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 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.

• Unless otherwise specified, 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: _____/ 5
# 2: _____/ 4
# 3: _____/ 6
# 4: _____/ 4
# 5: _____/ 6

TOTAL: _____/25

Total Pages = 8
Question 1.  [5 marks]

Fill in the boxes below with what you would see in your Python shell if you were to type the following expressions. If that code would result in an error, then write ERROR and provide a brief explanation in the corresponding box.

| (a) | >>> word = 'breathe'  
     | >>> len(word) |
| (b) | >>> s = 'book'  
     | >>> (s < 'article') and (s[4] == 'k') |
| (c) | >>> l1 = ['a', 'b', 'c']  
     | >>> l1.isalpha() |
| (d) | >>> y = 10 // 4  
     | >>> x = y  
     | >>> y = 20  
     | >>> x |
| (e) | >>> planets = ['earth', 'uranus']  
     | >>> planets[1][2] |
| (f) | # Assume the following function  
     | # definitions:  
     | def f(x):  
     |     return 2 + x  
     |     print("Function f")  
     | def g(y):  
     |     print(f(y))  
     |     print("Function g")  
     | >>> f(2)  
     | >>> g(10) |
Question 2.  [4 marks]

See below the definition for a function named mystery. Complete the two docstring examples below by filling in the boxes with the return values for these function calls. Also, fill in this function’s type contract. The function description has been left out on purpose.

```python
def mystery(val1, val2):
    ""
    # Function description not shown on purpose.
    ""
    for i in range(val1, val2, 1):
        if i % 4 == 0:
            return i
    return -1

>>> mystery(10, 12)

>>> mystery(13, 40)

""
```

for i in range(val1, val2, 1):
    if i % 4 == 0:
        return i
return -1
def compute_points(word):
    """ (str) -> int
   "
    return len(word) - sum(1 for char in word if char.isdigit()) + 5 * sum(1 for char in word if char in ['8', '9'])
Question 4. [4 marks]

Write the body of the following function according to its docstring description.

```python
def replicate_elements(my_list, repl_times):
    """ (list of str, int) -> NoneType
    Precondition: repl_times > 0
    Modify every element of my_list by replicating it repl_times times.
    >>> words = ['hi', '120']
    >>> replicate_elements(words, 2)
    >>> words
    ['hihi', '120120']
    """
```
[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 5.  [6 marks]
Write the body of the following function according to its docstring description.

```python
def compare_word_length(list1, list2):
    """ (list of str, list of str) -> list of str

    Precondition: list1 and list2 both contain the same number of elements.

    Return a new list where every element is either 'same' or 'diff' depending on whether
    the words at the corresponding positions in list1 and list2
    have the same length or different length.
    
    >>> compare_word_length(['abc'], ['csc'])
    ['same']
    >>> compare_word_length(['code', 'in', 'Python'], ['read', 'many', 'books'])
    ['same', 'diff', 'diff']
    """
```
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 or string x.
  max(a, b, c, ...) -> object
      With two or more arguments, return the largest argument.
  min(a, b, c, ...) -> object
      With two or more arguments, return the smallest argument.
  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 string 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].
  S.isalnum() --> bool
      Return True if and only if all characters in S are alphanumeric and there is at least one character 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 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 S converted to lowercase.
  S.upper() -> str
      Return a copy of S converted to uppercase.

list:
  x in L --> bool
      Produce True if and only if object 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.
  L.remove(value) -> NoneType
      Remove first occurrence of value from list L. Raises error if value not present.