Final Exam for V22.0002.XXX
Sample for Practice only

There are two sections, each will be worth 60 points. The maximum score on the test will be 120.
It is essential that you **PUT YOUR NAME ON ALL TEST MATERIALS**. It can be difficult identify the author of an unsigned test and it would be better to avoid this problem.

Note: This sample test approximates the format of the final. However, the actual final may have a different number of questions, a different proportion of easy/difficult questions, etc.

Section 1: Below you will find several Pieces of Code followed by a question and a place to fill in an answer. If there is a bug in the code, write ERROR in the space provided. Otherwise, answer the question. Sample questions 1 and 2 (with the answers filled in) are followed by 7 real questions. **Please Only Answer 5 out of 7 questions.** Questions 1 to 4 are easier than questions 5 to 7. So many people will leave out two from the list: 5, 6 or 7. If you answer more than 5 questions, please cross out answers or otherwise indicate which questions you would like to be counted. If you don't indicate which ones you want, I will just pick the first 5.
Note for sample test: Several places include the words *extra practice*. On the test, these words would say *extra credit* instead.

Sample 1:

```
total = 0

for number in range(1,6):
    total = total + 1
```

Question: What does *total* equal after the loop executes?
Answer: 5

Sample 2:

```
instructions = ['read these instructions','put cup on table', \
                'put teabag in cup', 'pour hot water in cup', \
                'wait ten minutes', 'drink tea']

print(instructions[6])
```

Question: Which instruction is printed?
Answer: ERROR

Question 1:

```
total = 0

for number in range(1,6):
        total = total + number
```

Question: What does *total* equal after the loop executes?
Answer:

Question 2:

```
def spell_out(word):
    for character in word[:-1]:
        print(character, end="+")
    print(word[-1])

spell_out('Establishment')
```

Question: What prints out?
Answer:

Question 3:

```
import re
phonenumbers = {'Rachel\'s Rose Shop': '333-400-3423', 'Donald\'s Daisies': \
                '333-121-2121', 'Pat\'s Petunias': '333-434-5500', \
                'Samuelson\'s Sunflowers': '333-667-9689'}

for key in phonenumbers:
    if re.search('00$',phonenumbers[key]):
        print('My choice is', key+':', phonenumbers[key])
```

Question: What prints out?
Answer:

Question 4:
A file called *list-of-quotations.txt* contains the following lines:

```
Benjamin Franklin | A penny saved is a penny earned
Richard Nixon | I am not a crook
John Lennon | Reality leaves a lot to the imagination
George W. Bush | I know the human being and fish can coexist peacefully
Ralph Nader | The function of leadership is to produce more leaders, not more followers
```

Given the following python code:

```
import os
def enter_line_in_dictionary(line):
        clean_line = line.strip(os.linesep)
        line_list = clean_line.split('|')
        for index in range(len(line_list)):
                line_list[index] = line_list[index].strip(' ')
        quotation_dictionary[line_list[0]] = line_list[1]

def read_in_quotations(file):
        global quotation_dictionary
        quotation_dictionary = {}
        instream = open(file,'r')
        for line in instream:
                enter_line_in_dictionary(line)
        instream.close()

read_in_quotations('list-of-quotations.txt')
```

Question: What is the value of *quotation_dictionary*?

Question 5:

```
def ed_exceptions = {'light': 'lit', 'arise': 'arose', 'rise':'rose',\
                     'sit':'sat', 'babysit':'babysat', 'bite':'bit', \
                     'backslide':'backslid', 'bear': 'bore', 'beat': 'beat',\
                     'read': 'read', 'become': 'became', 'come': 'came',\
                     'fall':'fell','tell':'told'}


def add_ed(verb):
    if verb in ed_exceptions:
        return(ed_exceptions[verb])
    elif verb[-1] == 'e':
        return(verb+'d')
    elif (len(verb) > 2) and (verb[-1] in 'bdglmnprt') \
         and (verb[-2] in 'aeiou') and (not (verb[-3] in 'aeiou')):
        return(verb+verb[-1]+'ed')
    elif (len(verb) > 2) and (verb[-1] == 'y') and (not (verb[-2]  in 'aeiou')):
        return(verb[:-1]+'ied')
    else:
        return(verb+'ed')

for word in ['fix','touch','spill','pat','find','tell','have','try']:
    print(add_ed(word))
```

Question: What is printed out by the for loop?
Answer:

