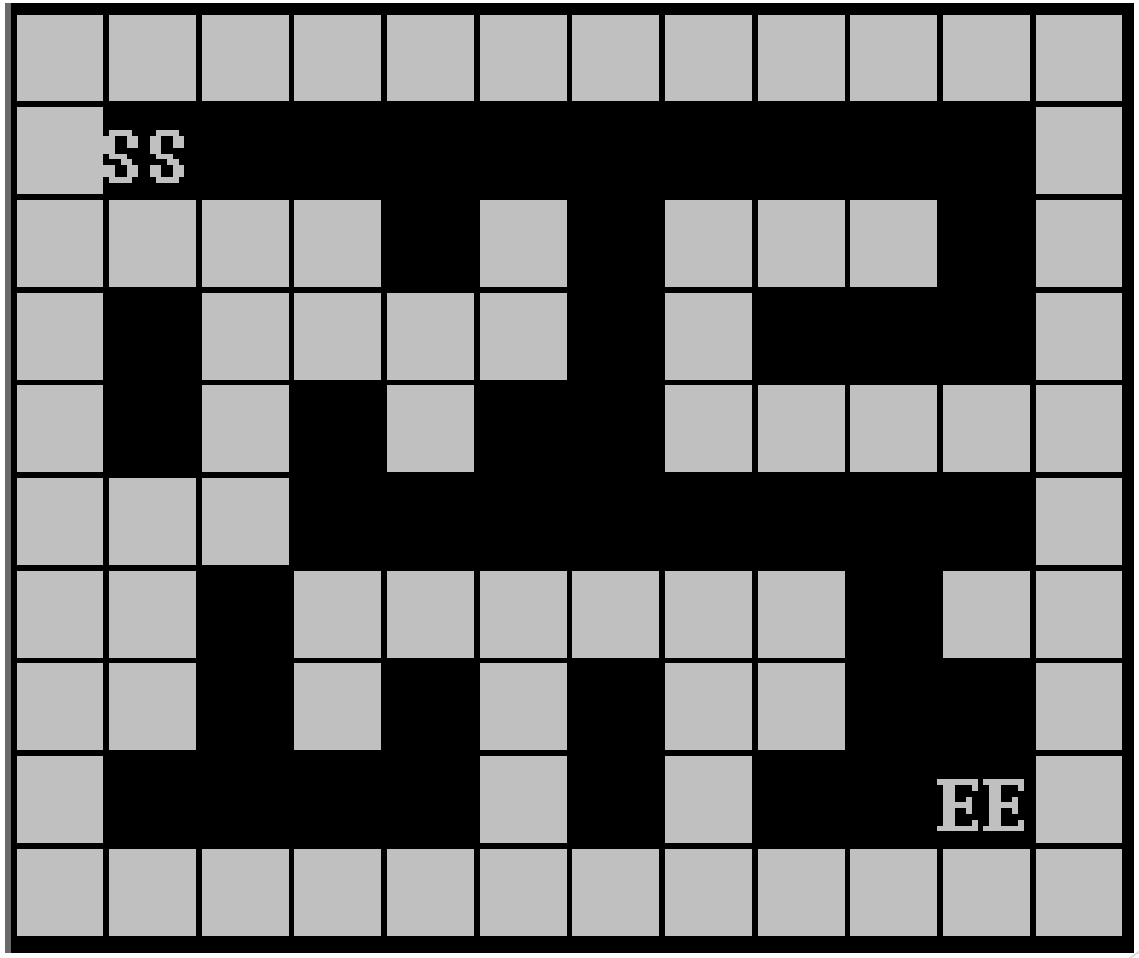
Maze Miner

Speaker: Wei-Chian Wang

Department of Electrical Engineering

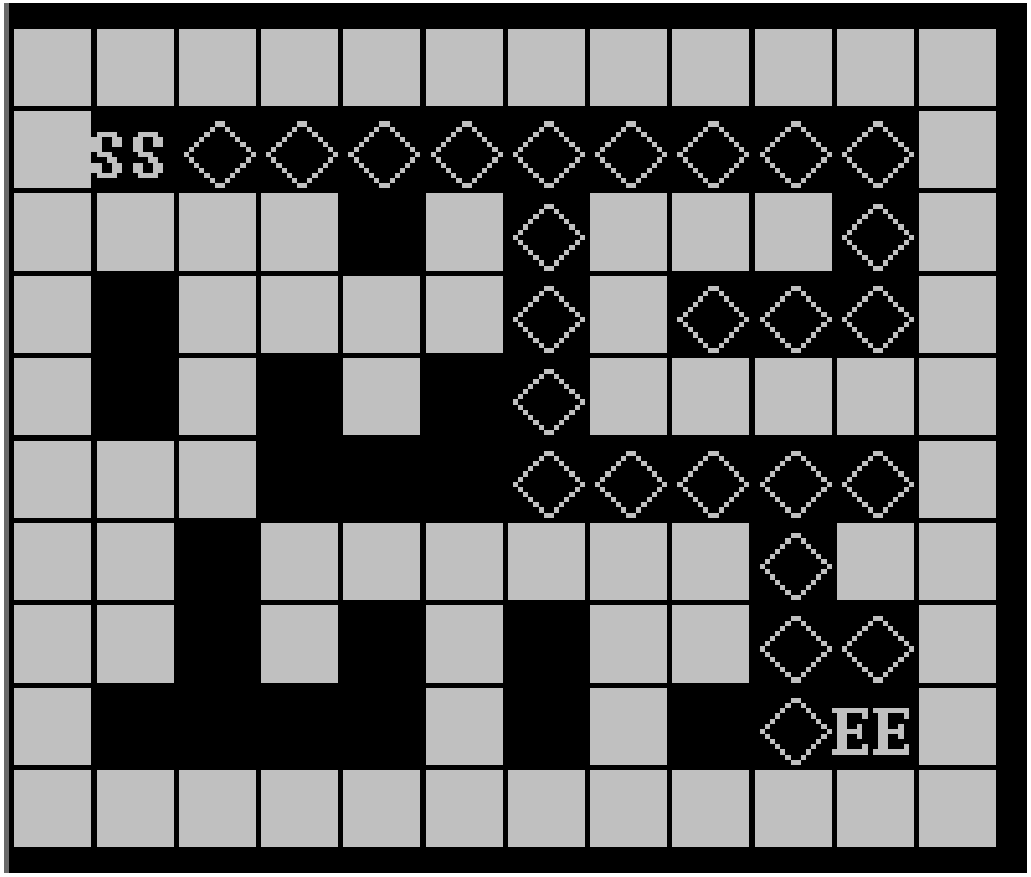
National Cheng Kung University

Example of Maze

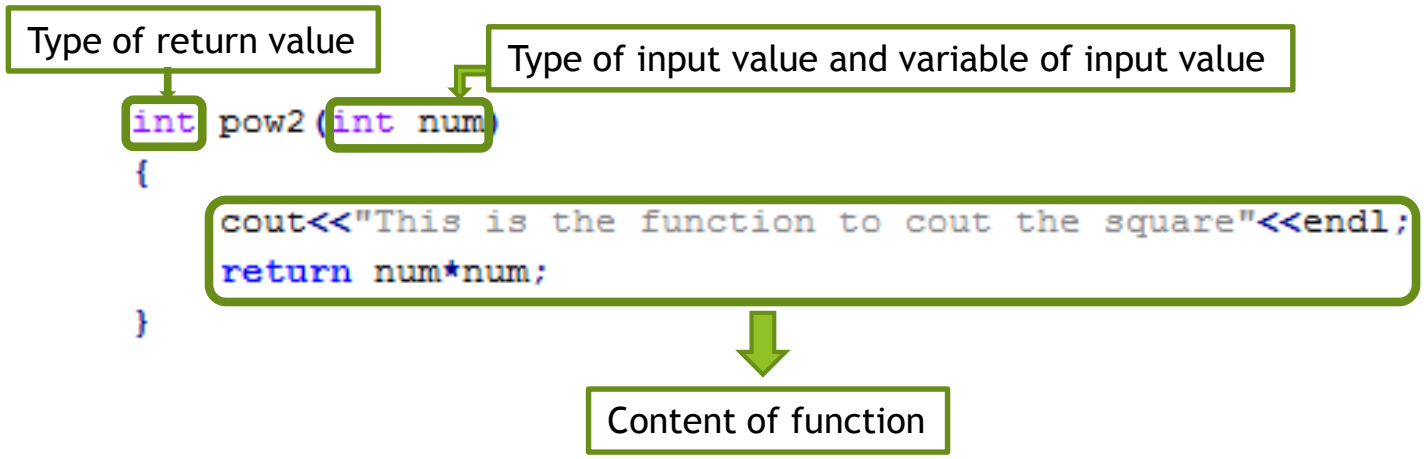


Result of Maze

- Find the path from start point to destination point



Example of Function



Complete Example of "Function"

```
#include <iostream>

using namespace std;

int pow2(int num);

void main()
{
    int i;
