TIC1001—Introduction to Computing and Programming I
National University of Singapore
School of Continuing and Lifelong Education

# Practical Examination 2

17 November 2018

**Time allowed:** 2 hours

**Instructions (please read carefully):**

1. This is **an open-book exam**. You are allowed to bring in any course or reference materials in printed form. No electronic media or storage devices are allowed.
2. This practical exam consists of **two** questions. The time allowed for solving this test is **2 hours**.
3. The maximum score of this test is **20 marks**. Note that the number of marks awarded for each question **IS NOT** correlated with the difficulty of the question.
4. While you are also provided with the templates `q1-template.cpp` and `q2-template.cpp` to work with, your answers should be submitted on Coursemology. Note that you can **only run the test cases on Coursemology for a limited number of tries** because they are only for checking that your code is submitted correctly. You are expected to test your own code for correctness locally on VS Code and not depend only on the provided test cases. Do ensure that you pasted your answers correctly by running the test cases at least once.
5. In case there are problems with Coursemology.org and we are not able to upload the answers to Coursemology.org, you will be required to name your file `q1-<student no>.cpp` and `q2-<student no>.cpp` where `<student no>` is your student number and leave the file in the `tic1001-pe2` folder on the Desktop. If your file is not named correctly, we will choose any .cpp file found at random.
6. Please note that it shall be your responsibility to ensure that your solution is submitted correctly to Coursemology and correctly left in the desktop folder at the end of the examination. Failure to do so will render you liable to receiving a grade of **ZERO** for the Practical Exam, or the parts that are not uploaded correctly.
7. Please note that while sample executions are given, it is **not sufficient to write programs that simply satisfy the given examples**. Your programs will be tested on other inputs and they should exhibit the required behaviours as specified by the problems to get full credit. There is no need for you to submit test cases.
8. **Marks may be penalized** for poor code formatting and design choice. Indentation of code should be consistent and variable names aptly chosen.

# ALL THE BEST!

## Question 1: Up up and away [10 marks]

SpaceY has developed a rocket engine that is capable of increasing the speed of the rocket by 1 unit per unit time. The speed of the rocket directly increases its distance. In other words, suppose at time $t$ the rocket speed is $s$ and it is $D$ distance from the ground. If a single rocket engine is on at time $t$, then at time $t+1$ the rocket speed will be $s+1$ and its distance will be $D+s+1$.

Now it is possible to use multiple engines on a rocket to give a higher acceleration. Suppose at time $t$ the rocket speed is $s$ and $n$ engines are on, then at time $t+1$ the rocket speed will be $s+n$.

All engines have a limited capacity and once turned on, it cannot be turned off until it has used up all its fuel, after which it can not be turned on again. Suppose an engine has a capacity of $c$ time units, i.e. it has enough fuel to be on for $c$ time units. In other words, if the engine is turned on at time $t$, then it will use up its fuel and turn off at time $t+c$.

The problem with using multiple engines is that too much power may be delivered if all engines are turned on at the same time. So SpaceY delays the ignition of each engine by $d$ time units. If an engine is turned on at time $t$, then the next engine will only be turned on at time $t+d$. You may assume $d > 0$.

The following illustration shows the launching sequence of a rocket with 3 engines, each of capacity 5, and ignition delay of 2. "$*$" indicates that the engine is on in that time slot.

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | | | | | |
| Engine 2 | | | * | * | * | * | * | | | |
| Engine 3 | | | | | * | * | * | * | * | |
| Speed | 0 | 1 | 2 | 4 | 6 | 9 | 11 | 13 | 14 | 15 |
| Distance | 0 | 1 | 3 | 7 | 13 | 22 | 33 | 46 | 60 | 75 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Your task is to implement the function

```
int distance(int num, int capacity, int delay, int t)
```

which takes as input the number of engines, the capacity of each engine, the ignition delay and a time $t$. Assume that the first engine is turned on at time 0, when the rocket speed is 0 and it is 0 distance from the ground. The function should output the distance of the rocket from the ground at time $t$.

