Nama : Helmi Efendi Lubis

NIM : 1301223338

Tugas Pendahuluan Modul 14

Link github: https://github.com/helmiel/StrukturDataTP/tree/main/TP14

Header

header.h | main.cpp | source.cpp

```
#ifndef HEADER_H_INCLUDED
#define HEADER_H_INCLUDED

#include <iostream>
using namespace std;

typedef struct vertex *adrVertex;
typedef struct edge *adrEdge;
struct vertex{
    char id;
    adrVertex nextVertex;
};
struct edge{
    adrVertex vertex_1;
    adrVertex vertex_2;
    adrEdge nextEdge;
```

header.h | main.cpp | source.cpp

```
    adrEdge nextEdge;
};
struct listOfVertex{
    adrVertex firstVertex;
};
struct listOfEdge{
    adrEdge firstEdge;
};
struct graph{
    listOfVertex Vertex;
    listOfEdge Edge;
};

void createVertex_1301223338(char newVertexID, adrVertex &v);
void initGraph_1301223338(graph &G);
void addVertex_1301223338(graph &G, char newVertexID);
```

header.h | main.cpp | source.cpp

```
25      listOfVertex Vertex;
26      listOfEdge Edge;
27  };
28
29  void createVertex_1301223338(char newVertexID, adrVertex &v);
30  void initGraph_1301223338(graph &G);
31  void addVertex_1301223338(graph &G, char newVertexID);
32  void buildGraph_1301223338(graph &G);
33
34  bool vertexExists_1301223338(graph& G, char id);
35  void printVertex_1301223338(graph G);
36
37
38
39  #endif // HEADER_H_INCLUDED
40
```

Source

header.h | main.cpp | source.cpp

```
1   #include "header.h"
2
3   void createVertex_1301223338(char newVertexID, adrVertex &v){
4       v = new vertex;
5       v->id = newVertexID;
6       v->nextVertex = NULL;
7   }
8
9   void initGraph_1301223338(graph &G){
10      G.Vertex.firstVertex = NULL;
11      G.Edge.firstEdge = NULL;
12  }
```

header.h | main.cpp | source.cpp

```
13
14  void addVertex_1301223338(graph &G, char newVertexID){
15      adrVertex v;
16      createVertex_1301223338(newVertexID, v);
17
18      if (G.Vertex.firstVertex == NULL){
19          G.Vertex.firstVertex = v;
20      }else {
21          adrVertex p = G.Vertex.firstVertex;
22          while(p->nextVertex != NULL){
23              p = p->nextVertex;
24          }
25          p->nextVertex = v;
26      }
27  }
28
```

header.h | main.cpp | source.cpp

```
28
29  bool vertexExists_1301223338(graph& G, char id){
30      adrVertex v = G.Vertex.firstVertex;
31      while(v != NULL){
32          if(v->id == id){
33              return true;
34          }
35          v = v->nextVertex;
36      }
37      return false;
38  }
39
```

header.h | main.cpp | source.cpp

```
40  void buildGraph_1301223338(graph &G){
41      char id;
42      adrVertex v;
43      cout << "Masukkan id: ";
44      cin >> id;
45      while (id >= 65 && id <= 90){
46          if(!vertexExists_1301223338(G, id)){
47              addVertex_1301223338(G, id);
48          }else {
49              cout << "Vertex with ID " << id << " already exists." << endl;
50          }
51          cout << "Masukkan id: ";
52          cin >> id;
53      }
54  }
55
```

header.h | main.cpp | source.cpp

```
48          }else {
49              cout << "Vertex with ID " << id << " already exists." << endl;
50          }
51          cout << "Masukkan id: ";
52          cin >> id;
53      }
54  }
55
56  void printVertex_1301223338(graph G){
57      adrVertex v = G.Vertex.firstVertex;
58      while(v != NULL){
59          cout << v->id << " ";
60          v = v->nextVertex;
61      }
62  }
63
```

Main

header.h | main.cpp | source.cpp

```cpp
#include <iostream>
#include "header.h"

using namespace std;

int main()
{
    graph G;
    initGraph_1301223338(G);
    buildGraph_1301223338(G);
    printVertex_1301223338(G);
}
```

Output