CSC A08 2012 Midterm Test
Duration — 50 minutes
Aids allowed: none

Student Number: |_|_|_|_|_|_|_|_|_|

Last Name: ____________________ **First Name:** ____________________

Instructor: Brian Harrington

*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 3 questions on 8 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:	______/ 4
# 2:	______/ 6
# 3:	______/10
TOTAL:	______/20

Question 1. [4 marks]

Part (a) [1 mark] What is the output of the following?

```
x = 4
y = x + 5
x = 7
print(x, y)
```

Part (b) [1 mark] What is the output of the following?

```
s = 'CSC-A08 is Fun!'
i = 0
while ((i < len(s)) and (s[i].isalpha())):
    print(s[i])
    i = i + 1
```

Part (c) [1 mark] 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 or add code outside the box.

```
def bypass_line(age, has_vip_card):
    ''' (int, bool) -> bool
    Return True if and only if the person's age is greater than 70 or
    they have a V.I.P. card as indicated by has_vip_card.'''

    return
```

Part (d) [1 mark] 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 or add code outside the box.

```
s1 = "midterm"
s2 = "science"

# Using only s1, s2, concatenation, and indexing and/or slicing, print "test".

print(                                                    )
```

Question 2. [6 MARKS]

Part (a) [4 MARKS]

Read the following function and then follow the first four steps of the Design Recipe to complete the function header and docstring. Give two examples in your docstring.

```
def                                                    :
    '''



    '''
    result = ''
    for ch in s:
        if ch.isupper():
            result = result + ch.lower()
        else:
            result = result + ch.upper()
    return result
```

Part (b) [2 MARKS]

Write a small Python program (not a function) that prompts the user to enter a string, calls the function from Part (a) passing the string that the user entered as an argument, and prints the value returned by the function. You may not change the function.

Question 3. [10 marks]

Part (a) [5 marks] Complete the function according to its docstring.

```
def select_characters(s1, s2, selection):
    ''' (str, str, str) --> str

    Return a new string where each character is a character from either s1
    or s2 chosen based on selection. selection is made up of 1s and 2s and
    indicates whether the character at the corresponding position of the new
    string should be from that position in s1 or that position in s2.
    REQ: s1, s2 and selection are the same lengths.

    >>> select_characters('coat', 'hard', '1221')
    'cart'
    >>> select_characters('pizza', 'hints', '11222')
    'pints'
    '''
```

Part (b) [5 MARKS] A library charges overdue fees for a borrowed book using the following fee schedule:

- less than 4 days late: 1 dollar per day
- 4 to 6 days late: 2 dollars per day (for all days, including the first 3 days)
- more than 6 days late: 3 dollars per day (for all days, including the first 6 days)

Borrowers of books are in one of these age groups: `CHILD`, `ADULT` or `SENIOR`. A `CHILD` gets charged only half of the fees and a `SENIOR` gets charged only one quarter of the fees. An ADULT pays the full fee.

Complete the following function according to the description above and the docstring below.

```
CHILD = 'child'
ADULT = 'adult'
SENIOR = 'senior'

def overdue_fees(days_late, age_group):
    ''' (int, str) -> number

    Return the fees for a book a that is days_late days late for a borrower
    in the age group age_group.

    >>> overdue_fees(2, SENIOR)
    0.5
    >>> overdue_fees(5, ADULT)
    10
    '''
```

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]

Last Name: ______________________ **First Name:** ______________________

Short Python function/method descriptions:

```
__builtins__:
  abs(number) -> number
    Return the absolute value of the given number.
  max(a, b, c, ...) -> value
    With two or more arguments, return the largest argument.
  min(a, b, c, ...) -> value
    With two or more arguments, return the smallest argument.
  input([prompt]) -> str
    Read a string from standard input.  The trailing newline is stripped.  The prompt string,
    if given, is printed without a trailing newline before reading.
int:
  int(x) -> int
    Convert a string or number to an integer, if possible.  A floating point argument
    will be truncated towards zero.

str:
  S.count(sub[, start[, end]]) -> int
    Return the number of non-overlapping occurrences of substring sub in
    string S[start:end].  Optional arguments start and end are
    interpreted as in slice notation.
  S.find(sub[,i]) -> int
    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.isalpha() --> bool
    Return True if and only if all characters in S are alphabetic
    and there is at least one character in S.
  S.isdigit() --> bool
    Return True if and only if all characters in S are digits
    and there is at least one character in S.
  S.islower() --> bool
    Return True if and only if all cased characters in S are uppercase
    and there is at least one cased character in S.
  S.isupper() --> bool
    Return True if and only if all cased characters in S are uppercase
    and there is at least one cased character in S.
  S.lower() --> str
    Return a copy of S converted to lowercase.
  S.replace(old, new) -> str
    Return a copy of string S with all occurrences of the string old replaced
    with the string new.
  S.split([sep]) -> list of str
    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) -> bool
    Return True if S starts with the specified prefix and False otherwise.
  S.strip() --> str
    Return a copy of S with leading and trailing whitespace removed.
  S.upper() --> str
    Return a copy of S converted to uppercase.
```