

**CMPT 418 Assignment 1 (due by October 22, 6 pm)**  
 (to be accomplished in teams of two people)

**Agent Planning**

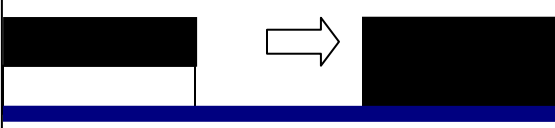
Imagine a coloured block-world consisting of black and white blocks that can be moved by a robotic arm. The blocks can be stacked, unstacked, picked-up and put down, just like in the classical Blocks World. In addition, they can be assimilated, an operation which changes the colour of the lower block to the colour of the upper block when two blocks are stacked upon each other (the upper block “assimilates” the lower block). To simplify the task, assume that assimilation happens only if the upper block is black.

Your task is using the STRIPS planning algorithm (in the notes) to write an automatic player in the “Assimilation game” played in the Blocks World. Your player should, given a description of any initial state of the world and a goal state (expressed as a set of predicates), generate a plan to achieve the goal state starting from the initial state. The plan will contain a list of actions. The goal states can be such that **there are no white blocks** and the **blocks are in particular configuration**.

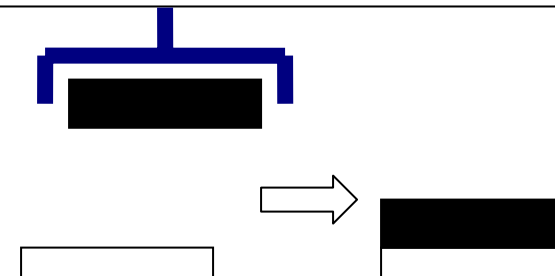
The list of predicates describing states in the Blocks World is: (as in textbook, pg. 73):

Predicate	Meaning
On (x, y)	Object x on top of object y
OnTable (x)	Object x is on the table
Clear (x)	Nothing is on top of object x
Holding (x)	Robot Arm is holding x
ArmEmpty	Robot Arm is empty
Black (x)	Object x is black
White (x)	Object x is white

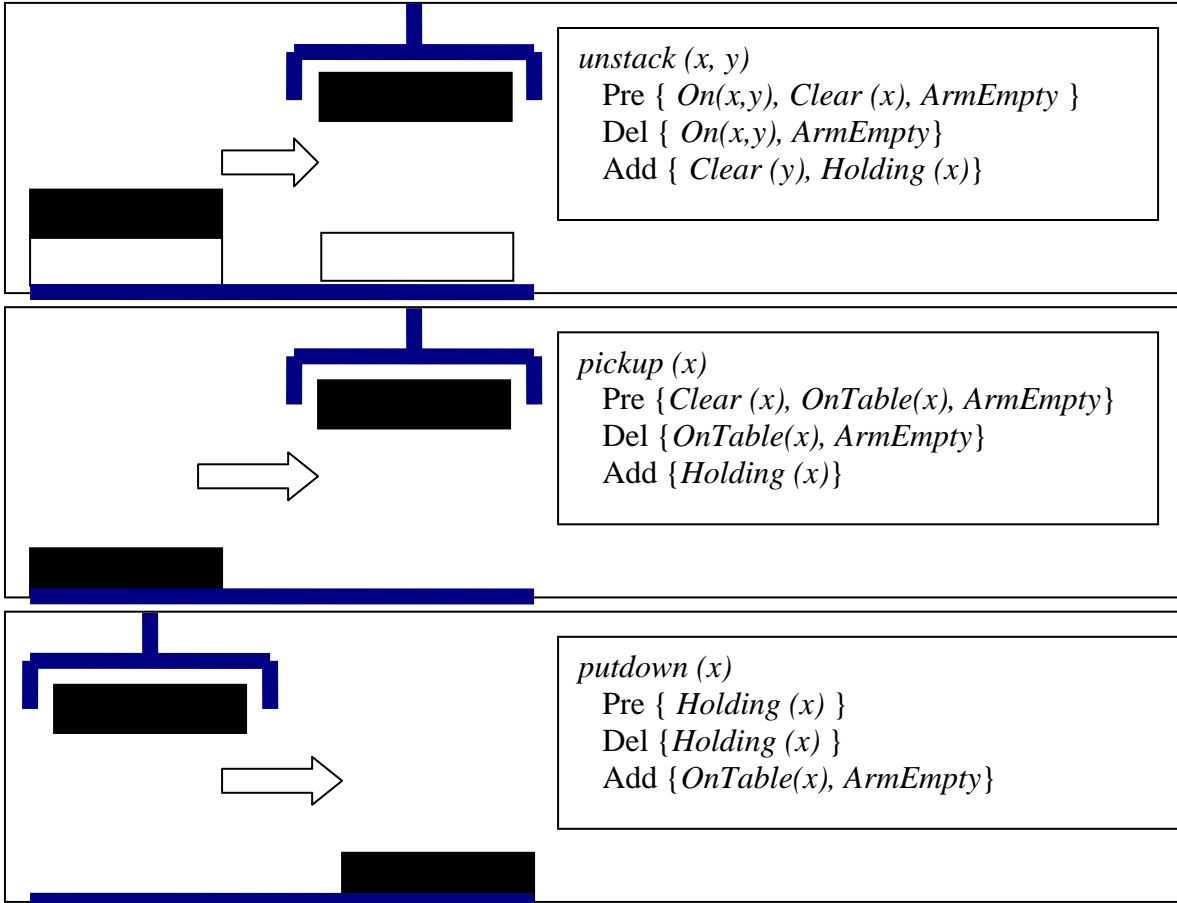
The list of possible actions is:



