Abeera Fatima
I180411
E

Question 4

Proof by Strong Mathematical Induction as done in Discrete Mathematics:

Let n be the number of vertices in the tournament.

When the vertices are less than or equal to two, a Hamilton path exists as the graph is connected. If arbitrary directions are given to the connection, they will lead either to or fro from one point to the other thus a Hamilton path exists.

If the vertexes are more than two, select a vertex at random (v) and partition the rest into two groups. One set contains all the vertexes, which have an edge leading out from v (Vout) to another vertex u and the other set contains the vertexes, which have an edge leading into v (Vin) from the vertex u.

Both sets will obviously contain less vertexes than the total number of vertexes (n) i.e. |Vout|<n and |Vin|<n. And, thus by the inductive hypothesis there exists a Hamilton path in each of the sets. Now using the two sets and the vertex v we can make a combined Hamilton path including all vertexes.

Algorithm to find a Hamilton Path with Backtracking:

Take a 2D array[V][V] which is the adjacency matrix of the graph.

There will be 1s in the place where is a direct edge from (i,j), otherwise add zeros.

```
int Circuit[v];

for ( x = 1 ; x <= V ; x++ )

Circuit[x] = -1;
```

Pick an arbitrary vertex to check if the circuit exists and add it to the Circuit array.

```
Circuit[1]=1; //changing the value from -1 to 1, the number of the vertex. Vertex 1 was selected in this case.

int vertex_added=1// this will check if all vertices are added to the circuit array

if (vertex_added == V)

{
        Check if the is an edge from the last included vertex to the first vertex.

        //This will be done by checking if there is a one in the graph array at

        if (Graph[Circuit[1]][circuit[vertex_added]]==1)

        {       cout<<"Solution Exists<<endl;

                for ( x = 1 ; x <= V ; x++ )

                cout<<Circuit[x]<<endl;
```

```
        }


        else
        cout<<"Circuit Does Not Exist<<endl;
}
 //Otherwise we will add more vertices to the Circuit
for ( x = 2 ; x <= V ; x++ )     //Vertex 1 already added
{
        //Check first if the new vertex to be added to the circuit is adjacent to the last vertex
        added.
        //This again is done by checking the graph
        If(graph[vertex_added][x]==1) // x is the new vertex to be added
        {
                //Check if it was already in the array before
                for ( y = 1 ; y <= vertex_added ; y++ )
                {
                        If(Circuit[y] = =x) //was found in the array
                        {
                                Circuit[vertex_added]= - 1;// adding the vertex does not help
                                the solution so it is skipped. And the next vertex will be
                                checked.
                        }
                        Else
                        Circuit[vertex_added++]=x;
                }
        }
}
```