•Which heuristics did you use for the A* algorithm?

Depending on how many stacks are equal to the stacks of the stacks add one point to the heuristic. Example: current (A); (B); ( ); (C). goal ( ); (A); (B); (C). Since 1 stack in the current state is equal to the goal state, then it will have an heuristic of one.

•Test your program with a couple of different problems. Increase the size of the problem to test the limits of your program. Make a table comparing **how many nodes are searched** to find the answer for each problem. For this table, you should compare a number of different problems (at least 3) to avoid a statistical bias. Which of the three algorithms (UCS, A with consistent and and A with an inconsistent heuristic) searches the least nodes and which one take the most?

Maxheight - Start state – Goal state	UCS	A* heuristic	A* bad heuristic
3 **(A); (B); (C); ()** **(); (A); (B); (C)**	82 nodes searched	27 nodes searched	53 nodes searched
2 **(A); (B); (C)** **(A, C); X; X**	7 nodes searched	5 nodes searched	5 nodes searched
2 **(A); (B); ()** **(A, B); X; X**	3 nodes searched	2 nodes searched	3 nodes searched
1 **(A); (B); ()** **(A, B); (); ()**	13 nodes searched	13 nodes searched	13 nodes searched

•Why does this happen?

Since A* uses an heuristic to “assume” it is going the right way, the it gets the solution in a shorted amount of states. Even using a bad heuristic A* has proven to be more optimal than UCS. Keep in mind that this all depends in the heuristic, I could have chosen a better heuristic the make the algorithm even faster, or use one even worse so it performs worse than UCS

•Which algorithms are optimal? Why?

Based on giving the shortest cost solution they are all optimal for this problem since the three of them gave me the same result. Based on time, A* is more optimal than UCS and the better the heuristic the better the performance of A*. When there is no solution they all give the same answer because of how I developed the algorithms, they will give all the possible tries until they find the solution or are out of options

- In your opinion, what are the benefits of simpler algorithms versus more complex ones?

Well, complex algorithm take more time do actually do, but they give the best reward so far, so we actually need them to solve real life difficult problems. Simple algorithms on the other hand seem to give a better result than the complex one when the problem is as well simple or short. Also simple algorithms are good for introductions.