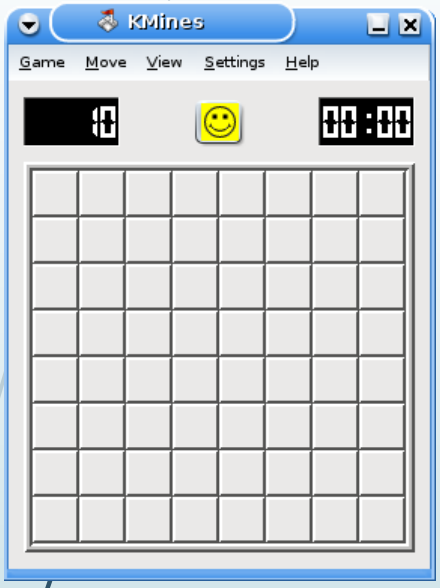
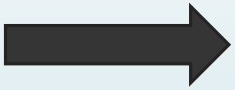
Design a Minesweeper

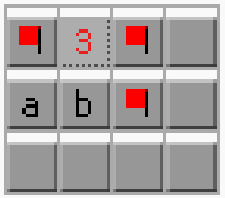
Knowledge and Information Discovery Lab

The Rules of Minesweeper



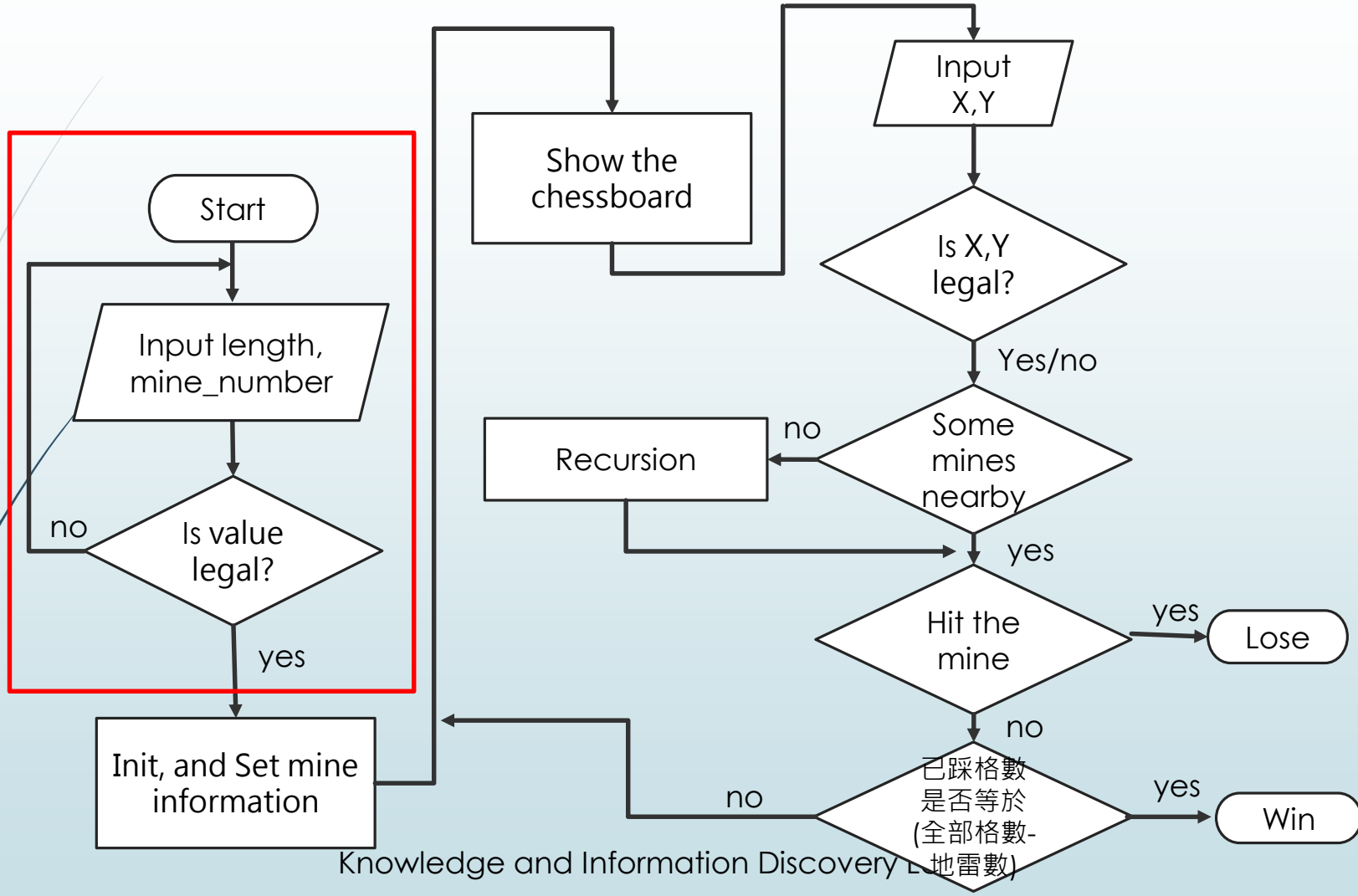


範例1



a和b沒有地雷，可輕易點開，因為已有3顆地雷滿足數字3

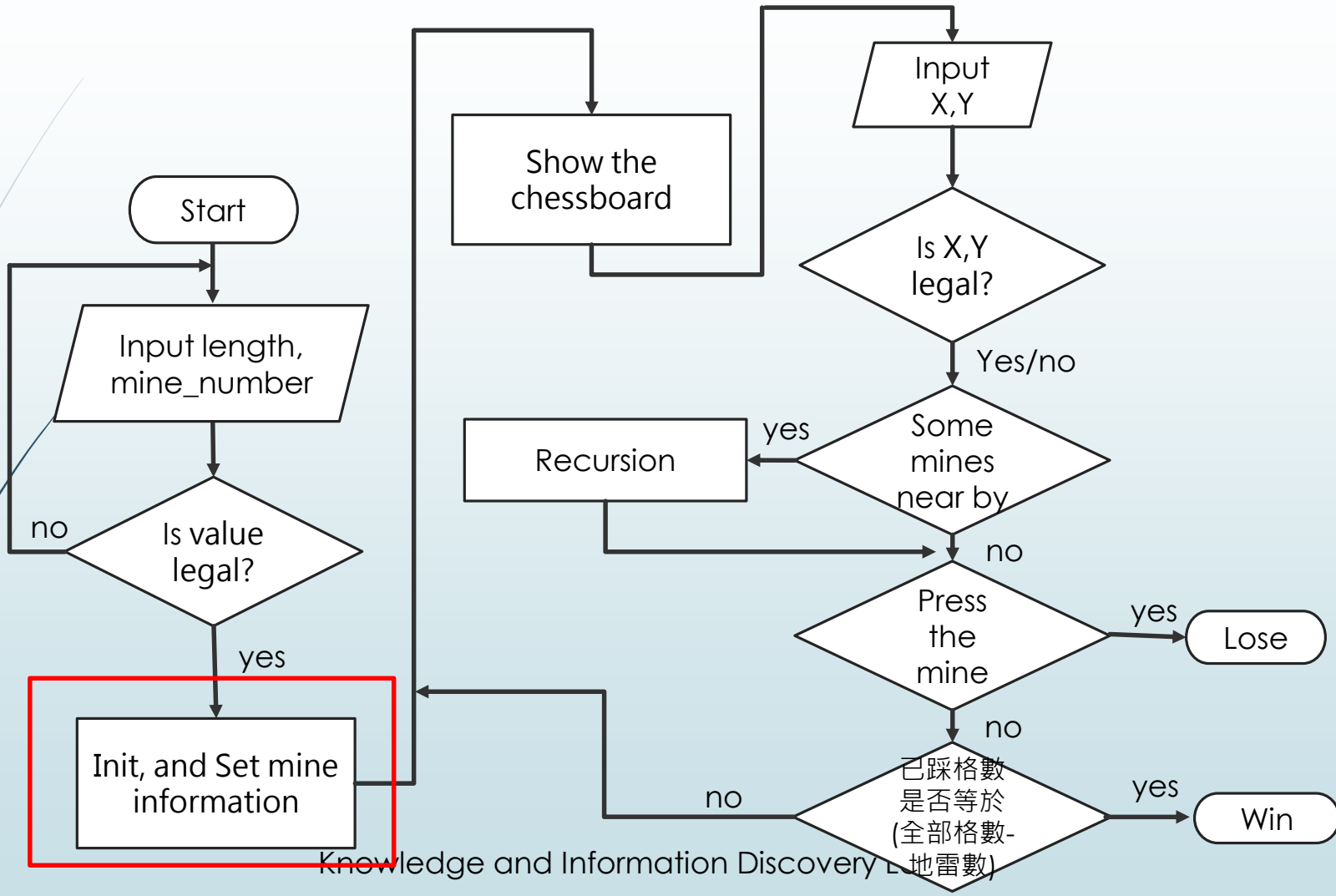
Flow chart


Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the chessboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines nearby
no
Recursion
yes
Hit the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

Start() function


```
void Start(){           //start game
    do{
        input_error = 0;
        cout << "Enter the board length  :";
        cin >> boardlength;
        cout << "Enter the mine number:" << "(" << "0~" << boardlength*boardlength << ")  :";
        cin >> mine_number;
        /*1111111111111111111111111111111111111test1111111111111111111111111111111111
        please fill the blank.
        When user's input value is illegal, then the process will ask user to retry.
        Hint: using the if() and the argument "input_error"
        */
        system("pause");
        system("cls");//clean screen
    } while (input_error);
    Init();
    bool coutinue = 1;
    while (coutinue){
        ShowMineboard(2);
        Press();
        coutinue = Check();
    }
    return;
}
```

Flow chart


Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the chessboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