    int result;

    cout<<"This is the program to count the square value."<<endl;
    cout<<"Please input a number: ";

    cin>>i;
    result = pow2(i);

    cout<<"The square value is: ";
    cout<<result<<endl;

    system("PAUSE");
}

int pow2(int num)
{
    cout<<"This is the function to count the square"<<endl;
    return num*num;
}
```

```
This is the program to count the square value.
Please input a number: 5
This is the function to count the square
The square value is: 25
請按任意鍵繼續 . . .
```

2-Dimensional Array

- 2 dimensional array

(0,1)=Array[0][1]
(2,2)=Array[2][2]

row\col	0	1	2
0	Array[0][0]	Array[0][1]	Array[0][2]
1	Array[1][0]	Array[1][1]	Array[1][2]
2	Array[2][0]	Array[2][1]	Array[2][2]

- 1 dimensional array

(1,2)=Array[5]= Array[3*1+2]
(2,2)=Array[8]= Array[3*2+2]

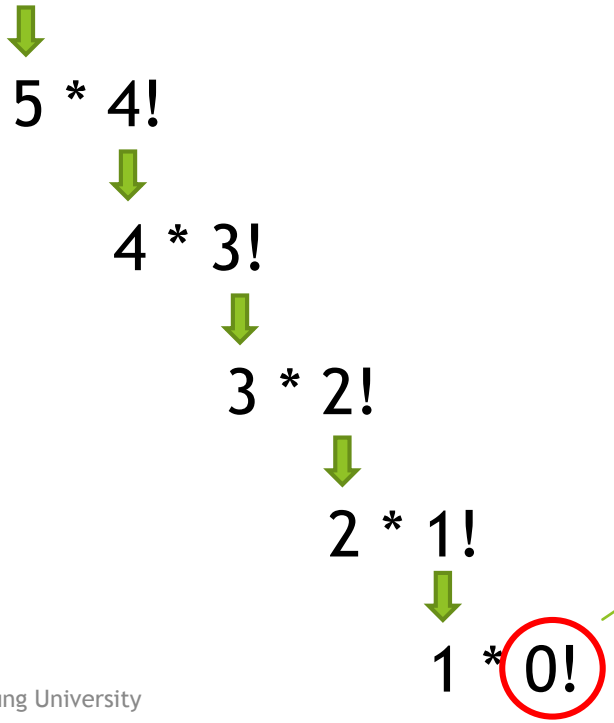
row\col	0	1	2
0	Array[0]	Array[1]	Array[2]
1	Array[3]	Array[4]	Array[5]
2	Array[6]	Array[7]	Array[8]

