CSCE 221 Cover Page
Programming Assignment #6

First Name Last Name UIN

User Name E-mail address

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero. According to the University Regulations, Section 42, scholastic dishonesty are including: acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion, read more: Aggie Honor System Office

Type of sources			
People			
Web pages (provide URL)			
Printed material			
Other Sources			

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.

"On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work"

Your Name Date

Programming Assignment #6

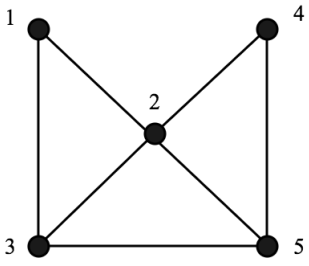
Electronic submission to eCampus due *April 28*

The assignment presentation (mandatory) in lab and a hard copy of a report due *April 29*

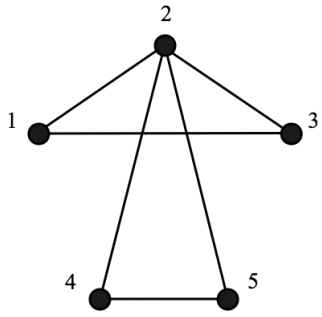
The final challenge!

Congratulations! You just unlocked your final challenge! Tai Lung is ready for you. Are you?

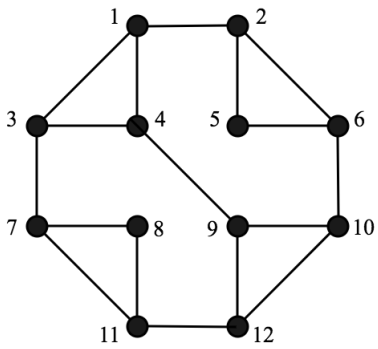
- As part of this challenge, you should write a C++ program based on the graph data structure that can determine if it is possible to draw a picture in one stroke, that is, without lifting a pen or retracing part of the picture (excluding nodes).
- Use the pictures listed below and solve the challenge by hand. If you find a way to draw a picture with these requirements, list the sequence of line segments, where each line segment is represented by two numbered endpoints (nodes). Can every picture be generated in one stroke?



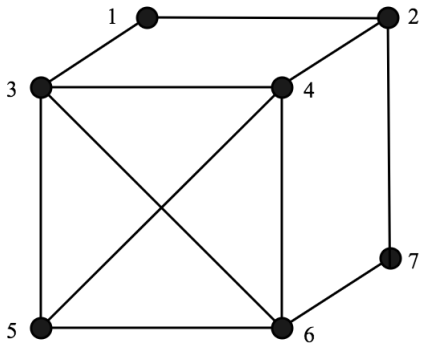
Picture 1



Picture 2

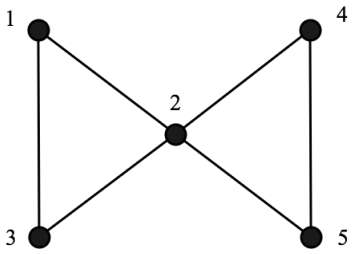


Picture 3

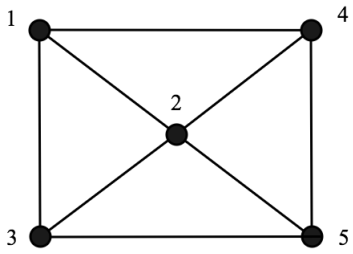


Picture 4

- The purpose of the exercise is to formulate the necessary and sufficient conditions for drawing one-stroke pictures. Those conditions allow you design an algorithm for determining the existence of a solution. Answer these questions in your report:
  - Determine for which group of pictures there are always solutions.
  - Determine for which group of pictures there are no solutions.
  - Determine for which group of pictures there exist solutions starting at one point and ending at the same point. What kind of point could be selected as the start/end point in such a case?
  - Determine for which group of pictures there exist solutions starting at one point and ending at a different point. What kind of points could be selected as the start/end points in such a case?
  - Eventually, provide the necessary and sufficient conditions for drawing one-stroke pictures.
- Hint for searching for solutions:



Picture 5



Picture 6

Notice that the output $(1,2) \to (2,3) \to (3,1)$ is incorrect because does not contain all the line segments of this pictures. You cannot continue drawing because retracing a line segment is not allowed. Your algorithm must be able to go back to the vertex 2, and search for another edge to draw.