Question 6:

```
def sails(width,columns, rows):
     for row in range(rows):
          for number in range(1,width+1):
               for column in range(columns):
                    print('~'*number+' '*(width-number),end='')
               print()

sails(5,3,3)
```

Question: What is printed out? (It can be a little sloppy, as long as I understand it.)
Answer:

Question 7:

```
import turtle

my_screen = turtle.Screen()
my_turtle = turtle.Turtle()


def regular_shape (side_number, side_length):
    angle = 360/side_number
    my_turtle.pd()
    for number in range(side_number):
        my_turtle.fd(side_length)
        my_turtle.left(angle)
    my_turtle.pu()

def super_regular_shape(side_number, side_length):
    angle = 360/side_number
    for number in range(side_number):
        regular_shape(side_number,side_length)
        my_turtle.fd(side_length*2)
        my_turtle.left(angle)

super_regular_shape(4,100)
```

Question: Draw the shape that the turtle draws. (The drawing should be approximate.) Extra Practice: What shapes are drawn for other values of side_number, e.g., 3 or 5?
Answer:

Section 2: Write Programs as specified. Choose two out of the three possible programs.

Question 8: Write a 'Madlibs' program, given the text below. Your program should first ask the user for instances of various parts of speech or semantic classes (person name, verb, noun, ...). The program should replace the appropriate words in the text with the user's words. If a word in the original text is repeated, the replaced word should repeat as well. The text follows:

Mary had a little lamb, little lamb, little lamb

Mary had a little lamb, whose fleece was white as snow

For example, suppose you select: *Mary, little, lamb, fleece, white,* and *snow* as words that should be replaced. Then a user's choices may yield the following result:

Herman had a blue pencil, blue pencil, blue pencil

Herman had a blue pencil, whose lead was red as fire

Question 9: Write a program that generates 5 random playing cards (a poker hand). Make sure that the program does not generate the same card twice. For example, it should not be possible for the hand to contain two instances of the King of Hearts. Remember there are a total of 52 different cards: 13 different faces times 4 different suits.

The face cards include cards numbered 2 through 9, as well as *Ace, Jack, Queen* and *King* (for a total of 13).

The four suits are: *Clubs, Diamonds, Hearts* and *Spades*.

Write an additional function to evaluate the hand as follows:

- If all of the cards in the hand have the same suit, then return 'Flush'
- Determine how many cards have the same face:
  - If four cards have the same face, return '4-of-a-kind'
  - If three cards have the same face, return '3-of-a-kind'
  - If two cards have the same face, return '2-of-a-kind'

Note that this is a minor simplification of the real problem. You do not need to identify: high card, straights, straight-flushes, two-pairs, or a full-house. However, you can add these for extra practice.

The required definitions for extra practice are:

- For purposes of ordering cards, it is assumed that: J,Q and K are equivalent to 11, 12 and 13. Furthermore, A can either be treated as 1 or 14.
- a straight is five consectutively numbered cards, e.g., 2, 3, 4, 5, 6 or the sets: {A,2,3,4,5}, {9,10,J,Q,K}, {10,J,Q,K,A}.
- a Straight-Flush is a hand that simultaneously meets the definition for 'Straight' and 'Flush', e.g., {A,2,3,4,5} where all the cards have the suit 'Hearts'.
- two pairs refer to when there are two sets of pairs, e.g., a hand that consists of two instances of 2 and two instances of K and one instance of something else.
- a full-house is a hand that includes one instance of 3-of-a-kind and one pair, e.g., {3,3,3,4,4}.
- a hand that does not meet any of these requirements always has one card that is higher than the others (as per the first bullet). This is the high card.

Question 10: Write a program that tries to guess a number from 1 to 1000. The player should (secretly) select a number between 1 and 1000 and not tell the computer program. Then the program will make a series of guesses. The player should indicate whether each of the program's guesses is: correct, too high or too low. The program's guess are counted. The game ends when the program's guesses correctly.

The program should use the following strategy to guess the correct number:

1. It should keep selecting an integer that is approximately in the middle of the highest and lowest currently possible. Eventually, this should result in it guessing the number correctly.
2. The initial low and high that the computer is assuming should be set to 1 and 1000.
3. If the user indicates that the guess is too high, the computer should reset the current high to 1 below its guess.
4. If the user indicates that the guess is too low, the computer should reset the current low to 1 above its guess.