LAB 4

Abhishek Kumar Tiwari
2013CSB1001
Nitin Kumar
2013CSB1020

To compile the code use g++ solution.cpp -o solution
and to run the executable fire the command ./solution filename.txt

Implementation details of BFS and A-star:

The child generator function is used to generate all the states that are feasible i.e., the states to which we can reach using any of the listed actions which are valid.

Goal test checks if the current state is the goal state or not.

Function BFS is the standard Breadth First Search algorithm.

In A-star, the heuristic function h(state a) decides the "badness" of the current state based on the current positions of the block and the positions of those blocks in the goal state.
If a block is not at the right position, penalty of 2^{n+1} is imposed on this state, where n is the number of blocks on top of the current block being inspected.
This heuristic works amazingly well!

BFS on 1.txt takes approximately 50 miliseconds whereas A-star on the same input takes approximately 0.3 miliseconds to reach the goal state on my PC.

Function Trace Path is helper function. Prints the series of actions to take to reach the goal.

Implementation details Goal Stack Planning:

I have created the literals as a string and the conjunction of the literals as the vector of the string.
I have used stack from cpp.

The Function precondition creates a vector of strings containing all the preconditions of the given action.

CheckLiteral Function checks condition of the literal.
PrefAction chooses the relevant actions based on the literal which is false.

The Problem with this approach was that some moves tended to change its own effect increasing the numbers of actions.

The Output is printed based on the format given.