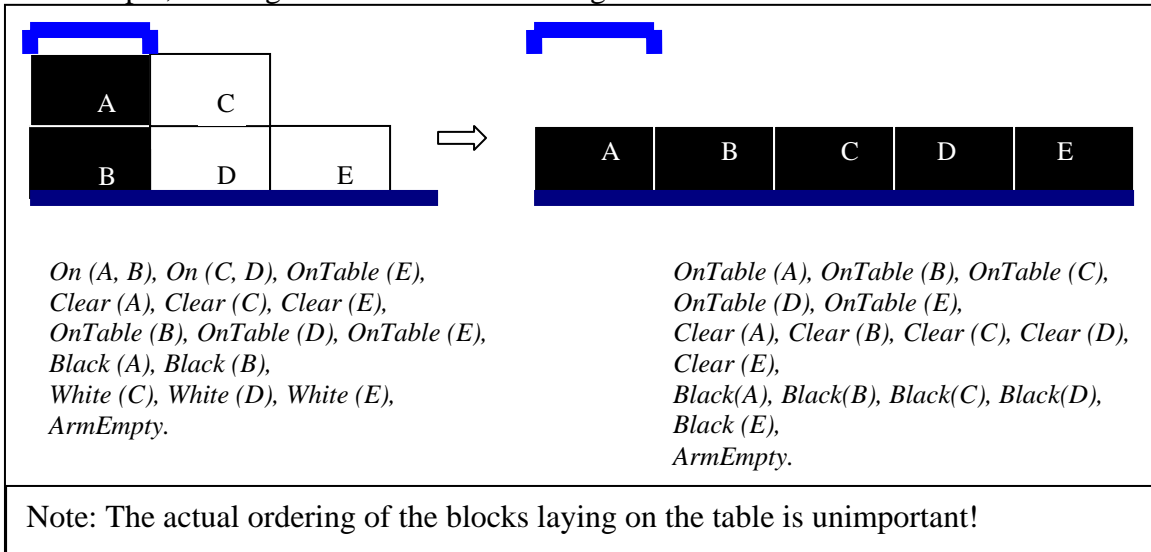
*assimilate (x, y)*  
 Pre {Black(x), White(y), On (x,y)}  
 Del {White (y)}  
 Add {Black (y)}



*stack (x, y)*  
 Pre {Clear (y), Holding (x)}  
 Del {Clear(y), Holding (x)}  
 Add {ArmEmpty, On(x,y)}



For example, to bring the initial state into the goal state shown below:



One possible plan will be: *unstack(C,D), putdown(C), unstack(A,B), stack(A,E), assimilate (A,E), pickup(B), stack(B,D), assimilate(B,D), unstack (B,D), stack(B,C), assimilate (B,C), unstack(A,E), putdown(A), unstack(B,C), putdown(B).*