Recursion

- Factorial

$6! = 6 * 5 * 4 * 3 * 2 * 1$

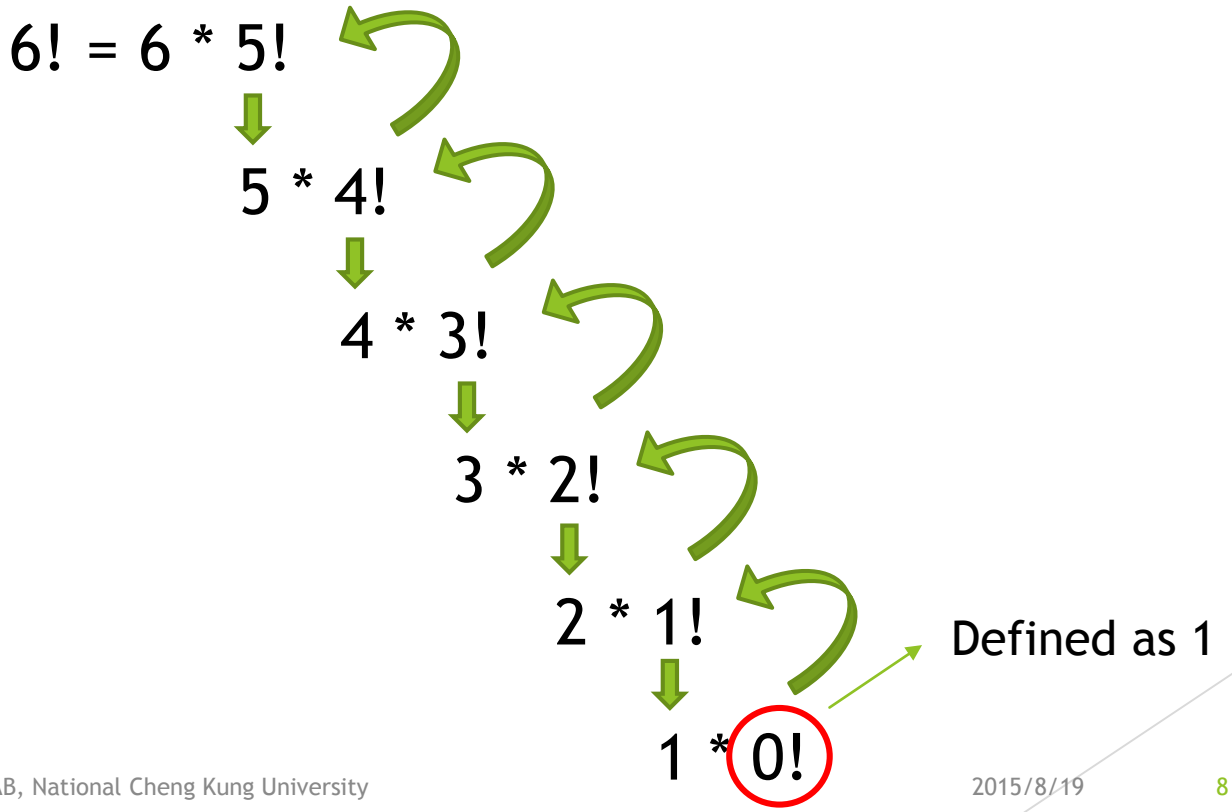
$6! = 6 * 5!$



Recursion

- Factorial

$6! = 6 * 5 * 4 * 3 * 2 * 1$



Recursion

```
int fac(int n)
{
    if(n == 0)
    {
        return 1;
    }
    else
    {
        return n * fac(n-1);
    }
}
```

