Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Aids Allowed: **None**

First (Given) Name(s):

Last (Family) Name(s):

10-Digit Student Number:

UTORid (e.g., `pitfra12`):

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, write your name, student number, and UTORid above
(please do this now!) and read the instructions below *carefully*.

- This term test consists of 7 questions on 12 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete.*
- Answer each question directly on the test paper, in the space provided, and use a "blank" page for rough work. If you need more space for one of your solutions, use one of the "blank" pages and *indicate clearly the part of your work that should be marked.*
- Comments are not required except where indicated, although they may help us mark your answers.
- No error checking is required: assume all user input and all argument values are valid.
- Do not remove any pages from the exam booklet.

Good Luck!

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Question 1. [10 marks]

This question tests your understanding of expression evaluation and variable use in Python.

Beside each expression in the table below, write the value produce when it is evaluated. If the code would cause an error, write ERROR and give a brief explanation.

Some of the expressions involve variables.

Assume this code is run before each expression is evaluated:

```
t = "tTrRuUeE"
wd = "Best Midterm Ever"
```

Expression	**Expression Value/ERROR**
`10 / 5 - 5`	
`t[1] + t[2] + t[4] + t[6] == True`	
`3 * "p"`	
`False or "october2" < "october18"`	
`True or wd[40]`	

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Question 2. [10 marks]

This question tests your understanding of the Function Design Recipe and parameter use.

Step 1 of the Function Design Recipe, creating examples, has been completed for a function `enclose_string`. (*Enclose* means to surround or close off on all sides.)

Complete steps 2, 3, and 4 of the Function Design Recipe:

- Step 2: fill in the function header (including the type contract) in the first box,
- Step 3: write a good description in the second box, and
- Step 4: write the body of the function in the third box.

```
def enclose_string(
    """
    """
    >>> enclose_string('course', '^')
    '^course^'
    >>> enclose_string('expensive','$$$')
    '$$$expensive$$$'
    """
```

Question 3. [10 MARKS]

This question tests your understanding of Assignment 1 string rotation.

Read the function header, description, and examples carefully. Make sure you understand what the function is supposed to do, then write the body of the function.

It may help to come up with a few more examples to see if you can notice a pattern.

You must not use loops.

```
def rotate(word: str, num_steps: int) -> str:
    """Return word right-rotated num_steps times.

    Precondition: num_steps < len(word)

    >>> rotate('CAT', 1)
    'TCA'
    >>> rotate('GOOSE', 2)
    'SEGOO'
    >>> rotate('CAT', 3)
    'CAT'
    """
```

Question 4. [10 MARKS]

This question tests your understanding of for loops, while loops, and the accumulator pattern.

Consider this function:

```
def gather_between(s: str, c1: str, c2: str) -> str:
    """Return a string containing all the characters in s that are greater than
    or equal to c1 and less than or equal to c2.

    Precondition: len(c1) == 1 and len(c2) == 1

    >>> gather_between('bet my cows go', 'k', 'o')
    'moo'
    >>> gather_between('Ping All CS Knowledge', 'A', 'L')
    'ACK'
    >>> gather_between('Nothing to see here', 'u', 'z')
    ''
    """
```

On the opposite page, you will write two versions, one that uses a for loop and one that uses a while loop.

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Part (a) [5 MARKS] Complete the function **using a for loop**.

```
def gather_between(s: str, c1: str, c2: str) -> str:
    """See the previous page for the docstring and examples."""
```

Part (b) [5 MARKS] Complete the function **using a while loop**.

```
def gather_between(s: str, c1: str, c2: str) -> str:
    """See the previous page for the docstring and examples."""
```

Question 5. [10 marks]

This question tests your understanding of string slicing with a negative step.

The following function is missing a description, and the two example calls are incomplete.

Write a description in the first box.

In the second and third boxes, add arguments that will result in the return values shown. For the example calls, there are many correct answers, and providing any one of them will earn full marks.

```
def midterm_function(string1: str, string2: str) -> bool:
    """

    >>> midterm_function(                                    )
    True
    >>> midterm_function(                                    )
    False

    """

    return string1 == string2[::-1]
```

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Question 6. [2 marks]

This question tests your understanding of how if statements work.

For the function below, write the value that `x` refers to in the boxes indicated. In each box, write the value as it would appear at a breakpoint, **before** the line is executed, just like the debugger does.

If a breakpoint is not reached, write `N/A` in the box. (N/A means "not applicable".)

```
1  def midterm_function_a():
2      """Used to test understanding
3      of how if statements are executed.
4
5      This docstring doesn't describe
6      the return value. Bad style!
7      :-)
8      """
9      x = 'a'
10
11     if x == 'a':
12         return x
13     elif x == 'f':
14         x = 'k'
15
16     x = 'm'
```

Breakpoint at line 9: `x` ☐

Breakpoint at line 12: `x` ☐

Breakpoint at line 14: `x` ☐

Breakpoint at line 16: `x` ☐

Question 7. [10 MARKS]

This question tests your understanding of if statements and constants.

The TTC (Toronto Transit Commission) is having a tough year in terms of quality of service given the constant incidents (e.g., train derailments, electrical shutdown, emergency alarms) in the subway lines. The TTC protocol establishes that once an incident in the subway occurs, shuttle buses must be sent in order to cover the affected route.

There are three subway lines that are particularly problematic: the Yonge-University Line (YELLOW) the Bloor-Danforth Line (GREEN), and the Scarborough Line (BLUE). Each line has its own set of buses:

Subway Line	**Number of Buses**
YELLOW	40
GREEN	50
BLUE	20

Each bus can carry 40 passengers.

The TTC has hired you to program a function that, given the number of passengers affected and the line where the incident occurs, returns the minimum number of buses needed to fully cover the demand produced by an incident. It should return `-1` if there are not enough buses to help all the passengers.

On the opposite page, we have completed the first three steps of the Function Design Recipe. Complete the function body. You must use the constants in your solution.

Hint: you can use this formula to calculate the required number of buses:

```
(num_passengers - 1) // BUS_CAPACITY + 1
```

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

```
YELLOW_LINE = 'YELLOW'
GREEN_LINE = 'GREEN'
BLUE_LINE = 'BLUE'

NUM_YELLOW_BUSES = 40
NUM_GREEN_BUSES = 50
NUM_BLUE_BUSES = 20

BUS_CAPACITY = 40

def number_of_buses(num_passengers: int, line_color: str) -> int:
    """Return the minimum number of buses needed to transport num_passengers
    passengers on the subway line designated by line_color.

    Precondition: num_passengers > 0 and line_color is one of
    YELLOW_LINE, GREEN_LINE, and BLUE_LINE.

    >>> number_of_buses(1, 'GREEN')
    1
    >>> number_of_buses(40, 'YELLOW')
    1
    >>> number_of_buses(41, 'YELLOW')
    2
    >>> number_of_buses(2001, 'GREEN')  # Because 50 * 40 is 2000
    -1
    >>> number_of_buses(801, 'BLUE')  # Because 20 * 40 is 800
    -1
    """
```

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Midterm Test A

CSC108H1F / LEC0301/0401

Monday 3 February 2020 — Duration: **75 minutes**

Use the space on this "blank" page for scratch work, or for any solution that did not fit elsewhere.
Clearly label each such solution with the appropriate question and part number.

Total Marks = 62