Init() function


```
void Init(){                                    //initialize
    int counter = 0;
    int i, j;
    for (i = 1; i <= boardlength; i++){   //initialize the mineboard
        for (j = 1; j <= boardlength; j++)
        {
            mine_info[i][j] = 0;
            map_press[i][j] = 0;
        }
    }
    for (i = 0; i <= boardlength + 1; i++){   //initialize the wall
        mine_info[i][0] = -2;
        mine_info[i][boardlength + 1] = -2;
    }
    for (j = 0; j <= boardlength + 1; j++){   //initialize the wall
        mine_info[0][j] = -2;
        mine_info[boardlength + 1][j] = -2;
    }
```

Init() function


```
    srand(time(NULL));
    //2222222222222222222222222test22222222222222222222222222222222222222
    /*Please compelete the following code.
    Set n mines randomly on location (x, y), n = mine_number.
    Let x and y belong to 1 ~ boardlength.
    Hit: using the argument "counter" and %.
    The mine_info[][] can express the mine.
    */
    /*
    while (){  //control the mumber of mines
        int x = rand();
        int y = rand();
        if (){

            SetMineLable(x, y);  //do not change this line!!

        }
    }
    */
    //2222222222222222222222222test22222222222222222222222222222222222222
}
```

mine_info[x][y]

-1 there's a mine on (x, y), 0 there is no mine around, >0 how many mines around, -2 is the wall

int map_press[x][y]

