CS311 Yoshii - HW6 Intro to Graphs (based on Week 10 - Week 12)

DUE: Week 13 Monday
Start working on HW7P1 early so that you will have lots of time to do
HW7P2.

Total: 36 points Your score is:

Your Name: Eduard Martinez

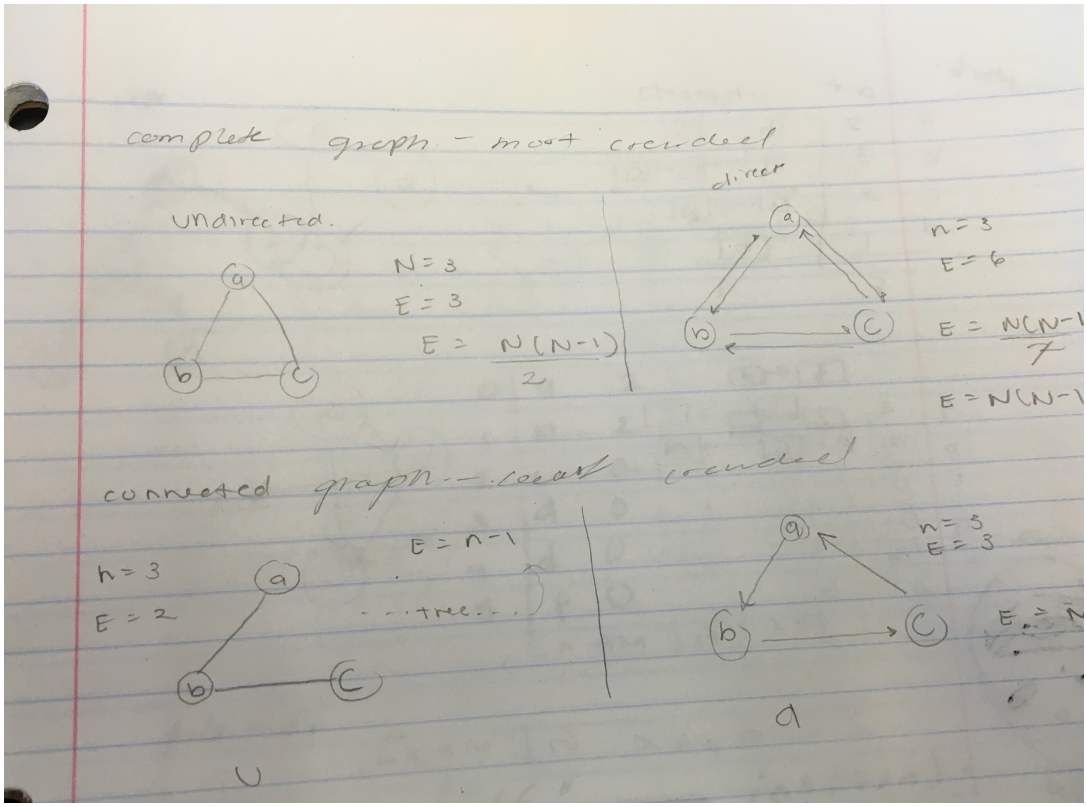
Date Turned In:

Purpose: To form fundamental understanding about graphs

Review Questions from Notes-10B:[8pts] Your score:

A **complete graph** has every vertex connected **directly** to every other vertex.
Inter2 **Draw an example with 3 vertices.[1]**
- **a. undirected graph**
- **b. directed graph**

A **connected graph** has to have a **path** between every pair of vertices.
Inter3 **Draw a smallest connected graph with 3 vertices. [1]**
- **a. undirected graph**
- **b. directed graph**



***Inter4* How many edges does a tree of N nodes have? Why?[1]**
complete u -- n(n-1)/2
complete d -- n(n-1)
connected u -- (n-1)
connected d -- n

***Inter6* If my directed graph has 200 cities, and I want to make sure there is a direct flight from any city to any other city, how many flights are needed?[1] 200*(200-1)=39800**

