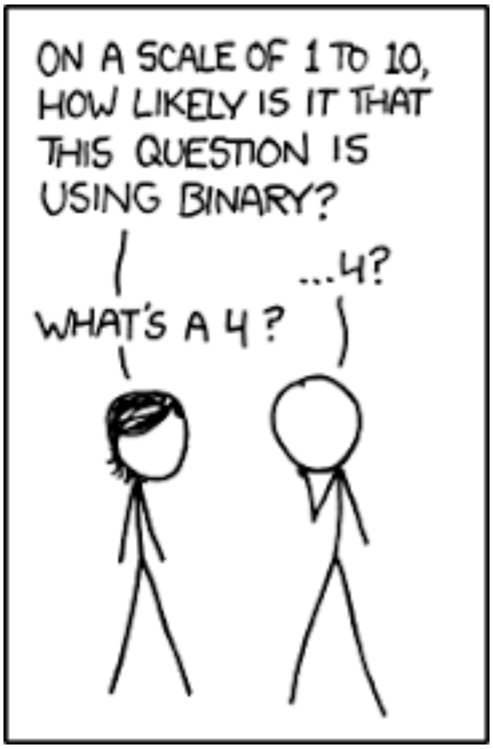
# CITS3001 Mid-semester Test 2017

## Semester 2

## Fifty minutes

## Answer all four questions

## Total marks 60




**Name:** ANSWERS

**Student Number:** ____________________

**When showing the operation of an algorithm, include enough detail to make it clear that you understand how *the algorithm* solves the problem.**

**Answer all questions in the spaces provided in the booklets.**

**Question 1: Strings** **(15 marks)**

Old arcade games had limited inputs: a joystick (*Up, Down, Left, Right*) and two buttons (*B1* and *B2*). Some games also had cheat codes, which we will assume are a sequence of 20 inputs that can unlock an advantage for a player. For example, it might be the case that entering *UUDD1UUDD2UUDD1UUDD2* gives the player an extra life.

You have access to a large string (more than 100000 characters) describing the inputs of an experienced player who you assume knows and frequently uses the cheat codes. You would like to use this string to identify candidate sequences of inputs that could be cheat codes (the sequences of 20 inputs that occur most frequently).

a) Describe an algorithm that achieves this task.

7 marks

b) Explain why your algorithm is correct.

4 marks

c) Give the worst case and expected case complexity of your algorithm.

4 marks

This is not relevant to 2018 as it is a String question.

a) Solution: A variant of Rabin Karp String matching algorithm. Hash every 20 characters to an integer, and then check for the most commonly occuring integers. Each one then has to be checked for spurious matches.

b) Identical Strings will have identical hashes so a well chosen hash function will identify re occuring strings.

c) This will have an expected case of linear time. In the worst case, there will be many spurious matches which would give $O(|p|.|w|)$.

## Question 2: Optimisation (15 marks)

a) Define the term *Greedy Algorithm* and describe one problem where greedy algorithms always give a correct answer, and one problem where greedy algorithms sometimes give the wrong answer.

7 marks

b) Demonstrate the dynamic programming algorithm for the 0-1 knapsack problem, with the items with weight and value given in the following table.

Error: The knapsack has size 12.

| Item | A | B | C | D |
|---|---|---|---|---|
| **Weight** | 5 | 7 | 9 | 11 |
| **Value** | 3 | 4 | 5 | 6 |

8 marks

a) A greedy algorithm is one which makes locally optimum decisions, and never backtracks. This works for minimum spanning tree, but not for Travelling salesmen.

b)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| B | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 7 |
| C | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 4 | 4 | 5 | 5 | 5 | 7 |
| D | 0 | 0 | 0 | 0 | 0 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 7 |

Items A and B.

**Question 3: *A*** **(15 marks)**

The tower of Hanoi is a puzzle that consists of three rods and a number of disks of different sizes, which can slide onto any rod. The puzzle starts with the disks in a neat stack in ascending order of size on leftmost rod, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to the rightmost rod, obeying the following simple rules:

1. Only one disk can be moved at a time.
2. Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack.
3. No disk may be placed on top of a smaller disk.

Consider the problem of using the A* algorithm to find the fastest solution to this problem.

a) Explain the terms *admissible*, *monotonic* and *dominates* in the context of heuristics for A* algorithm?

6 marks

b) Consider the heuristic that estimates the number of moves remaining to be the total number of discs minus the number of discs on the rightmost rod. Describe a new heuristic and show that it is admissible, monotonic and dominates the old heuristic.

9 marks

a) i) A heuristic $h$ is admissible if $h(x) \leq h^*(x)$ for all $x$, where $h^*$ is the true cost of reaching the goal from $x$.

ii) A heuristic $h$ is monotonic if $h(x) \leq c(x, a, y) + h(y)$.

iii) Heuristic $h_1$ dominates $h_2$ if for all $x$ $h_1(x) \geq h_2(x)$.

b) see next page.

b) A new heuristic could be

$h(s)$ is $\max\{n \mid disk_n$ is not on the rightmost rod$\}$

(assuming are ordered from smallest to largest)

i) admissible: If disk $n$ is not on the rod, then it needs to be put on the rod __before__ the $n-1$ smaller disks, so at least $n$ moves.

ii) monotonic: Clearly $h(s)$ can only decrease by 1 at most, and the step costs are 1 s.t. $h(s) \leq h(s') + 1$.

iii) dominates: If there are $n$ disks not on the rightmost rod, the largest disk must have order $\geq n$.