=0 unstepped,=1 stepped

Wall	Wall	Wall	Wall	Wall	Wall	Wall
Wall	(1,1)					Wall
Wall						Wall
Wall						Wall
Wall						Wall
Wall						Wall
Wall	Wall	Wall	Wall	Wall	Wall	Wall

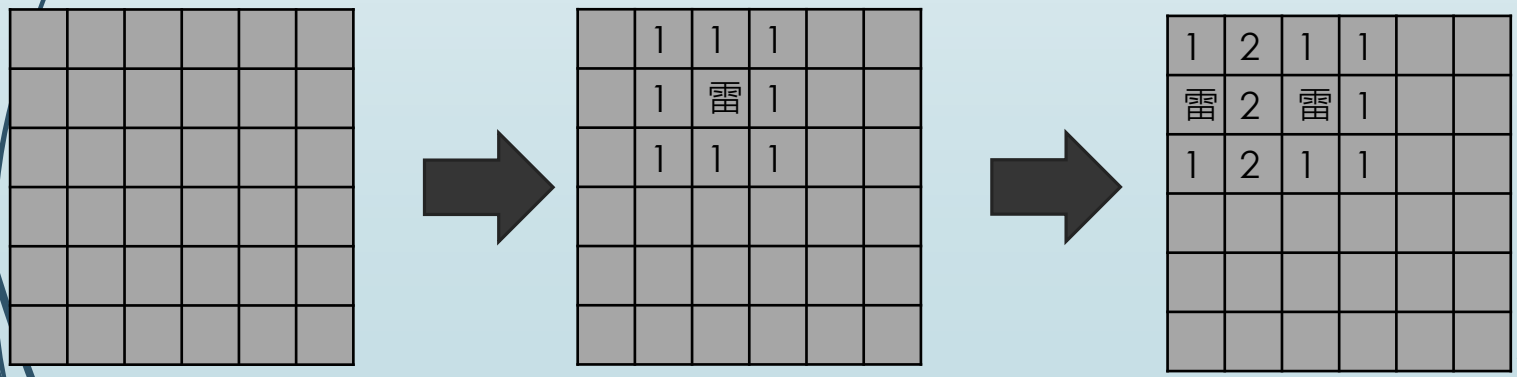
MineBoard

(boardleagth, boardleagth)

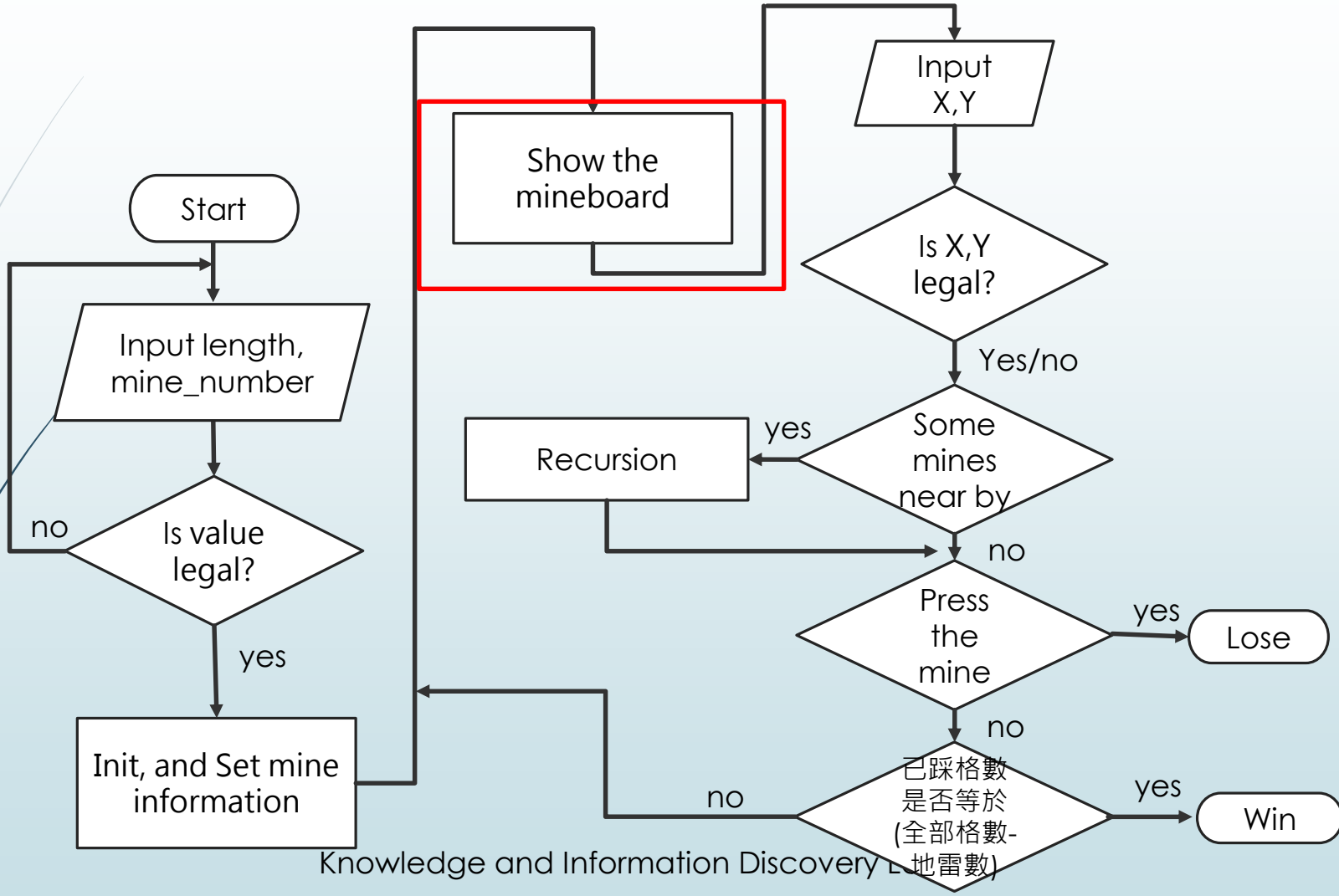
SetMineLable() function


```
void SetMineLable(int x, int y) //update how many mines around
{
    //333333333333333333333test33333333333333333333
    /* check 8 location that around the (x, y),
    if the location is not mine and not wall then update the mine number
    Hit: using the mine_info[][]
    */


    //333333333333333333333test33333333333333333333
}
```



