1) A. bool pathway[8] = {[0] = true, [2]=true};
B. bool pathway[8] = {true, false, true};

```
#define SIZE 8

int main(void)
{
    int start, counter;
    //initialization of array
    int road_networks[SIZE][SIZE] = {
        //use designated initialization to set specific index values to 1 while else is 0
        {[0]=1, [1]=1, [5]=1},
        {[0]=1, [1]=1, [2]=1},
        {[1]=1, [2]=1, [4]=1, [5]=1},
        {[3]=1, [4]=1},
        {[3]=1, [4]=1},
        {[0]=1, [2]=1, [5]=1},
        {[0]=1, [3]=1, [6]=1},
        {[5]=1, [7]=1}
    };
```

2)

```
    //initialization of necessary parameters to be used in the process
    int i = start;
    int j =0;
    //while the point does not arrive at a charging station, the loop continues
    while (i!=3 && i!=2 && i<SIZE)
    {
        //if the point does have a 1, then that becomes the new starting point
        if (road_networks[i][j] == 1)
        {
            i = j;
            j++;
        }
        else{
            j++;
        }
    }

    //printing of final output
    if (i == 2)
    {
        printf("\npoint: C arrived to charging station");
    }
    else if (i == 3)
    {
        printf("\npoint: D arrived to charging station");
    }
    else{
        printf("\npoint: No charging station near");
    }
```

I believe that these two parts are the most important in the whole code of the number 2 question. Firstly, in the first photo, the main process of the code is to declare and initialize the adjacency matrix given in the question. What I did is to simply use designation initialization to initialize the value of 1 to the corresponding indexes inside the adjacency matrix. In the second photo, the main process involved is to evaluate each point whether it can meet the points C and D, which are charging stations. As you see, I initialized a start variable (Note: the start variable is the input of the user or the starting point) and a j variable. Next is I wrote a while condition, wherein unless the point is at 3 or 2, which are the corresponding points for the charging stations, or it surpasses the maximum index, then the loop will not exit. Inside that loop is an if condition, where I check whether the starting point does have a route to a charging station or not. And if it does, then the next point in the route would then become my new starting point. I repeat this until I arrive at a charging station point, which is what I wrote in the while condition. Also, if the maximum index is surpassed and the code still does not arrive at points C and D, then the loop will also exit. Then after I exit the while loop, I then print the corresponding output depending on the last point the loop was on.

Github Link: https://github.com/ainzzcutie/CMSC21.git