How to Use Function to Finish the Maze Miner

- 0: Road
- 1: Wall
- 2: Start Point
- 3: Destination Point

- Write the code in the “visit” function

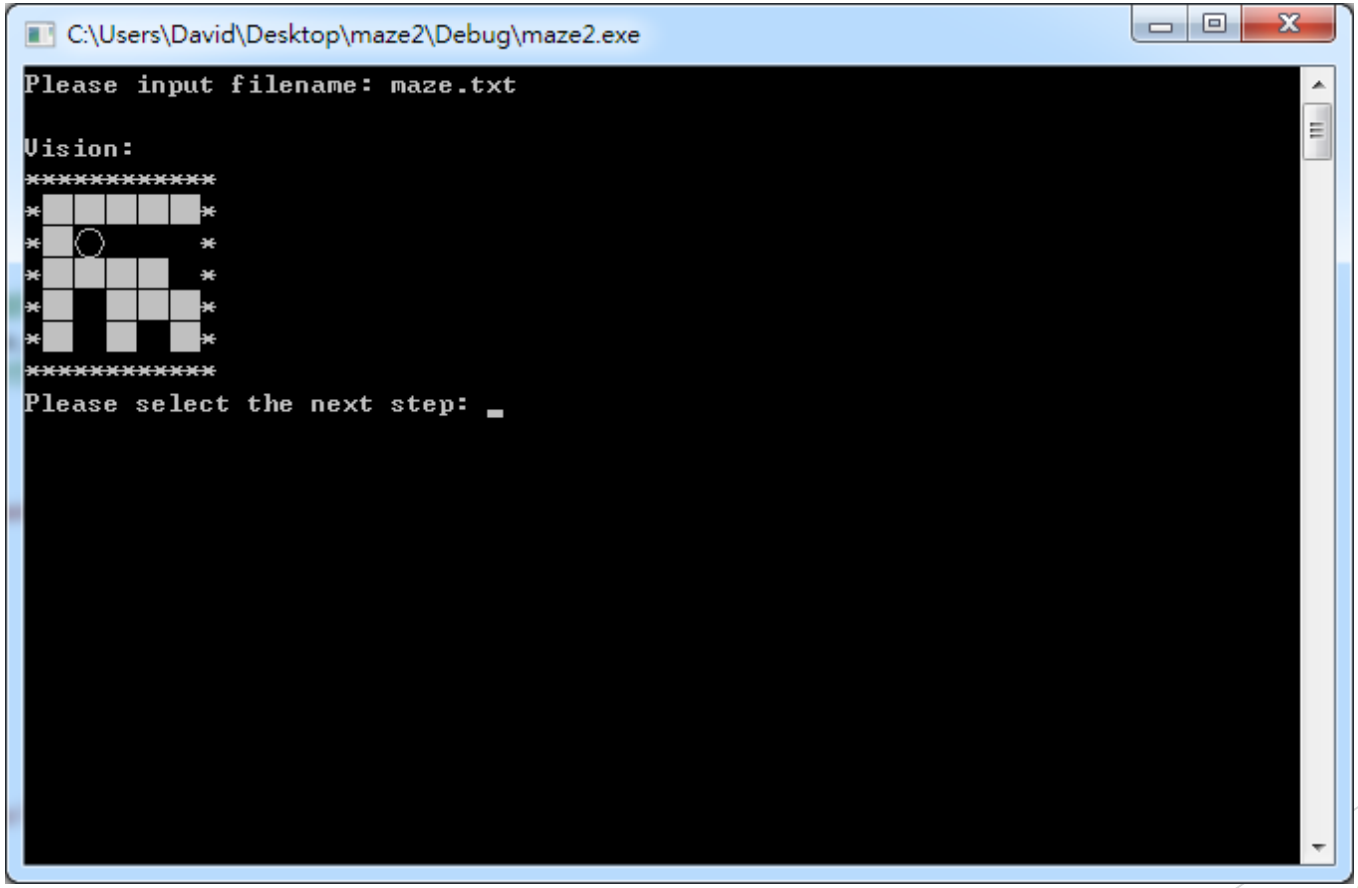
1. Check each direction can move or not
2. Move one step each time
3. Finish the program when location is at destination point