Flow chart

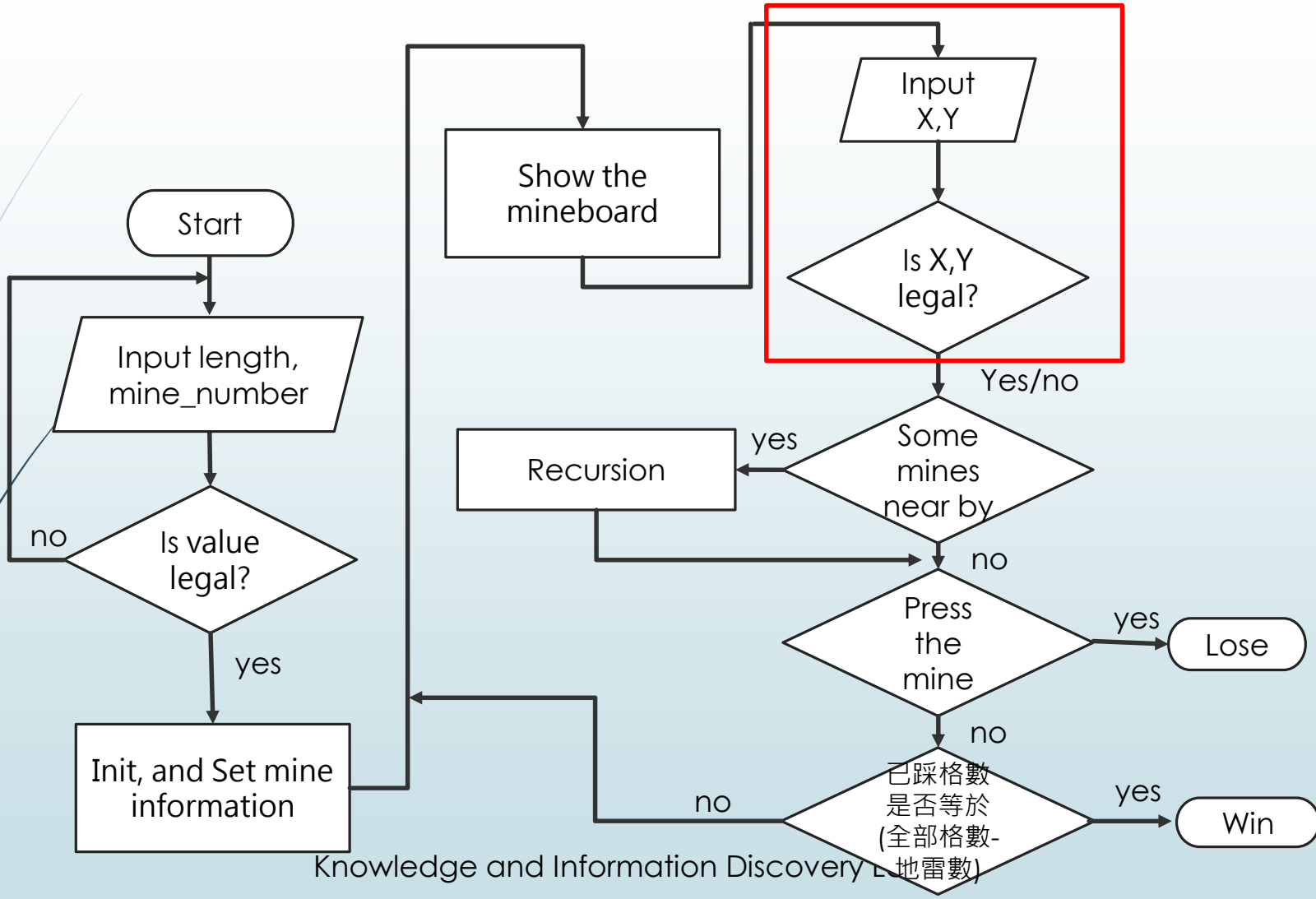

Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the mineboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

