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 5 questions on 6 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.

TOTAL: _____/20

# 1: _____/ 5
# 2: _____/ 2
# 3: _____/ 4
# 4: _____/ 5
# 5: _____/ 4

Total Pages = 6
Question 1. [5 marks]

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

tweet1 = '#uoft_cs Turing award winner Steve Cook'
tweet2 = 'Want cheap snacks? Visit @cssu office in BA2283'

<table>
<thead>
<tr>
<th>Code</th>
<th>Output or Cause of Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>print(tweet2[2] + tweet1[-2])</td>
<td></td>
</tr>
<tr>
<td>print(tweet1[tweet2[-1]])</td>
<td></td>
</tr>
<tr>
<td>print(tweet2[:tweet2.find('?')]</td>
<td></td>
</tr>
<tr>
<td>print('cheap' in tweet2)</td>
<td></td>
</tr>
<tr>
<td>print('of' in tweet1 and 'of' in tweet2)</td>
<td></td>
</tr>
</tbody>
</table>

Question 2. [2 marks]

Complete the example function calls by adding arguments that result in the return values shown. (For each example call, there are several correct answers, and providing any one of them will earn full marks.)

```python
def mystery(L):
    """ (list of str) -> str
    >>> mystery(                  )
    '22'
    >>> mystery(                  )
    '113355'
    """

    result = ''
    for item in L:
        result = result + item[0] + item[0]

    return result
```
Question 3.  [4 marks]

Read the function header and body and then complete the docstring. Give a meaningful function name, the type contract, the description, and two examples that return different values.

```python
def (s):

    i = len(s) - 1
    x = ''
    while i >= 0:
        if s[i].islower():
            x = s[i] + x
            i = i - 1
    return x
```
Question 4.  [5 marks]

Complete this function according to its docstring description.

```python
def hide_alnum(phrase):
    """ (str) -> str

    Return a new string that is the same as phrase but with each letter
    changed to the underscore symbol ('_') and each digit changed to the
    number sign symbol ('#').
    """

    >>> hide_alnum('csc 108')
    '___ ###'
    >>> hide_alnum('I h8 midterms!!!')
    '_ _# ________!!!'
    >>> hide_alnum('R2D2')
    '_#_#'
    """
```
**Question 5.**  [4 marks]

Complete the function according to its docstring description.

```python
def get_at_symbol_indices(s):
    """(str) -> list of int
    Return a list containing the index of each '@' in s.
    """

>>> s = 'Username @uoft_cs is the Dept of Computer Science at UofT'
>>> get_at_symbol_indices(s)
[9]
>>> s = 'See @UofTNews story on 2013 grad @TobiOgunbiyi & his startup @divnotes'
>>> get_at_symbol_indices(s)
[4, 33, 61]
>>> s = "#UofT grad students #startup makes it easier for artists"
>>> get_at_symbol_indices(s)
[]
"""
```
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([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 string `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.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`.

- `L.extend(iterable) --> NoneType`
  Extend list `L` by appending elements from the iterable. Strings and lists are iterables whose elements are characters and list items respectively.