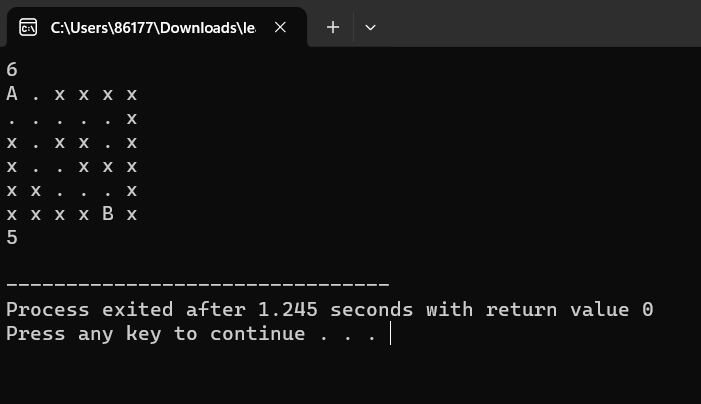
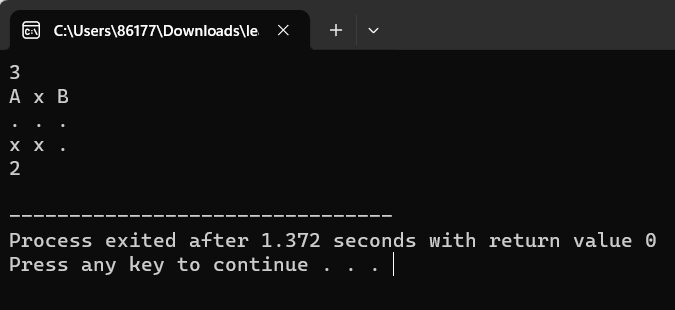
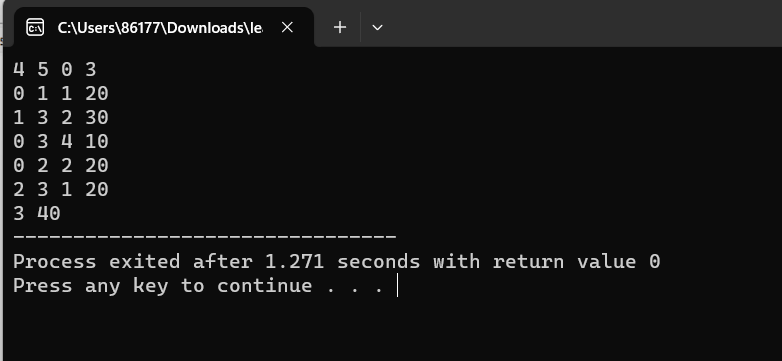
本周task1难度较大，特别是其中对于转弯的判定 参考了csdn相关专栏的代码并且请教了助教老师 剩余三题主要考察了dijkstra算法。

Task1

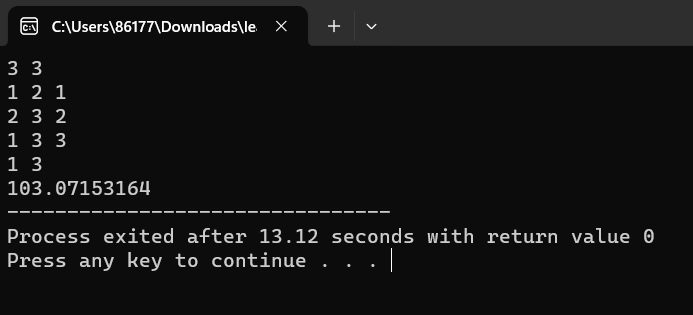




Task2



Task3



Task4

#define INF 1000

class Solution {

public:

    int networkDelayTime(vector<vector<int>>& times, int n, int k) {

        int map[n+1][n+1];

        bool visited[n+1];

        int dist[n+1];

        for(int i=0;i<=n;i++) dist[i]=INF;

        for(int i=0;i<=n;i++) visited[i]=false;

        for(int i=0;i<=n;i++){

            for(int j=0;j<=n;j++) map[i][j]=-1;

        }

        for(int i=0;i<times.size();i++){

            map[times[i][0]][times[i][1]]=times[i][2];

        }

        dist[k]=0;

        while(true){

            int v=-1;

            int min=INF;

            for(int i=0;i<=n;i++){

                if(!visited[i]){

                    if(dist[i]<min){

                        v=i;

                        min=dist[i];

                    }

                }

            }

            if(v==-1) break;

            visited[v]=true;

            for(int i=0;i<=n;i++){

                if(!visited[i]&&map[v][i]!=-1){

                    if(dist[i]>dist[v]+map[v][i]){

                        dist[i]=dist[v]+map[v][i];

                    }

                }

            }

        }

        int max=0;

        for(int i=1;i<=n;i++){

            if(dist[i]==INF){//the node hasn't been visited

                return -1;

            }

            else{

                if(dist[i]>max) max=dist[i];

            }

        }

        return max;

    }

};