***Inter7* If my undirected graph has 200 cities, and I want to make sure I can drive somehow from any city to any other city, how many road are needed?[1] 200-1=199**

For an adjacency list representation, G is the table:
***Inter12* If you have N vertices and M edges,**
- how many array slots are needed in G?[1]
- how many linked list nodes are required altogether
for directed G? Why? [1]
for undirected G? Why? [1]

Review Questions from Notes-11A:[10pts] Your score:

Depth First Traversal from the notes ends like this:

```
Pop I.                              [F G top]
I has not been marked yet.
Mark I.
Adjacent vertices are A G and E.
```

push E push G push A [F G E G A top]

***Inter2* Complete this trace from this point on using exactly the same wording and the same format until the stack becomes empty.[7]**

```
Pop A.              [F G E G top]
A has been marked
Pop G.              [F G E top]
G has been marked
Pop E.               [F G top]
E has not been marked yet
Mark E.
No adjacent vertices
Pop G.              [F top]
G has been maked
Pop F.              [ top]
Adjacent vetices are I and E
Push I push E.     [I E top]
Pop E.              [I top]
E has been marked
Pop I               [top]
I has been marked
```

***Inter3* On the graph you drew in Inter1, number the vertices in the order they are marked by DFS <insert drawing here>[3]**

a-1, b-2, f-3, c-4, g-5, i-6, h-7, e-8

Program Graph Implementation:[2+16=18pts] Your score:

dgraph.h, dgraph.cpp and hw6Client.cpp must be used.

Create a **directed graph class** which has the following data member:

I provided for you HW3 files in case your HW3 is not working.

Gtable[20](an array) which contains the following (struct)in each slot:

Struct Gvertex: (this is declared outside the class)

- a vertex name (char)
- the mark/visit number (int)
- the out degree (int)
- a linked list object for adjacent vertices**(from HW3P3 slist but with char element type)**

And the following methods/member functions:

dgraph Constructor - initializes the table entries
[Make sure the names are initialized to be ' ' and visit number is 0]

dgraph Destructor - destroys the table
[Does this call the list destructor automatically? If not, you have to destroy the lists. Test and see.]

displayGraph() - displays the table content in a very readable format
But make sure you do not display unused slots

fillTable() - reads the input file **table.txt** to fill the table

Open and close the input file table.txt in here
int findOutDegree(char) – returns the out degree of the vertex whose name is given as an argument.
Use a loop but search through used slots only.
May throw an exception.
slist findAdjacency(char) – returns the linked list of adjacent vertices of the vertex whose name is given as an argument.
Use a loop but search through used slots only.
May throw an exception.
[This one calls your HW3P3 copy constructor automatically because a list is being returned.]

Note that the mark/visit number is not being used yet by these functions. It will be used in the next HW.

Note that the linked list of adjacent vertices is of type slist and thus, you can use any of the slist member functions on it.

table.txt should have the following format:
Each line is
name out-degree a-list-of-its-adjacent-vertices-separated-by-blanks
e.g.
A 2 B F

I have provided you with the input file based on Notes-11A.doc. You must use it to test your program.
Client program (hw6Client.cpp) will:
1. fillTable()
2. displayGraph()
3. findOutDegree(char) for various vertices in the graph (a loop)
a. the user will specify which vertex
b. displays the returned result
4. findAdjacency(char) for various vertices in the graph (a loop)
a. the user will specify which vertex
b. displays the returned list (use HW3P3 function)
And catches exceptions.

Q) The state of the program statement [2pts]

- **Does your program compile without errors?**
- **List any bugs you are aware of, or state "No bugs":**

Submit these 5 files:
1. **This assignment sheet with your answers**
2. **dgraph.h**
3. **dgraph.cpp**
4. **hw6Client.cpp**
5. **Test – compilation and the results of thorough test cases (script) using my table.txt**

Whether working or not, test result must include the lines for compiling your files or we will not grade your program i.e. 0 points for the program.

Did you check your comments and style against CS311 How To Comment.doc??

```
Did you answer all the questions?
```