How to Use Function to Finish the Maze Miner

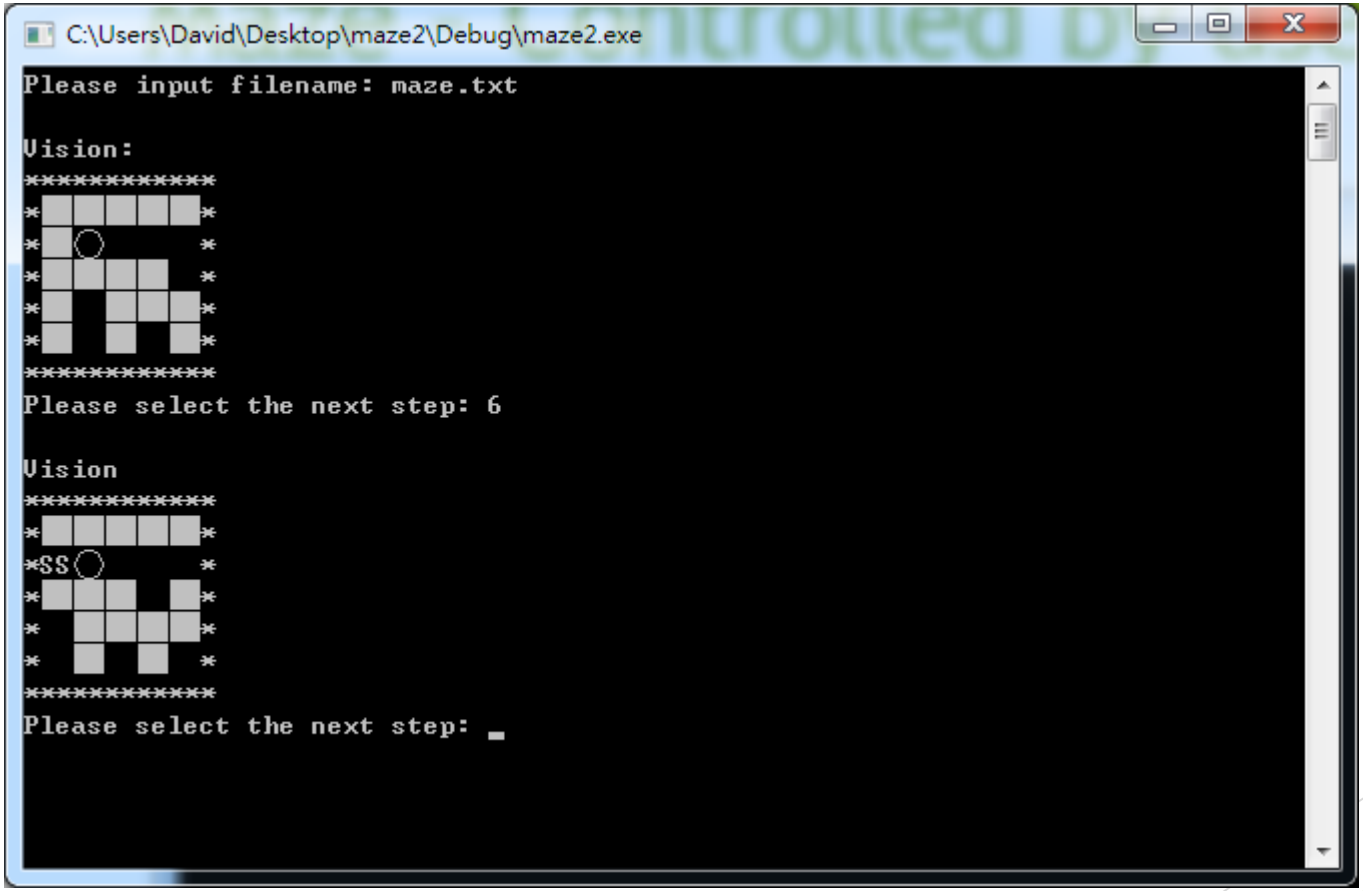
```
//------------------拜訪座標(i,j)的function----------

void visit(int i, int j)
{
    num[i][j] = 4; // (i,j)設置為4，代表拜訪過

    //到達終點之後印出地圖
    if(i == dest_row && j == dest_col)
    {
        cout << endl << "Show the path：" << endl;
        for(int m = 0; m < row;m++)
        {
            for(int n = 0; n < colume; n++)
            {
                if(num[m][n] == 1)
                    cout << "■";
                else if(num[m][n] == 4)
                    cout << "◇";
                else
                    cout << "  ";
            }
            cout << endl;
        }
        stop=1; //stop令為1，visit函式不再繼續拜訪
    }

    //判斷右方是否可以走，可以則拜訪
    if(??????????????)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i << "," << j+1 << ")" << endl;
        ??????????????  //拜訪右方格子
    }

    //判斷下方是否可以走，可以則拜訪
    if(??????????????)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i+1 << "," << j << ")" << endl;
        ??????????????  //拜訪下方格子
    }

    //判斷左方是否可以走，可以則拜訪
    if(??????????????)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i << "," << j-1 << ")" << endl;
        ??????????????  //拜訪左方格子
    }

    //判斷上方是否可以走，可以則拜訪
    if(??????????????)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i-1 << "," << j << ")" << endl;
        ??????????????  //拜訪上方格子
    }
}
```

```
    //判斷右方是否可以走，可以則拜訪
    if(num[i][j+1] == 0 || num[i][j+1] == 3 || num[i][j+1] == 2)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i << "," << j+1 << ")" << endl;
        visit(i, j+1);  //拜訪右方格子
    }

    //判斷下方是否可以走，可以則拜訪
    if(num[i+1][j] == 0 || num[i+1][j] == 3 || num[i+1][j] == 2)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i+1 << "," << j << ")" << endl;
        visit(i+1, j);  //拜訪下方格子
    }

    //判斷左方是否可以走，可以則拜訪
    if(num[i][j-1] == 0 || num[i][j-1] == 3 || num[i][j-1] == 2)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i << "," << j-1 << ")" << endl;
        visit(i, j-1);  //拜訪左方格子
    }

    //判斷上方是否可以走，可以則拜訪
    if(num[i-1][j] == 0 || num[i-1][j] == 3 || num[i-1][j] == 2)
    {
        if(stop==0)
            cout << "(" << i << "," << j << ") -> (" << i-1 << "," << j << ")" << endl;
        visit(i-1, j);  //拜訪上方格子
    }
}
```

