CSC 108H1 S 2012 Test 2
Duration — 45 minutes
Aids allowed: none

Student Number: 999292509

Last Name: Qiu **First Name:** Rui

Lecture Section: L0101 **Instructor:** Campbell

Do **not** *turn this page until you have received the signal to start.*
(Please fill out the identification section above, **write your name on the back of the test**, and read the instructions below.)
Good Luck!

This midterm consists of 4 questions on 6 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* Comments and docstrings are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code. No error checking is required: assume all user input and all argument values are valid.
If you use any space for rough work, indicate clearly what you want marked.

# 1:	1 / 3
# 2:	1.5 / 3
# 3:	4 / 5
# 4:	7.5 /13
TOTAL:	14 /24

Question 1. [3 MARKS]

For each block of code in this question, write its output in the box below it. If it would generate an error, say so, and give the reason for the error.

Part (a) [2 MARKS]

```
inner = [1, 2, 3]
nested = [inner, inner]
print nested[0]
```

[1, 2, 3] ✓

```
# continued from above
nested[0].append(4)
print nested
```

[[1,2,3,4],[1,2,3,4]]

[[1,2,3],[1,2,3]] ✗

Part (b) [1 MARK]

```
inner2 = [4, 5, 6]
nested2 = [inner2[:], inner2[:]]
nested2[0].insert(3, 7)
print nested2
```

[[4,5,6,7],[4,5,6]]

[[4,5,6],[4,5,6]] ✗

1.5

Question 2. [3 MARKS]

In A2, a board was a list of lists of strs where each inner list represented a row on the board. Complete the following function, according to its docstring description.

```
def display_column(board, col):
    '''(list of list of strs, int) -> NoneType
    Print column col of the board.  You may assume: 0 <= col < len(board)'''
```

```
    ~~count = 0~~
    column_col=[]
    n=0
    while n!=(len(board)+1):
        column_col += column.append(board[n][col])
        n += 1
    return column_col
```

xo

too far error

+1

append doesn't return anything.

Print

for row in range(len(board)):
    print board[row][col]

Student #: ________________

Question 3. [5 MARKS]

Complete the following function according to its docstring description.

```
def num_start_with(sentences, letter):
    '''(list of lists of strs, str) -> int
    The strs in the lists in sentences are lowercase words
    (e.g., [['hi', 'there'], ['this', 'is', 'fun'], ['hooray']])
    and letter is a single lowercase letter.
    Return the number of words from sentences that start with letter.'''
```

```
    letters = 'abcdefghijklmnopqrstuvwxyz'
    count = 0
    n = 0
    ~~for item in sentences[n]~~
    while ~~it~~ n != (len(sentences) + 1):
        ~~for item in sentences[n]~~
        m = 0
        while m != (len(sentences[n]) + 1):
            if sentences[n][m] in letters:
                count += 1
            m += 1
    n += 1
    return count
```

4

```
count = 0
for sentence in sentences:
    for word in sentence:
        if word.startswith(letter):
            count += 1
return count
```

You may use the space below for rough work. This section will not be marked unless you clearly indicate the part of your work that you want us to mark.

```
n = 0
while n != len(board) + 1
d ~~ob~~ column += column_col.append( board[0][col])
m = 0
for item in sentences
    ~~if [illegible]~~
    ~~while item~~
        n = 0
        while n

~~n = 0~~        len(sentences)
~~for~~      m = 0
             sentences[ m]
             while ~~[illegible]~~ n
```

7.5

Question 4. [13 MARKS]

An employee is keeping track of the dates he is scheduled to work in a file, where each line is of the form:
`YEAR,MONTH,DAY`
`YEAR`s are 4-digits integers; `MONTH`s and `DAY`s are each 2-digit integers.

Here is a sample "schedule file":

```
2012,03,15
2012,03,18
2012,04,01
2012,10,10
```

A "schedule list" is generated based on a "schedule file" like the one describe above.

The "schedule list" for the file above is:
`["15-03-2012", "18-03-2012", "01-04-2012", "10-10-2012"]`.
Note: the date format in the file differs from the list (e.g., `2012,03,15` as compared to `15-03-2012`).