The following shows the grading criteria if the function outputs the correct value with the given assumptions:

- 1 engine, with unlimited capacity (i.e. $n = 1$ and $c > t$). [2 marks]
- 1 engine, with limited capacity (i.e. $n = 1$). [2 marks]
- $n$ engines, with unlimited capacity, with a delay of 1 (i.e. $c > t$ and $d = 1$). [2 marks]
- $n$ engines, with limited capacity, with delay of 1 (i.e. $d = 1$). [2 marks]
- $n$ engines, with limited capacity and arbitrary delay $d$. [2 marks]

The function has several components and can become quite complex if not approached properly. We advise you start solving with the strictest assumption above and slowly progress to the bottom.

Sample cases:

| Function call | Output | Explanation |
|---|---|---|

`distance(1, 20, 1, 10)` — 55

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | * | * | * | * | * | * |
| Speed | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Distance | 0 | 1 | 3 | 6 | 10 | 15 | 21 | 28 | 36 | 45 | 55 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

`distance(1, 5, 1, 10)` — 40

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | | | | | | |
| Speed | 0 | 1 | 2 | 3 | 4 | 5 | 5 | 5 | 5 | 5 | 5 |
| Distance | 0 | 1 | 3 | 6 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |

`distance(3, 20, 1, 10)` — 64

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | * | * | * |
| Engine 2 | | * | * | * | * | * | * | * |
| Engine 3 | | | * | * | * | * | * | * |
| Speed | 0 | 1 | 3 | 6 | 9 | 12 | 15 | 18 |
| Distance | 0 | 1 | 4 | 10 | 19 | 31 | 46 | 64 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

`distance(3, 5, 1, 7)` — 60

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | | | |
| Engine 2 | | * | * | * | * | * | | |
| Engine 3 | | | * | * | * | * | * | |
| Speed | 0 | 1 | 3 | 6 | 9 | 12 | 14 | 15 |
| Distance | 0 | 1 | 4 | 10 | 19 | 31 | 45 | 60 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

`distance(3, 5, 2, 7)` — 46

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Engine 1 | * | * | * | * | * | | | |
| Engine 2 | | | * | * | * | * | * | |
| Engine 3 | | | | | * | * | * | * |
| Speed | 0 | 1 | 2 | 4 | 6 | 9 | 11 | 13 |
| Distance | 0 | 1 | 3 | 7 | 13 | 22 | 33 | 46 |
| Time | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

## Question 2: Hidden in plain sight [10 marks]

Ms. Orycle "invented" a new way to write secret message. The idea is very simple, as you scan the message from left to right, you will take note of any **repeating** characters and count the number of repeats, *R*. You will then **shift** the next non-repeating character *R* steps **forward** in the ASCII order.

The following are some examples to illustrate the scheme:

| Original | Output | Comment |
|---|---|---|
| `***A` | `D` | "`*`" repeated **3** times, so "`A`" was moved 3 steps. |
| `abc` | `abc` | No repeating characters, so everthing stays unchanged. |
| `aaA!!!!Bbc` | `CFbc` | "`A`" $\rightarrow$ 2 steps, "`B`" $\rightarrow$ 4 steps, "`bc`" unchanged. |
| `**AAAB` | `CD` | The two "`*`" move the next "`A`", then the remaining two "`A`" moved `B` two steps. |

Your task is to implement the function `string reveal(string original)` which takes a string as an input and return the processed result according to the scheme described.

Several assumptions to take note:

1. If there are repeating characters, then there must be at least one different character afterwards for the shifting to take place. So, strings like "`AAAAA`" is not valid and will not be tested.
2. The amount of shift is **guaranteed** to stay in the ASCII range, there is no need to perform wrap-around.
3. The string can contain any valid ASCII code, not limited to the printable ones, i.e. there is no need to check for alphanumeric characters etc, just process accordingly.

[10 marks]

**— E N D  O F  P A P E R —**