## Question 4: Game-Playing **(15 marks)**

Two children are sharing a pile of lollies, and they come up with the following method:

The first child (the divider) divides the lollies into two non-empty piles. The second child (the chooser) can either take the smallest pile, leaving the largest pile for the divider, or they can give the smallest pile to the divider, and then they repeat the process with the largest pile, except this time they swap roles. They repeat this process until all the lollies are given out.

For example, with a pile of 10 lollies, the first child could break the piles in 2 and 8 lollies. The second child is not happy with 2, so he gives those to the first child, and then divides the pile of 8 lollies into 2 and 6. The first child is not happy with 4 lollies, so he gives the pile of 2 lollies to the second child, and spilts the pile of 6 into a pile of 3 and 3. The second child accepts the split and takes the 3 lollies, leaving 3 more for the first child and they both end up with 5.

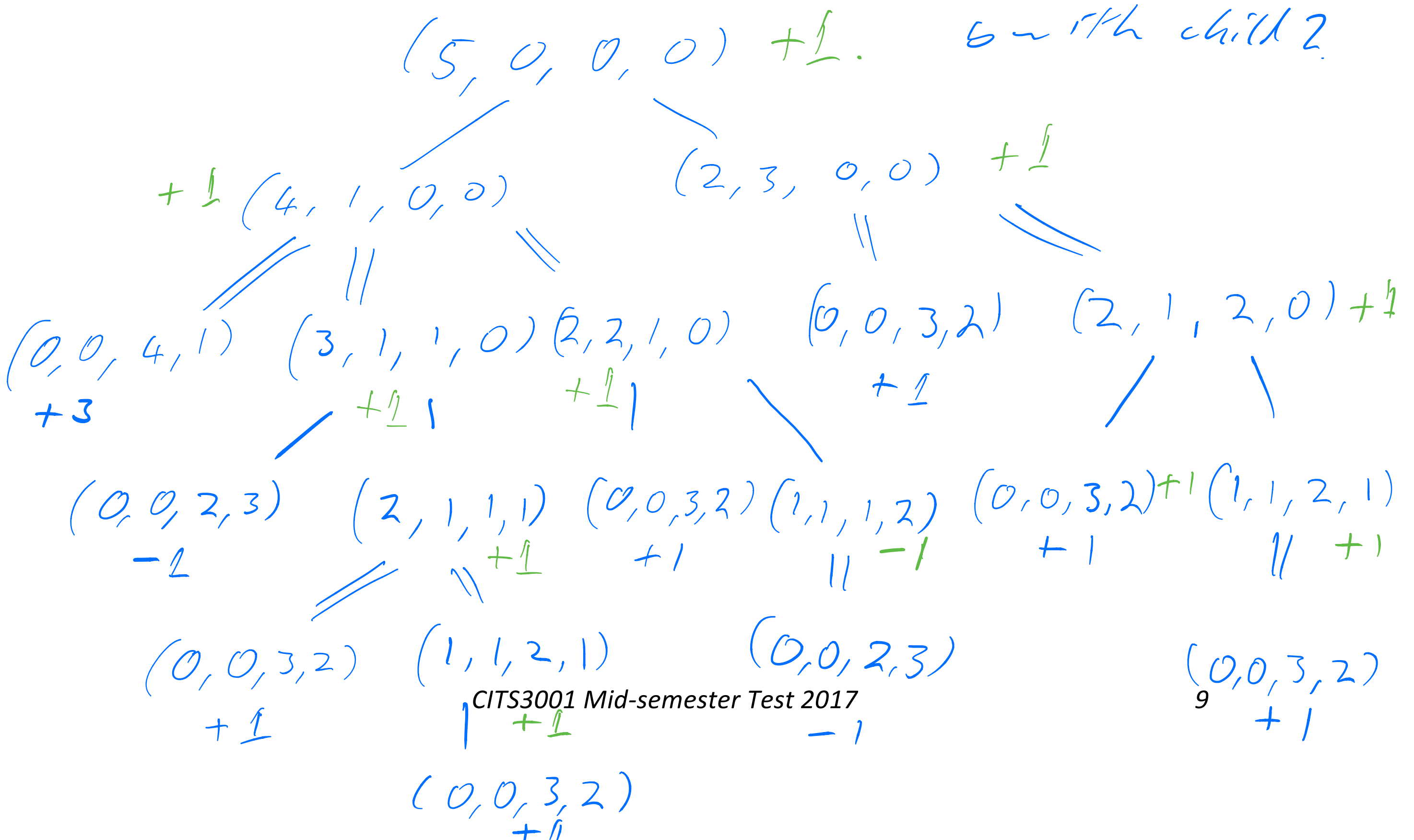
a) Demonstrate the minimax algorithm for this game assuming that the pile starts with 5 lollies.

10 marks

b) Suppose that we started with a pile of 100 lollies. What would you use as an evaluation function in order to able to use alpha-beta pruning? Explain your choice.

5 marks

a use (l, r, a, b) for
l in the left pile
r in the right pile
a with child 1
b with child 2



b) Evaluation of $(a, b, c, d)$ would be $\left(c + \frac{a+b}{2}\right)$ (what max already has, plus the half of all unclaimed lollies.). This assumes the process is roughly fair.