Abeera Fatima
I180411
E

Question 6

Algorithm:

1. Create a graph with adjacency lists.
   Make an array of vertices. In each vertex if there is an edge leading from it, the destination vertex is added to that vertex's adjacency list. A vertex is a structure which records the Graph's Vertex's number and has a pointer for all destination vertexes (this makes the adjacency list).
2. Create 1 Boolean array to store the visit status of the vertices, which will be the size of the number of vertexes. Initialize it as all false.
3. Create a stackn to keep track of all neighbor vertices

   bool visited [V]; //will keep track of the visited vertices
   Stack Stackn; //Will keep track of all the neighbors

   for(int x = 1; x <= V ; x++)
   {
   visited [x]=0;

   }
4. Create a function isCycle(){
   for(int x = 1; x <= V ; x++)
   {
   Mark the position of the vertex true in the visited array[x] =1;
   Stack.push(x);

   While(!stack.isEmpty())
   {
   Then push all the neighbors of the stack.top() in the stack by traversing through the adjacency list of the graph.
   Stackn.pop(); //pop the starting vertex

   At each push traverse through the visited array to check if the vertex is true in the visited array;

   If (vertex is true in the array) //cycle exists
   Return true
   else
   Mark the pushed vertex as true in the visited array.
   }
   }

}

This algorithm is basically using the idea of Breadth First Search to find cycles.

The searching algorithm is applied from all vertices to see if a cycle exists at any point.