

Nama : Helmi Efendi Lubis

NIM : 1301223338

Tugas Pendahuluan Modul 14

Header

```
header.h x main.cpp x source.cpp x
1  #ifndef HEADER_H_INCLUDED
2  #define HEADER_H_INCLUDED
3
4  #include <iostream>
5  using namespace std;
6
7  typedef struct vertex *adrVertex;
8  typedef struct edge *adrEdge;
9  struct vertex{
10     char id;
11     adrVertex nextVertex;
12 };
13 struct edge{
14     adrVertex vertex_1;
15     adrVertex vertex_2;
16     adrEdge nextEdge;
17 };
18 struct listOfVertex{
19     adrVertex firstVertex;
20 };
21 struct listOfEdge{
22     adrEdge firstEdge;
23 };
24 struct graph{
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);
```

```
header.h X main.cpp X source.cpp X
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 X main.cpp X source.cpp X
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 }
```

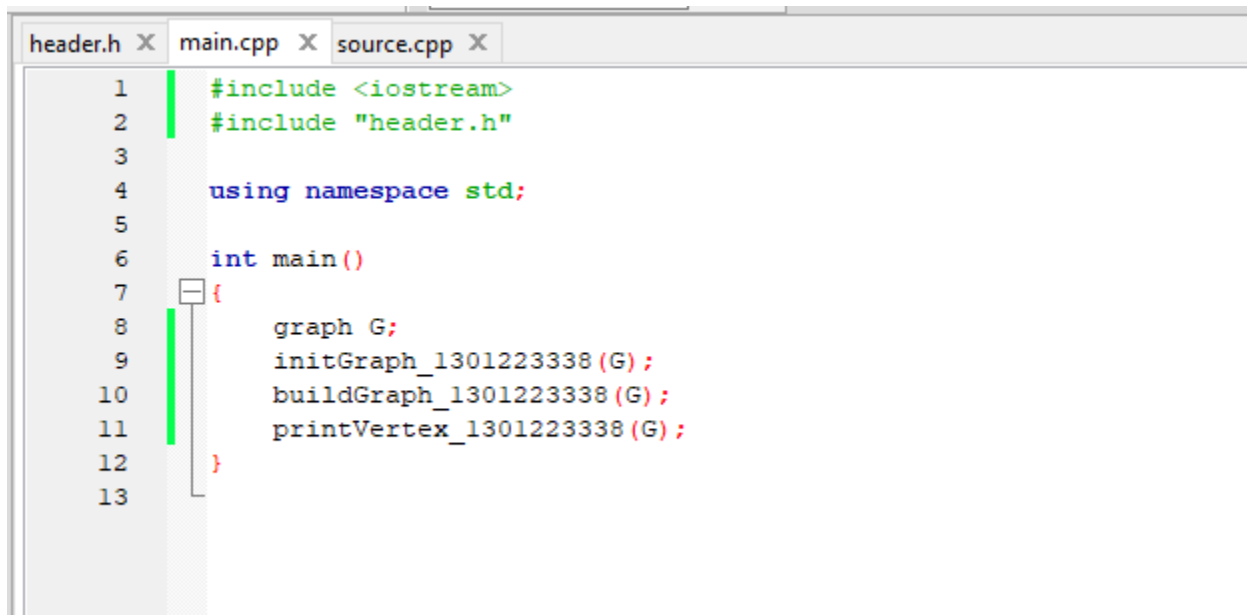
```
header.h X main.cpp X source.cpp X
13
14 void addVertex_1301223338(graph &G, char newVertexID){
15     adrVertex v;
16     createVertex_1301223338(newVertexID, v);
17
18     if (G.Vertex.firstVertex == NULL){
19         G.Vertex.firstVertex = v;
20     }else {
21         adrVertex p = G.Vertex.firstVertex;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

```
header.h X main.cpp X source.cpp X
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 X main.cpp X source.cpp X
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
```

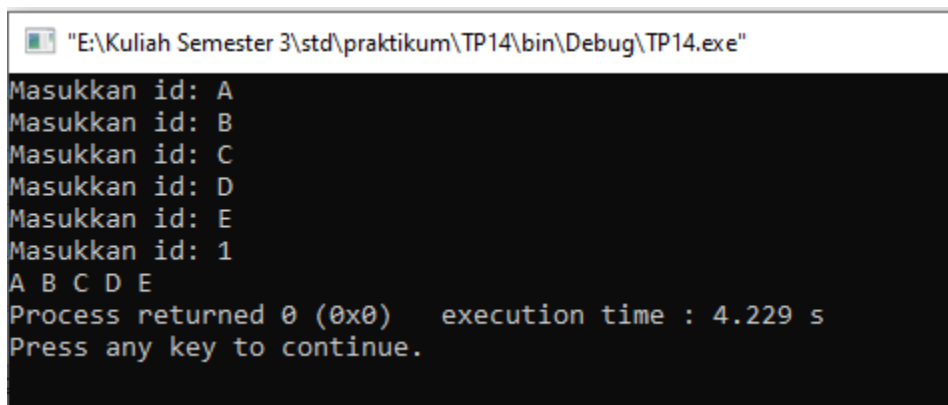
```
header.h X main.cpp X source.cpp X
48 }else {
49     cout << "Vertex with ID " << id << " already exists." << endl;
50 }
51 cout << "Masukkan id: ";
52 cin >> id;
53 }
54 }
55
56 void printVertex_1301223338(graph G){
57     adrVertex v = G.Vertex.firstVertex;
58     while(v != NULL){
59         cout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

Main



```
header.h X main.cpp X source.cpp X
1  #include <iostream>
2  #include "header.h"
3
4  using namespace std;
5
6  int main()
7  {
8      graph G;
9      initGraph_1301223338 (G);
10     buildGraph_1301223338 (G);
11     printVertex_1301223338 (G);
12 }
13
```

Output



```
"E:\Kuliah Semester 3\std\praktikum\TP14\bin\Debug\TP14.exe"
Masukkan id: A
Masukkan id: B
Masukkan id: C
Masukkan id: D
Masukkan id: E
Masukkan id: 1
A B C D E
Process returned 0 (0x0)   execution time : 4.229 s
Press any key to continue.
```