Part (a) [6 MARKS] Complete the following function according to its docstring description.

```
def get_schedule_list(schedule_file):
    '''(file) -> list of strs
    Return a "schedule list" for the "schedule file" schedule_file.'''
```

1.5

~~file~~
~~list = open (schedule_file)~~
~~Schedule_list = []~~
~~While line != "" in schedule_file:~~
~~date = line[2], "-", line[1], "-", line[0]~~
~~schedule_list = schedule_list.append(date)~~
~~schedule_file.readline()~~
~~list.readline()~~
~~return schedule_list~~

```
schedule_list=[]
for line in schedule_file:
    line = line.strip()
    date = line[8:] + "-" + line[5:7]
    + "-" + line[:4]
    schedule_list.append(date)
return schedule_list
```

```
schedule_list=[]   / +0.5
while line != "" in schedule_file:
    date = line[2], "-", line[1], "-", line[0]
    schedule_list = schedule_list.append(date)
    schedule_file.readline()
return schedule_list   / +0.5
```

you can't do this

you didn't split the line , or strip -1.5
so you won't be able
to do this

don't assign to a list

not necessary

Part (b) [4 MARKS] Complete the following function according to its docstring description.

```
def is_booked(schedule_list, proposed_date):
    '''(list of strs, str) -> bool
    proposed_date is a date in the same format as those in a "schedule list".
    Return True if proposed_date is booked (employee is scheduled to work)
    according to "schedule list" schedule_list, and return False otherwise.'''
    n=0
    while n != (len(schedule_list)+1):          ← off by one error, -0.5
        if proposed_date = schedule_list[n]:
            return ~~true~~ True
        n+=1
    return False
```

return proposed_date in schedule_list

3.5

Part (c) [3 MARKS] In the question below, fill in the box with python code that will make the program behaviour match the comments. You may **not** make any other changes to the code.

```
if __name__ == '__main__':
    schedule_file = open('schedule.txt')
    schedule_list = get_schedule_list(schedule_file)

    # Continually prompt the user (using raw_input) to enter a date (in DD-MM-YYYY format)
    # until they enter a date that is not booked.
```

```
date = raw_input()
while date ~~not~~ in schedule_list:
    print "Please enter a date (in DD-MM-YYYY format)", date
return date
```

2.5

```
    print "The date %s is not booked.", (date)
```

```
date=raw_input(Please enter the proposed
 date in DD-MM-YYYY format.)
while is_booked(schedule_list, ~~proposed~~ date):
    ~~p~~date = raw_input(Sorry, that date is
     booked. please enter another date.)
return date
```

Last Name: Qiu First Name: Rui

Short Python function/method descriptions:

```
__builtins__:
  len(x) -> int
    Return the length of the list, tuple, dict, or string x.
  open(name[, mode]) -> file object
    Open a file.
  range([start], stop, [step]) -> list of integers
    Return a list containing the integers starting with start and ending with
    stop - 1 with step specifying the amount to increment (or decrement).
  raw_input([prompt]) -> str
    Read a string from standard input.  The trailing newline is stripped.
file (also called a "reader"):
  F.close() --> NoneType
    Close the file.
  F.read([size]) -> string
    Read at most size bytes; with no size, read until EOF.
  F.readline([size]) -> string
    Read next line, retaining newline; return empty string at EOF.
float:
  float(x) -> float
    Convert a string or number to a floating point number, if possible.
int:
  int(x) -> int
    Convert a string or number to an integer, if possible.  A floating point
    argument will be truncated towards zero.
str:
  str(x) -> str
    Convert an object into its string representation, if possible.
  S.find(sub[,i]) -> integer
    Return the lowest index in S (starting at S[i], if i is given) where the
    string sub is found or -1 if sub does not occur in S.
  S.replace(old, new) -> string
    Return a copy of string S with all occurrences of the string old replaced
    with the string new.
  S.split([sep]) -> list of strings
    Return a list of the words in S, using string sep as the separator and
    any whitespace string if sep is not specified.
  S.startswith(prefix) -> boolean
   Return True if S starts with the specified prefix and False otherwise.
  S.strip() --> string
    Return a copy of S with leading and trailing whitespace removed.
list:
  L.append(x) --> NoneType
    Append x to the end of the list L.
  L.index(value) -> integer
    Return the lowest index of value in L.
  L.insert(index, x) --> NoneType
    Insert x at position index.
```