraph();      return 0;  } |

**Dekripsi:**

Yang pertama adalah melakukan Deklarasi variabel yang nanti akan digunakan. Dilanjut membuat graph, dan pointernya. Ada fungsi untuk inisialisasi dan isEmpty untuk mengetahui isi memori kosong atau isi. Lalu ada fungsi untuk membuat matriks dengan input dari user, dan jumlah simpul disesuaikan dari user juga. Dan fungsi menampilkan graph. Yang terakhir ada main fungsi untuk memanggil fungsi-fungsi yang dibuat.

**Output:**

Text

Description automatically generated

# Tugas (Unguided)

Modifikasi program Guided III dengan menambahkan fungsi untuk mencari busur terpendek!

|  |
| --- |
| #include <iostream>  #include <string>  using namespace std;  ///PROGRAM REPRESENTASI GRAPH DENGAN LINKED LIST  //Deklarasi Global  int jumlahSimpul = 7;  string \*dataSimpul; //digunakan untuk membuat array dinamis  int \*\*dataBusur; //digunakan untuk membuat array dinamis dua dimensi  //Deklarasi Linked List  struct Graph  {      string data;      Graph \*kanan;      Graph \*kiri;  };  Graph \*simpul, \*busur, \*awal, \*akhir, \*\*alamat, \*bantu, \*bantu2;  //Inisialisasi  void init()  {      awal = NULL;      akhir = NULL;  }  //isEmpty  int isEmpty()  {      if (awal == NULL && akhir == NULL)      return 1; //true      else      return 0; //false  }  //Buat Matriks  void buatMatriks()  {      dataSimpul = new string[jumlahSimpul];      dataBusur = new int\*[jumlahSimpul];      dataBusur[0] = new int[jumlahSimpul \* jumlahSimpul]; //inisialisasi ukuran elemen matriks      for (int i = 1; i < jumlahSimpul; i++){          dataBusur[i] = dataBusur[i-1] + jumlahSimpul;      }      //Membuat matriks 2 dimensi dengan ukuran sesuai masukan user      //Proses input nama simpul      cout << " Silahkan masukkan nama kota" <<endl; //input nama simpul sebanyak jumlah simpul      for (int i = 0; i < jumlahSimpul; i++){          cout << " Kota " <<i+1<< " : ";          cin >> dataSimpul[i];      }        //Proses input nilai busur (bobot)      cout << endl << " Silahkan masukkan bobot antar kota" << endl; //input bobot busur antar simpul      for (int m = 0; m < jumlahSimpul; m++){          for (int n = 0; n < jumlahSimpul; n++){              cout << " " << dataSimpul[m] << " --> " << dataSimpul[n] << " : "; //input bobot sebagai baris dan kolom              cin >> dataBusur[m][n];          }      }  }  //Buat Simpul Graph  void buatSimpulGraph()  {      alamat = new Graph\*[jumlahSimpul]; //membuat pointer alamat sebanyak jumlah simpul      buatMatriks();      for (int i=0; i<jumlahSimpul; i++){          if (isEmpty() == 1){              simpul = new Graph;              simpul->data = dataSimpul[i];              simpul->kanan = NULL;              simpul->kiri = NULL;              awal = simpul;              akhir = simpul;              alamat[i] = awal;          }          else{              simpul = new Graph;              simpul->data = dataSimpul[i];              akhir->kiri = simpul;              akhir = simpul;              simpul->kiri = NULL;              simpul->kanan = NULL;              alamat[i] = akhir;          }      }      bantu = awal;      for (int m=0; m<jumlahSimpul; m++){          bantu2 = bantu;          for (int n=0; n<jumlahSimpul; n++){              if (dataBusur[m][n] != 0){                  simpul = new Graph;                  simpul->data = to\_string(dataBusur[m][n]);                  bantu2->kanan = simpul;                  simpul->kiri = alamat[n];                  simpul->kanan = NULL;                  bantu2 = simpul;              }          }          bantu = bantu->kiri;      }  }  //Tampil  void tampilGraph()  {      cout << endl << " Hasil graph: " << endl;      if (isEmpty() == 0){          bantu = awal;          while (bantu != NULL){              cout << " " << bantu->data << ": ";              bantu2 = bantu->kanan;              while (bantu2 != NULL){                  cout << bantu2->kiri->data << ": " << bantu2->data << " ";                  bantu2 = bantu2->kanan;              }              cout << endl;              bantu = bantu->kiri;          }      }      else{      cout << " Graph masih kosong!" << endl;      }  }  // fungsi mencari busur terpendek  void cariBusur()  {      for (int i=0; i<jumlahSimpul; i++){          for (int m=0; m<jumlahSimpul; m++){              if (dataBusur[i][m] != 0){                  // mencari simpul yang memiliki busur terpendek                  if (dataBusur[i][m] < dataBusur[i][m+1]){                      dataBusur[i][m+1] = dataBusur[i][m];                  }              }          }      }      cout << endl << " Simpul terpendek: " << dataBusur[jumlahSimpul-1][jumlahSimpul-1] << endl;  }  int main()  {      init();      cout << " Silahkan masukkan jumlah kota: ";      cin >> jumlahSimpul;      cout << endl;      buatSimpulGraph();      tampilGraph();      cariBusur();      return 0;  } |

**Deskripsi:**

Program sama seperti pada latihan 3, dengan tambahan fungsi untuk mencari simpul terpendek, dengan menggunakan for looping, dan if lalu dimasukan ke dalam array.

**Output:**

Text

Description automatically generated

# Kesimpulan

1. Bisa membuat graph
2. Belajar lebih dalam pengaplikasian graph dengan double linked list
3. Lebih mahir dalam menggunakan bahasa C++
4. Bisa melakukan problem solving bagi program yang error
5. Lebih paham dalam membuat program
6. Melatih daya pikir, imajinasi, dan langkah-langkah dalam membuat program