A correct output is a sequence of the line segments marked by its endpoints:
$(1,2) \to (2,4) \to (4,5) \to (5,2) \to (2,3) \to (3,1)$

If possible, select the vertex 1 as your starting point; otherwise, you must start at another vertex to complete the drawing.

Notice that a correct output is not unique, even if you select the same starting point.

- Need some Help?

Well if you are lost, you can try to run this pseudocode on Pictures 5 and 6:

```
Graph G;
Stack S;

verify that a solution exists;
select vertex u in G as the starting point and call Search(u);
print S as output if Search(u) returns true;

bool Function Search(u)
   S.push(u);
   if no edges available
      return true;
   else
      for all adjacent vertices v of u in graph G
         e = (u,v);
         delete edge e from G;
         f = Search(v);
         if f is true
            return true;
         else
            undo deleting: restore e by adding to G;
         end if
      end for
      S.pop();
      return false;
   end if
end Function
```

Answer the following questions:

- What is the status of the stack `S` at the end of the function `Search`?
- If we apply this pseudocode to a picture that doesn't actually have a solution, what will the output be?

Get a computer help!

- Write C++ code to find your solutions.
- In order to successfully complete this challenge, you need to use the graph data structure which helps you organize information about a picture in order to implement it in C++.
  - **Input**: an undirected connected graph $G(V,E)$, with all the vertices numbered.
  - **Output**: True/False
- The input pictures are stored in plain text files named `graph1.data`, ..., `graph6.data`. This is a sample of an input file for Picture 5:

```
5 6
1 2
2 3
3 1
2 4
4 5
5 2
```

The first line of the input file contains the number of vertices and the number of edges, respectively. The next lines contain the endpoints of edges. Parse the input file (`graph1.data`, `graph2.data`, ...) to obtain their computer graph representation using the sequence of the input edges. You can use the adjacent matrix in this assignment to store graphs.

- (20 points) **Part 1 of the assignment**: Implement a graph representation. It is due in the labs on April 17.

  1. Create a graph data structure using the adjacency matrix representation.
  2. Please read the textbook along with the supplementary material on the class webpage before you start writing C++ code for the graph data structure.
  3. Compile your program using the Linux machine command line:
     `c++ -std=c++14 *.cpp`
     or
     `make all`
  4. Run your program on each input file by executing
     `./main input_file`
     for example: `./main graph1.data`
  5. Display the output in the format *vertex name* followed by the *list of adjacent vertices* separated by ->. This is the illustration of the output for a graph corresponding to Picture 5.

     1 -> 2 -> 3
     2 -> 1 -> 3 -> 4 -> 5
     3 -> 1 -> 2
     4 -> 2 -> 3
     5 -> 2 -> 4

- (10 points) **Part 2 of the assignment**: Use this graph data structure and determine the possibility to draw a picture with one stroke. The algorithm should output True or False. Use the provided input files for testing.

- (5 points) If the algorithm outputs True, you need to find the starting vertex to draw a path. If possible, select the vertex 1 as the starting vertex; otherwise, you must start at another vertex to complete the drawing.

- (25 points) **Part 3 of the assignment**: You need to find the starting vertex and a path to draw the graph in one stroke. To check your program for correctness draw a graph based on the obtained output.

Skadoosh!

Congratulations! You are almost there. Just one last bit. To complete the challenge, you will need to submit the following.

1. (Total: 60 points) Your C++ source code with the header block including: your name, user name, section number and e-mail address. Submit your tar file to eCampus not later than April 28 by midnight.
2. (40 points) A report which should consist of the following parts:
   (a) The cover page with your (electronic) signature
   (b) The description of the data structures implemented in your program
   (c) The necessary and sufficient conditions for drawing one-stroke pictures.
   (d) Description of the algorithm and its running time.
   (e) The evidence of testing your program for correctness.
3. You will also be required to submit a hard copy of the report during your demonstration for this assignment in the labs on April 29th.