**Project report**

1) Queue.h

This file contains the generalized code for class queue. It has the following functions

queue()

Default constructor

bool isEmpty() const

Return true if the queue is empty

void enqueue(const T value)

Used to insert value in a queue after checking if its empty or not

T dequeue()

Returns the value from the queue of type T (templatized)

void makenull()

Used to clear queue by calling dequeue function repeatedly

2) Stack.h

This file contains generalized code for class stack with following functions:

stack()

default constructor

bool isEmpty()

return true if the stack is empty else return false

void Push(X val)

Function to push value at top of stack

void Pop()

Removes or returns value at top after checking if its empty or not

X Peak()

Returns the value at top of stack

3) LinkList.h

This file contains the code for singly linked list which is being used in our directed and undirected graph

List()

default constructor

Node\* getHead()

Returns the Node\* head of singly linked list

void add(int val)

This functions is used to add node at the end of singly linked

list after checking if it is the first node to be inserted or not

3) DirectedGraph.h

This file contains the formation of directed graph from the given data set as well as some other functions which are mentioned below.

Graph(int v)

This is the parametric constructor which gets the values of total no of nodes calculate in main cpp file.

Void addegde(int m, int v, int r)

Void addsrc(int l, int s)

This functions is used to insert the source value in the list made corresponding to the array.

Void print(int l, int checkvar)

This functions prints the adjacency list of the given node in the parameters

Void srchck (int until)

To check the no of source nodes existing in the whole file, this function is used. A source nodes exist if the indegree of the node is equal to 0.

Void sinkchck(int until)

To check the no of sink nodes which means a node whose out degree is 0.

Void idegremaker(int n)

This function goes through the adjacency list of every node and calculates the indegree.

int maxc(int n)

The function return the maximum indegree from the total no of nodes.

Void isolatedchck(int n)

This functions displays the no of isolated nodes in the whole file. an isolated node is formed if it’s in degree and out degree are both 0.

Void DFS(int s, int asize)

This function is the basic depth first search function which is implemented on the total no of nodes using stack.

Int\* cbfs (int s, int asize)

This function returns an array containing the no of nodes that can be visited using this function of the given node as mentioned in the parameters.

Int\* in()

The in algoritham takes up a source and then perform DFs on every vertex in the datset. If source exist in the returning array then we add that vertex.

Int\* out()

It just takes bfs of every vertex.

Int ssc()

Sc algritham performs in and out algo on every vertex and then find their intersection and returns the final array.

4) source.cpp

The is the main cpp file through which the code is running. Initially a few global variables are declared and the remaining functions are as followed.

Void color()

A function used to set the color of text in the terminal window.

Void initializer(int val)

This function acts as a parametric constructor which assigns the values to the dynamically constructed directed and undirected graph as well as an array.

Void display()

For displaying menu as well as choices for the user initially.

Void displaynodes()

The basic function to display the no of nodes which is declared globally.

Void displayedges()

The function used to display the no of edges .

Void indegreetable(int index, int degree)

This is a helping function for printing indegree distribution which is linked to its other function mentioned below. It prints the index number as well as its probability distribution.

Void forindegree(int val)

This functions is used to calculate in degree distribution for all the vertices of directed graph. It calls the degreemaker function which formulates the degree for every vertex. We used the maxc function to get the maximum value for the running of the loop. The probability is found by dividing the count with the total no of nodes which are 5242 here.

Void outdegreetable(int index, int degree)

This is a helping function for printing outdegree distribution which is linked to its other function mentioned below. It prints the index number as well as its probability distribution.

Void foroutdegree(int val)

This functions is used to calculate out degree distribution for all the vertices of directed graph. We used the maxc function to get the maximum value for the running of the loop. The probability is found by dividing the count with the total no of nodes which are 5242 here.

Void start()

The function plays an important role in file fetching as well as implementing our basic logic of making graphs and edges.

We fetched every line and tokenized it to get the source and destination and further made the edges of graphs using certain Boolean variables and functions.

Int main()

It calls the display function to show the menu as well as the start function which makes the graphs. A switch is implemented to call every function according to the menu.

5) Undirected.h

Bdfs()

A specialized function for finding bridge nodes.

Adfs()

For finding articulation nodes

Wcc()

A function that finds largest WCC by just taking bfs on every vertex.