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 8 pages (including this one). When you receive the signal to start, please make sure that your copy is complete.

• 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.

• If you use any space for rough work, indicate clearly what you want marked.

• You may use a pencil; however, work written in pencil will not be considered for remarking.

# 1: _____/ 7
# 2: _____/ 3
# 3: _____/ 8
# 4: _____/ 7
TOTAL: _____/25

Total Pages = 8
Question 1.  [7 marks]

The following python statements have been executed:

tweet1 = 'Computer science students organize #UofT Hacks to create new solutions'
tweet2 = 'Meet Ross, the @IBM Watson-powered lawyer @IBM'

# docstrings removed to save space
def f(x, y, z):
    return x * y * z + 3
def g(x):
    return x % 2

x = 8

Beside each code fragment in the table below, write what is printed when the code fragment is executed. If the code would cause an error, write ERROR and give a brief explanation.

<table>
<thead>
<tr>
<th>Code</th>
<th>Output or Cause of Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>print('problem' not in tweet1 and 'lawyer' in tweet2)</td>
<td></td>
</tr>
<tr>
<td>print(5 // 3 * 6 == 10)</td>
<td></td>
</tr>
<tr>
<td>print(tweet1[3].islower() and tweet2[-len(tweet2)].islower())</td>
<td></td>
</tr>
<tr>
<td>print(tweet1[3].islower() or tweet2[len(tweet2)].islower())</td>
<td></td>
</tr>
<tr>
<td>print(tweet2[len(tweet2)].islower() or tweet1[3].islower())</td>
<td></td>
</tr>
<tr>
<td>print(g(x))</td>
<td></td>
</tr>
<tr>
<td>print(f(x * g(x), 1, x))</td>
<td></td>
</tr>
</tbody>
</table>
Question 2.  [3 marks]

Assume that variable \( n \) refers to an \texttt{int} and that \( n \geq 0 \). Consider the following python code:

```python
total = 0
i = 0
while i <= n:
    total = total + i
    i = i + 2
```

Rewrite this code so that it uses a \texttt{for} loop instead of a \texttt{while} loop.

You will not need to use much of the blank space on this page.
[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.]
Question 3.  

[8 marks]

In the Republic of Danmania, the cost of mailing a letter is 40 coins for letters weighing up to 30 g (inclusive) and 55 coins for letters over 30 g and up to 50 g (inclusive). Letters weighing more than 50 g cost 55 coins plus an additional 25 coins for each extra 50 g or part thereof. For example, a 100 g letter would cost 80 coins to mail (55 coins for the first 50 g plus an additional 25 coins for the next 50 g). A 101 g letter would cost 105 coins to mail (55 coins for the first 50 g plus an additional 25 coins for the next 50 g and then an additional 25 coins for the extra 1 g, which is part of the next 50 g increment).

Using the Function Design Recipe that you have learned in the course, write a complete python function named `coins_required` that returns the number of coins required to mail a letter with a given integer weight in grams (g).

Your function may assume that when it is called, it is given a positive integer letter weight. You may assume that the `math` module has been imported.
[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.]
Question 4.  [7 marks]

For the purposes of this question, any sequence of consecutive characters that does not contain a blank character is considered to be a word. Using the constant BLANK_CHAR whenever possible, complete the function according to its docstring description.

Reminder: The following expressions are all True:
'a'.upper() == 'A',    '4'.upper() == '4',    ';'.upper() == ';',    and    '.upper() == ' '.

BLANK_CHAR = ' '

def capitalize(s):
    """ (str) -> str

    Return a version of s that has the first character of every word in s converted to upper-case.

    >>> capitalize('february 11th is a GREAT day for writing a test!')
    'February 11th Is A GREAT Day For Writing A Test!'
    >>> capitalize(' 123 ')
    ' 123 '
    """

    >>> capitalize('february 11th is a GREAT day for writing a test!')
    'February 11th Is A GREAT Day For Writing A Test!'
    >>> capitalize(' 123 ')
    ' 123 '
    """
Short Python function/method descriptions:

__builtins__:
  int(x) -> int
  Convert x to an integer, if possible. A floating point argument will be truncated towards zero.
  len(x) -> int
  Return the length of list, tuple, or string x.
  print(value) -> NoneType
  Prints the values.
  range(stop) -> list-like-object of int
  range(start, stop[, step]) -> list-like-object of int
  Return the integers starting with start and ending with stop - 1 with step specifying the amount to increment (or decrement). If start is not specified, the sequence starts at 0. If step is not specified, the values are incremented by 1.
  str(x) -> str
  Return an object converted to its string representation, if possible.

str:
  x in s -> bool
  Produce True if and only if x is in s.
  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.isalnum() -> bool
  Return True if and only if all characters in S are alphanumeric and there is at least one character is 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 lowercase 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 the string 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.startswith(sub) -> bool
  Return True if and only if S starts with the substring sub.
  S.upper() -> str
  Return a copy of the string S converted to uppercase.

list:
  x in L -> bool
  Produce True if and only if x is in list L
  L.append(object) -> NoneType
  Append object to end of list L.