ShowMap() function

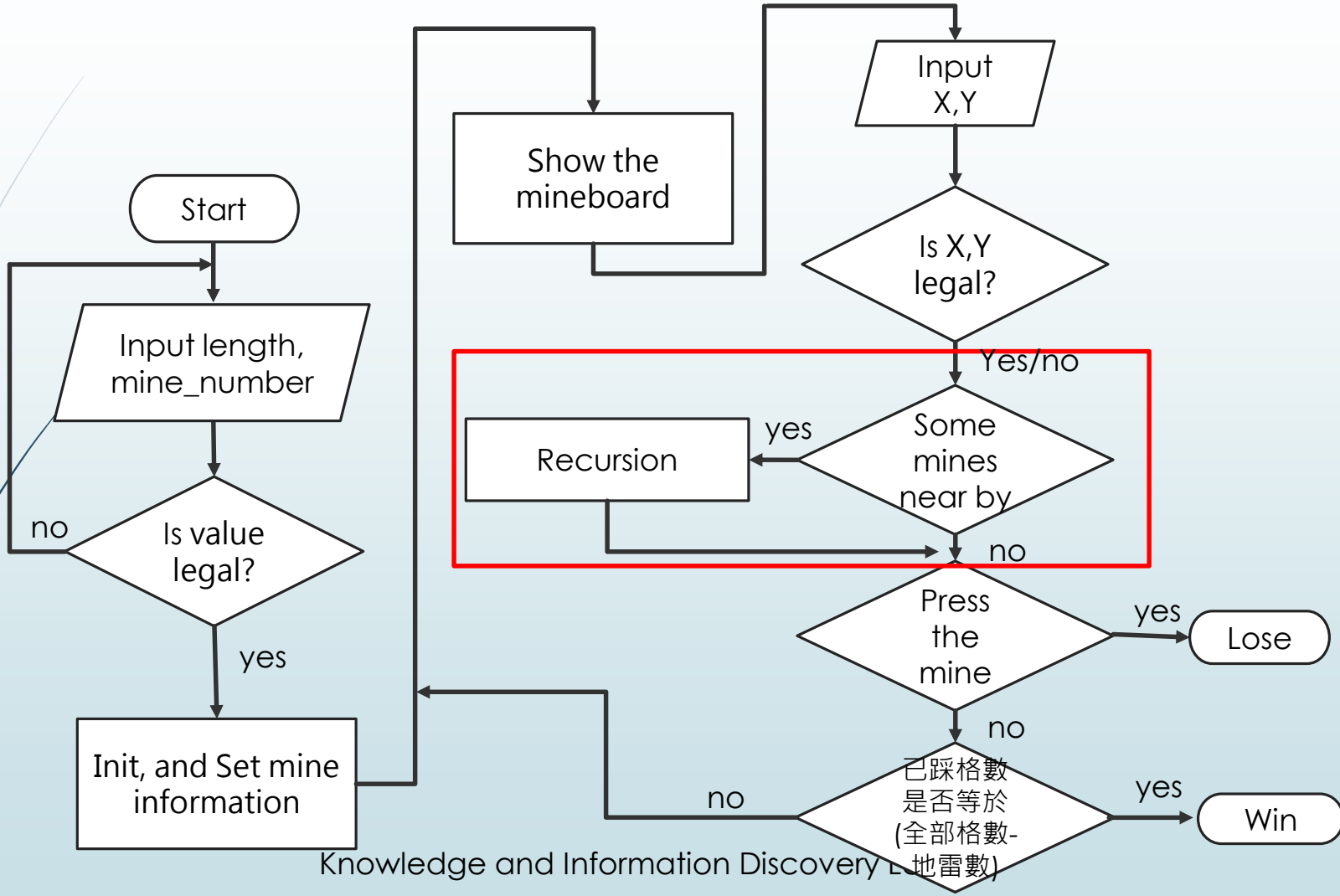
```
void ShowMineboard(int win){   //show the mineboard

    cout << "(y) ";
    for (int i = 0; i < boardlength; i++){
        cout << "__";
    }
    cout << "\n";
    for (int j = boardlength; j > 0; j--){
        if (j > 9)cout << j << " |";
        else cout << " " << j << " |";
        for (int i = 1; i <= boardlength; i++){
            if (map_press[i][j]){
                if (mine_info[i][j] == -1 && win != 1){
                    cout << "爆";
                }
                else if (mine_info[i][j] == -1 && win == 1){
                    cout << "雷";
                }
                else if (mine_info[i][j] == 0)  cout << "  ";
                else  cout << " " << mine_info[i][j];
            }
            else
                cout << "■";
        }
        cout << "| " << '\n';
    }
```

Flow chart

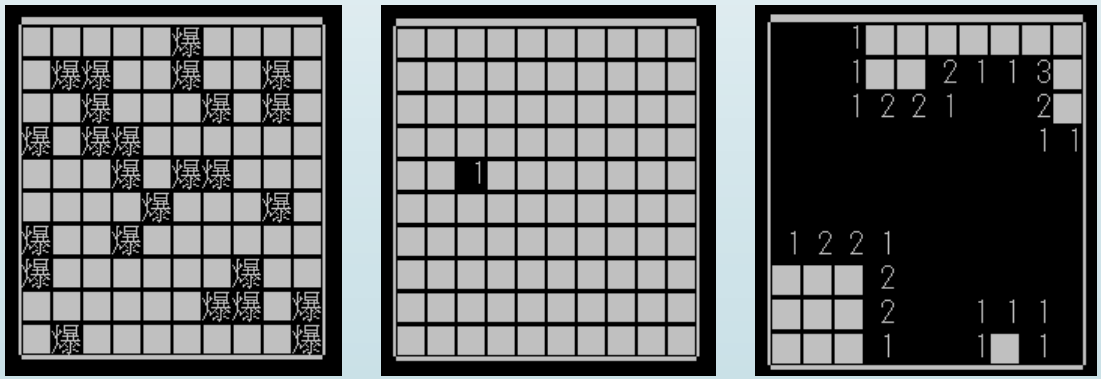

Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the mineboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

Flow chart


Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the mineboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