Maze – Controlled by user



Maze – Controlled by user



Maze – Controlled by user

```
//宣告目前位置的行列
int position_row = 0;
int position_column = 0;
```

```
//印出可視範圍的function
void print_maze(int position_row, int position_column)
{
    cout<<endl;
    cout<<"Vision"<<endl;

    cout<<"************"<<endl;
    for(int i=position_row-1;i<=position_row+3;i++)
    {
        cout<<"*";
        for(int j=position_column-1;j<=position_column+3;j++)
        {
            if(num[i][j] == 1)
                cout << "■";
            else if(i==position_row&&j==position_column)
                cout << "○";
            else if(num[i][j] == 2)
                cout << "SS";
            else if(num[i][j] == 3)
                cout << "EE";
            else
                cout << "  ";
        }
        cout<<"*"<<endl;
    }
    cout<<"************"<<endl;
}
```

Maze – Controlled by user

```
//開始進行玩家操作
//8:上
//6:右
//2:下
//4:左
while(1)
{
    int next_step;
    cout<<"Please select the next step: ";
    cin>>next_step;

    if(next_step==6) //判斷往右走
    {
        if(num[position_row][position_column+1]==1) //右邊是牆 不能走
        {
            cout<<"Do not across the wall!!!"<<endl;
            ????????????? //跳過這次loop
        }
        if(?????????????) //右邊不是牆 可以走
        {
            ????????????? //往右走
            ????????????? //印出目前可視範圍
        }
    }
```

Maze – Controlled by user

```
if(??????????????) //判斷是否抵達終點
{
    cout<<"Congratulations!!! You arrive at destination"<<endl;
    ?????????????? //跳出整個loop
}
```

Maze – Controlled by user

```
//開始進行玩家操作
//8:上
//6:右
//2:下
//4:左
while(1)
{
    int next_step;
    cout<<"Please select the next step: ";
    cin>>next_step;

    if(next_step==6) //判斷往右走
    {
        if(num[position_row][position_column+1]==1) //右邊是牆 不能走
        {
            cout<<"Do not across the wall!!!"<<endl;
            continue; //跳過這次loop
        }
        if(num[position_row][position_column]!=1) //右邊不是牆 可以走
        {
            position_column++; //往右走
            print_maze(position_row, position_column); //印出目前可視範圍
        }
    }
```

Maze – Controlled by user

```
if(position_row==dest_row&&position_column==dest_col) //判斷是否抵達終點
{
    cout<<"Congratulations!!! You arrive at destination"<<endl;
    break; //跳出整個loop
}
```