**CS 311 HW7 Graph Algorithms Part 2**

**(based on week 11 – 12)**

**==========================================================**

**DUE: Week 14 Friday (the very last HW)**

bool dgraph::isMarked(char vertex)

{

int verte = (int)verte - 65;

You say no bugs but it has segmentation fault in the output due to the above.

**TOTAL 18 points Your score: 16**

**\*NAME: Parth Kapur**

**\*DATE SUBMITTED: 12/1/2017**

**----------------------------------------------------**

**Purpose: To be able to implement DFS of a graph**

**----------------------------------------------------**

**==================================================**

**HW7 Implementation DFS of a Graph**

**==================================================**

**[2+16=18pts] Your score:**

**Now that you have a directed graph class from HW6, you can implement DFS.**

**You also need a stack class (from HW1) so that you can push vertex names onto a stack. (Where do you need to include stack.h? What do you compile?)**

**You need to add the following 2 functions to the graph class:**

**void visit(int, char) which will enter the given visit number for a given vertex**

**this is to indicate the order in which vertices were visited.**

**Do not use a loop. Convert A to slot 0, B to slot 1 etc.**

**bool isMarked(char) which returns true if a given vertex was already visited**

**(0 means not visited)**

**Do not use a loop. Convert A to slot 0, B to slot 1 etc.**

**Make sure displayGraph now displays the visit numbers as well.**

**Your client (hw7Client.cpp) will implement the DFS algorithm from Notes-11A.doc using the stack class and the graph class functions as follows:**

**Display the graph before DFS begins.**

**Mark/visit A (\*\*), the start vertex visit number as 1.**

**Get the adjacency list of A and push adjacent vertices onto the stack.**

**Display the stack**

**While the stack is not empty do**

**{**

**Remove a vertex v from the stack.**

**Display the vertex name.**

**If v is not marked yet (visit number is 0) then**

**mark it (visit it \*\*) and inform the user E.G. “visited B”**

**get its adjacency list and put adjacent ones on the stack**

**(delete from the rear and push)**

**display the stack clearly labeling it as the stack**

**}**

**Display the Graph with visit numbers for all vertices.**

**Do not display unused (junk) entries of the Gtable.**

**(\*\*) visit numbers will start at 1 and increase as you traverse.**

**Add many labeling cout messages to make your output understandable.**

**Testing: Use the same input file as for HW6.**

**Submit the output for starting at vertex A.**

**State of the program statement [2pts]**

**Does your program compile without errors? Yes**

**List any bugs you are aware of, or state “No bugs”: No bugs that I know of.**

**Submit these 5 files:**

**This assignment sheet with your answers. DID YOU answer all questions?**

**dgraph.h - header,**

**dgraph.cpp - implementation, and**

**hw7Client.cpp - client (commented well)**

**Test – script of test results showing what you compiled and ran**

**Keep this set (HW7 files) of files for your future classes.**

**Congratulations! You have just finished writing a program that uses stack, linked list and graph classes.**