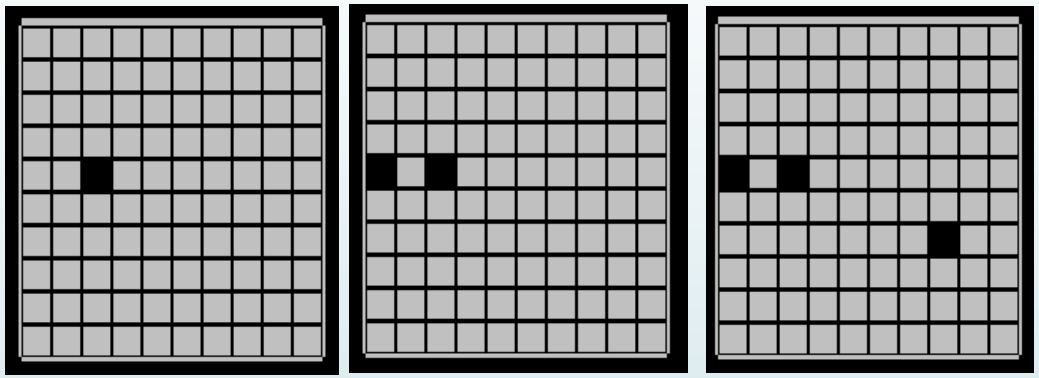
When the button is pressed

- We have to consider 3 cases.
  - Boom!
  - No explosion, but there are some bombs next the position.
  - No explosion, and no bomb nearby.

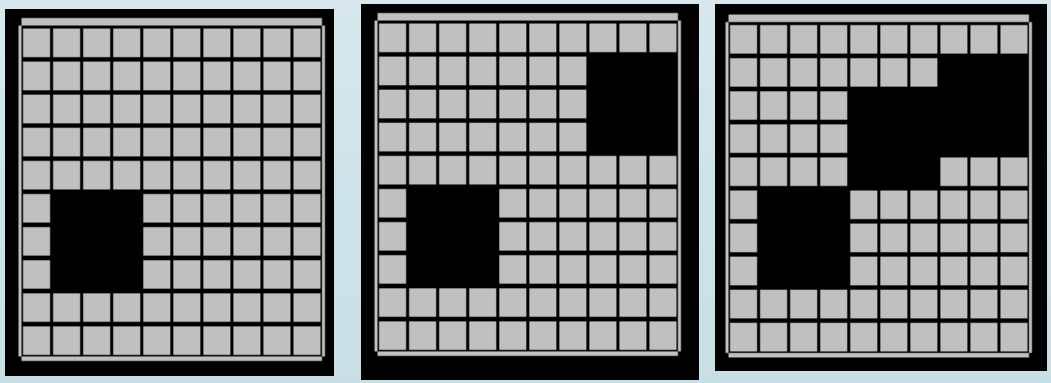


No bomb nearby

- Only press the button

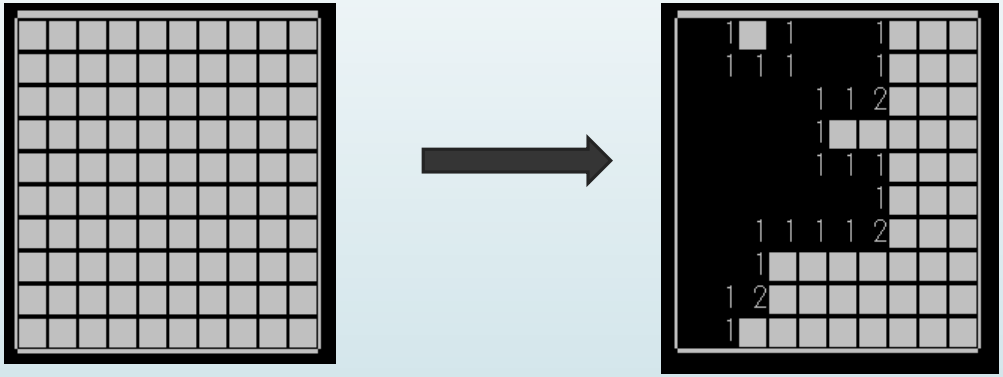


- Press other 8 buttons



No bomb nearby

- We need some information to continue the game, or it just guessing.



How can we do this just press the button once?

Press() function

```
void Press(){    //step
    input_error = 0;
    cout << "input x:  ";
    cin >> x;
    cout << "input y:  ";
    cin >> y;
    system("cls");
    if (0 < x && x <= boardlength && 0 < y && y <= boardlength)
        map_press[x][y] = 1;//step (x, y)
    else{
        input_error = 1;
        cout << "input error, please try again!" << "\n";
        system("pause");
        system("cls");
    }
    if (input_error == 0 && mine_info[x][y] == 0) NoMineAround(x, y);
    //if there is no mine on (x, y)，call NoMineAround()
}
```

No bomb nearby

- We have to use “Recursion”.

```
void NoMineAround(int x, int y){

    if (!map_press[x - 1][y - 1] && mine_info[x - 1][y - 1] != -2){//if (x - 1, y - 1) unstepped and is not a wall
        map_press[x - 1][y - 1] = 1;                //step (x - 1, y - 1)
        if (mine_info[x - 1][y - 1] == 0){// no mine around (x - 1, y - 1)
            NoMineAround(x - 1, y - 1);
        }
    }
    if (!map_press[x - 1][y] && mine_info[x - 1][y] != -2){
        map_press[x - 1][y] = 1;
        if (mine_info[x - 1][y] == 0){
            NoMineAround(x - 1, y);
        }
    }
    if (!map_press[x - 1][y + 1] && mine_info[x - 1][y + 1] != -2){
        map_press[x - 1][y + 1] = 1;
        if (mine_info[x - 1][y + 1] == 0){
            NoMineAround(x - 1, y + 1);
        }
    }
```

No bomb nearby

- We have to use “Recursion”.