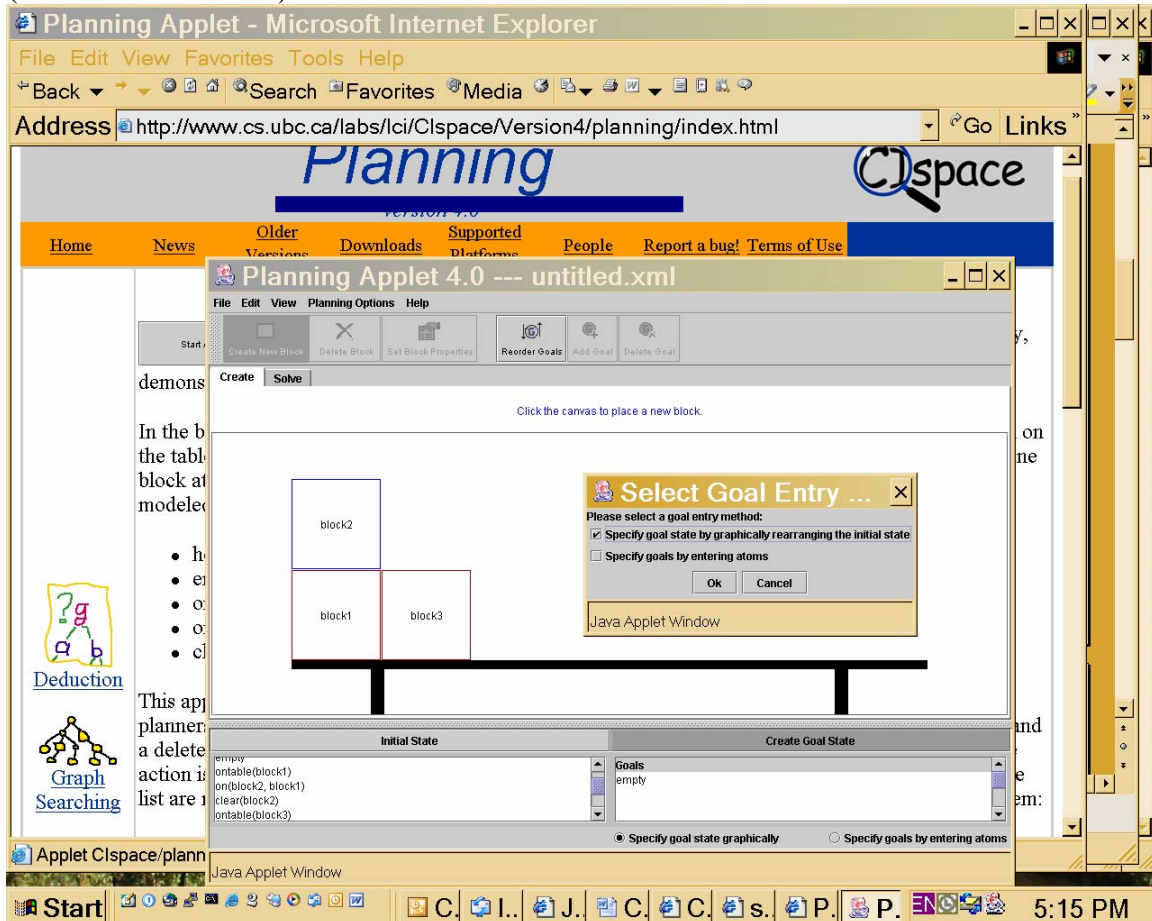
## The Program:

You can use Java, C++ or C# to program the algorithm.

The program should have a graphical user interface (GUI) to allow for easy testing. The interface should give three options for the user:

- describe the initial state and the goal state either by using text description (with predicates) or by using graphical representation (by creating blocks of certain colours in certain positions on the screen);
- allow the user to start the planning algorithm which will output the plan in text format; display the plan.
- execute the plan, by displaying the effects of the actions step by step in the graphical representation starting with the initial state until the goal state is reached (the ordering of blocks on the table doesn't matter).

Here is an example of the interface of a similar, planning applet developed at the UBC (without assimilation):



## Marking

Total: 20 marks

For full marks your program has to work reliably with more than 5 blocks (e.g. with 6), have a GUI and do colour assimilation. Below is a table that shows how many marks you will get, if some of the requirements are not met.

<b>Planning works *always* with ? blocks</b>	<b>Points</b>			
	<b>GUI &amp; Assimilation</b>	<b>No GUI, has Assimilation</b>	<b>GUI, but No Assimilation</b>	<b>No GUI &amp; No Assimilation</b>
> 5	19 or 20	17 or 18	15 or 16	13 or 14
5	17 or 18	15 or 16	13 or 14	11 or 12
4	15 or 16	13 or 14	11 or 12	9 or 10
3	13 or 14	11 or 12	9 or 10	8
2	11 or 12	9 or 10	8	6
< 2	10	8	6	4