ARTIFICIAL INTELLIGENCE (18CSC305J) LAB

EXPERIMENT 8

Implementation of Block World Problem

Harsh Goel
RA1811003010185
CSE-C1

Aim:

To implement Block World problem for an application.

Problem Description:

The blocks world has two kinds of components:

1) A table top with three places p, q, and r

2) A variable number of blocks A, B, C, etc., that can be arranged in places on the table or stacked on one another.

A legal move is to transfer a block from one place or block onto another place or block, with these restrictions:

- The moved block must not have another block on top of it.
- No other blocks are moved in the process.

Problem Formulation:

It is straightforward to think of a move in the blocks world as transferring from one place (the source) to another place (the destination). So the name of the block is not necessary to uniquely specify a move.

The three moves used in the example (see to the left) are:

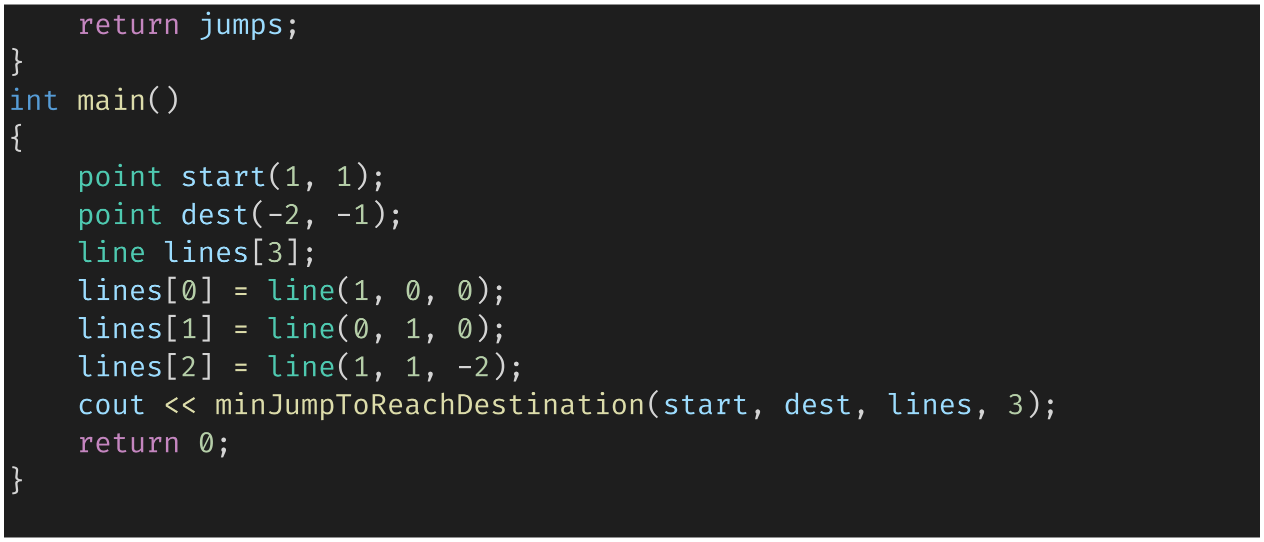
- Move block from p to r.
- Move block from p to q.
- Move block from r to q.

The doMove method in the blocks world move class must return null if there is no block on the source place.

Source Code:

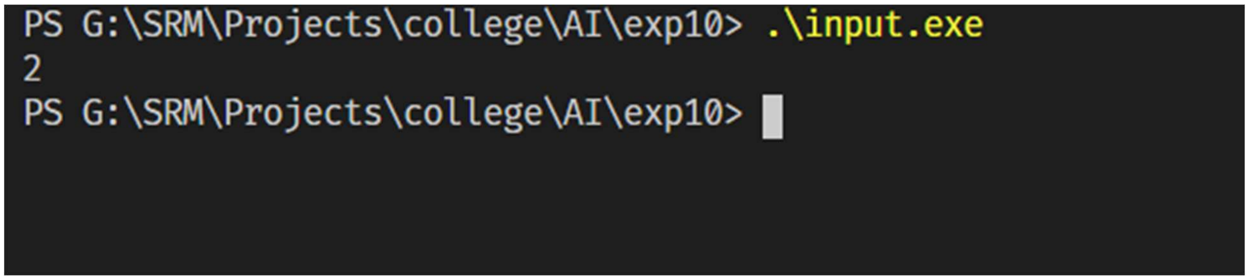
Language- **C++**

```
#include <iostream>
using namespace std;
struct point
{
    int x, y;
    point(int x, int y) : x(x), y(y)
    {
    }
};
struct line
{
    int a, b, c;
    line(int a, int b, int c) : a(a), b(b), c(c)
    {
    }
    line()
    {
    }
};
int evalPointOnLine(point p, line curLine)
{
    int eval = curLine.a * p.x +
               curLine.b * p.y +
               curLine.c;
    if (eval > 0)
        return 1;
    return -1;
}
int minJumpToReachDestination(point start,
                              point dest, line lines[], int N)
{
    int jumps = 0;
    for (int i = 0; i < N; i++)
    {
        int signStart = evalPointOnLine(start, lines[i]);
        int signDest = evalPointOnLine(dest, lines[i]);
        if (signStart * signDest < 0)
            jumps++;
    }
```



```
    return jumps;
}
int main()
{
    point start(1, 1);
    point dest(-2, -1);
    line lines[3];
    lines[0] = line(1, 0, 0);
    lines[1] = line(0, 1, 0);
    lines[2] = line(1, 1, -2);
    cout << minJumpToReachDestination(start, dest, lines, 3);
    return 0;
}
```

Output Verification:



```
PS G:\SRM\Projects\college\AI\exp10> .\input.exe
2
PS G:\SRM\Projects\college\AI\exp10> 
```

Verification:

lines[0] = line(1, 0, 0);

lines[1] = line(0, 1, 0);

lines[2] = line(1, 1, -2);

Source: 1,1

Destination : -2,-1

Minimum Jump : 2

Result: Hence, successfully implemented block world problem and verified the output and document result.