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1		★	1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1	★		1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1		★	1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1			1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1		★	1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

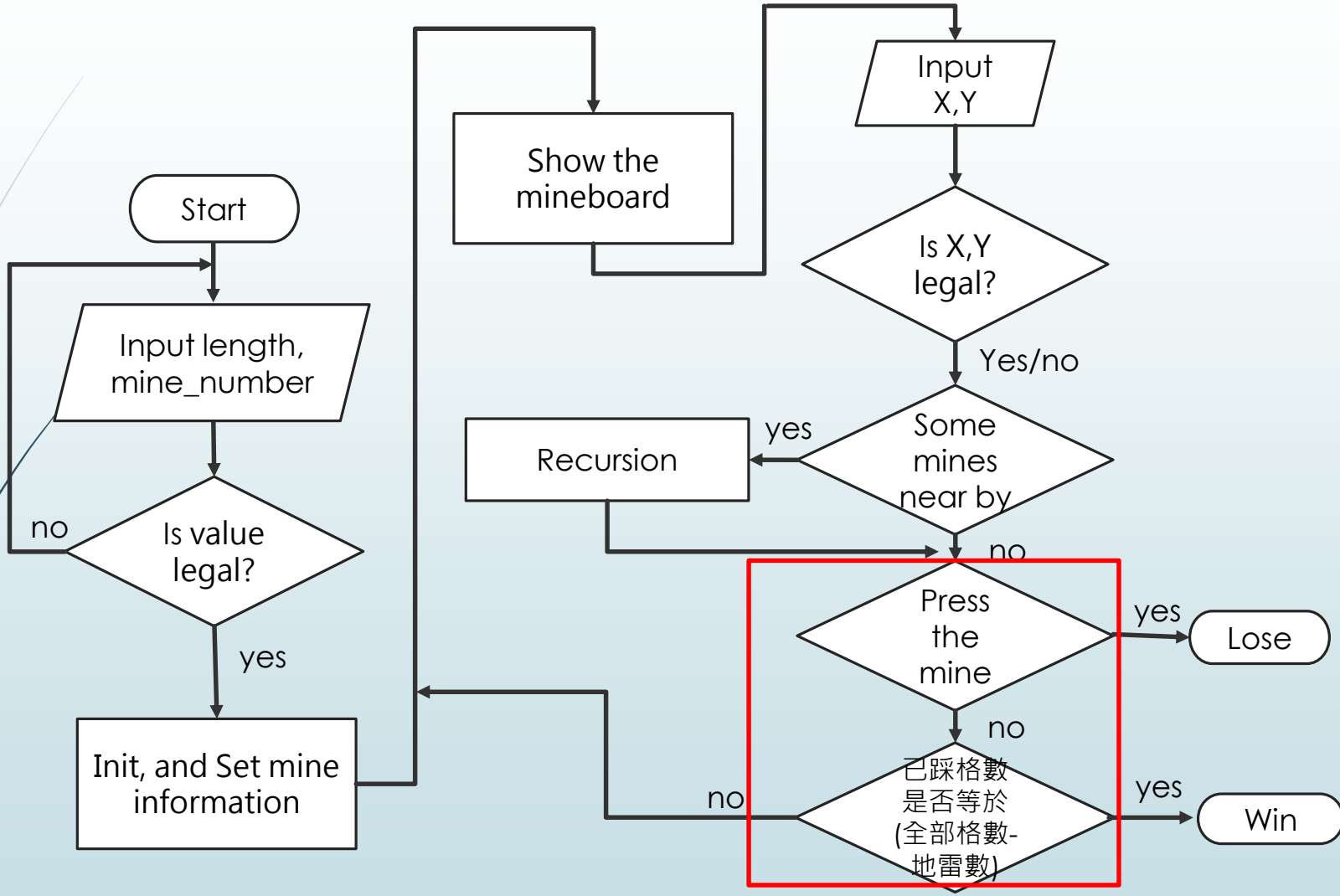
1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1	★		1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1		★	1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

1	2	1	1		
雷	2	雷	1		
1	2	1	1	1	1
1	1		★	1	雷
雷	2	1	2	3	3
1	2	雷	2	雷	雷

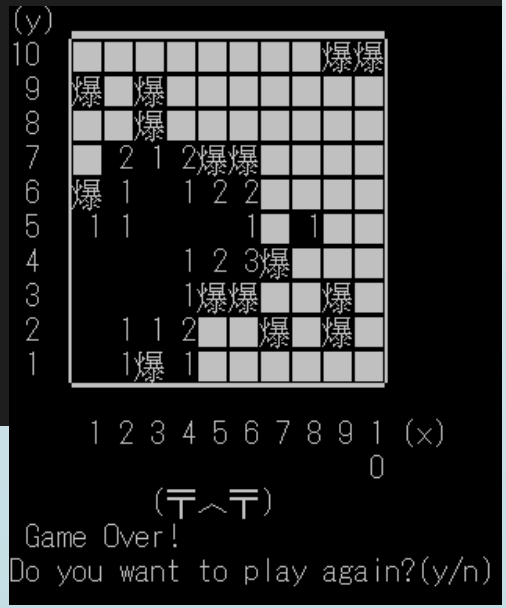
★=(x, y)的位置

Flow chart


Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the mineboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win

Check() function

```
bool Check(){     //check win or lose
    int stepped_number = 0;
    for (int i = 1; i <= boardlength; i++){
        for (int j = 1; j <= boardlength; j++){
            if ((map_press[i][j])){
                stepped_number++;
                if (mine_info[i][j] == -1){ //step on the mine
                    for (int m = 1; m <= boardlength; m++){
                        for (int n = 1; n <= boardlength; n++){ //show all mines on the board
                            if (mine_info[m][n] == -1)
                                map_press[m][n] = 1;
                        }
                    }
                    ShowMineboard(0);
                    cout << " Game Over!\n";
                    return 0;
                }
            }
        }
    }
```

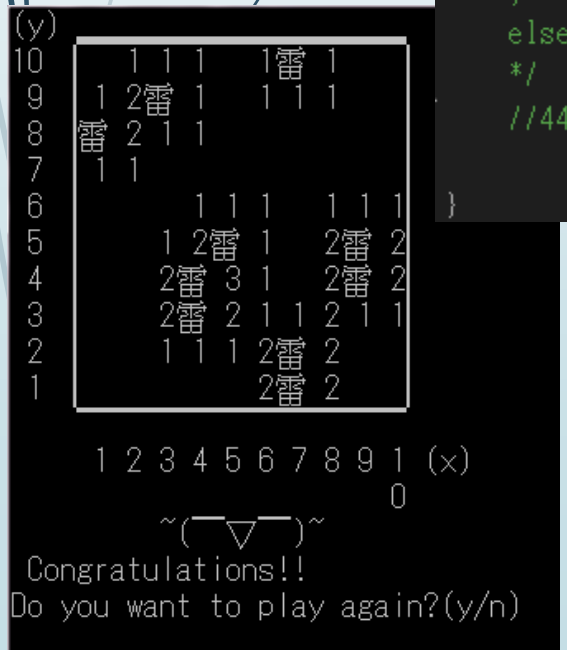


Check() function

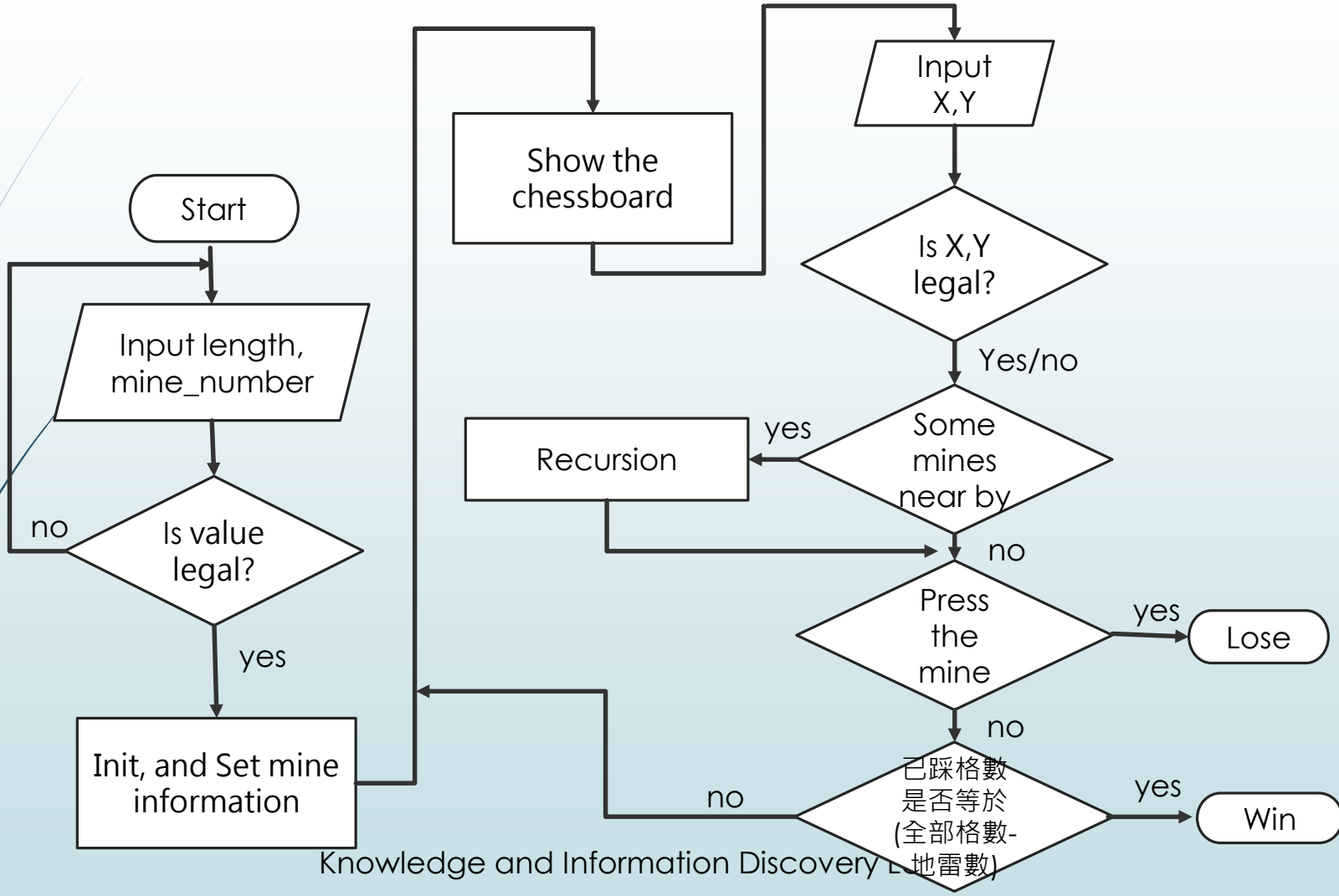
```
    //4444444444444444444444test44444444444444444444444444444
    /*Please compelete the following code.
    You have to define the condition of win.
    Also, you have to show the final result of the board
    Hit: using the relation between stepped_ number and boardlength and mine_number*/
    /*
    if (){       //the condition of win

        cout << " Congratulations!!\n";
        return 0;
    }
    else return 1; //game still continue
    */
    //4444444444444444444444test44444444444444444444444444444

}
```



Flow chart


Start
Input length, mine_number
Is value legal?
no
yes
Init, and Set mine information
Show the chessboard
Input X,Y
Is X,Y legal?
Yes/no
Some mines near by
yes
Recursion
no
Press the mine
yes
Lose
no
已踩格數是否等於(全部格數-地雷數)
no
yes
Win