Sample Midterm Exam 1 for V22.0002

Name: ________________________________
Net ID ________________

There are 2 sections, the first section is worth 40 points and the second section is worth 60 points, for a total of 100. It is essential that you PUT YOUR NAME AND NET ID ON ALL TEST MATERIALS. It can be difficult to identify the author of an unsigned test and it would be better to avoid this problem.

There is a GLOSSARY OF TERMS at the end of the test. Please feel free to look up some of the basics in this glossary. I will also answer any reasonable look-up style questions as I am more interested in your ability to reason than your ability to memorize.

Section 1: (40 points) Each example consists of Python code, followed by questions and places to fill in answers. Please read all questions carefully and answer them as precisely as possible.

Assume that there are no bugs in the code that will make the system crash, although the code may not solve a problem perfectly. If you find anything that you think is a bug, there is either a typo (and I should fix it for everyone) or you are mistaken.

Sample Question A:

```python
output = '1'+'1'
```

Question: What is the value of the variable `output`?
Answer: '11'

Note: Attention to detail is important. The quotes indicate that it is a string. Partial credit is possible. For example, leaving out the quotes would have lost just a little bit, but answering 2, would have resulted in an incorrect answer.

Question 1

```python
def print_5s():
    print(5,5,5,sep='*',end='**')
    print(5,5,5)

print_5s()
```

Question: What prints out by executing the code above (approximately)?
Answer:

Question 2

```python
def nine_complement(digit):
    return(9 - digit)

def nine_complement_string(number_string):
    output = ''
    for digit in number_string:
        output = output+str(nine_complement(int(digit)))
    return(output)

output = nine_complement_string('0123456789')
```

Question: What does `output` equal?
Answer:
def guess_a_letter(guessed_letters):
    letter = input('Guess a letter: ')
    letter = letter.lower()
    while (letter in guessed_letters) or (not (letter in 'abcdefghijklmnopqrstuvwxyz')):
        print(letter, 'is not a valid letter. ', 'Please try again.')
        letter = input('Guess a letter: ')
        letter = letter.lower()
    return(letter)

def print_body_part_message (number):
    ## full list [Head, left eye, right_eye, mouth, neck, left arm, right arm, torso,
    if number == 1:
        print('Head')
    elif number == 2:
        print('Head, left eye')
    elif number == 3:
        print('Head, left eye, right_eye')
    elif number == 4:
        print('Head, left eye, right_eye, mouth')
    elif number == 5:
        print('Head, left eye, right_eye, mouth, neck')
    elif number == 6:
        print('Head, left eye, right_eye, mouth, neck, left arm')
    elif number == 7:
        print('Head, left eye, right_eye, mouth, neck, left arm, right arm')
    elif number == 8:
        print('Head, left eye, right_eye, mouth, neck, left arm, right arm, torso')
    elif number == 9:
        print('Head, left eye, right_eye, mouth, neck, left arm, right arm, torso, left')
    elif number == 10:
        print('Head, left eye, right_eye, mouth, neck, left arm, right arm, torso, left

def hang_man(word):
    word = word.lower()
    guessed_letters = ''
    incomplete_word = len(word)*'_'
    unguessed_letters = 26
    incorrect_letters = 0
    while incorrect_letters<10:
        letter = guess_a_letter(guessed_letters)
        guessed_letters = guessed_letters+letter
        if letter in word:
            replacement_incomplete_word = ''
            for num in range(len(word)):
                if word[num] == letter:
                    replacement_incomplete_word=replacement_incomplete_word+letter
                else:
                    replacement_incomplete_word=replacement_incomplete_word+incomplete_word
            incomplete_word = replacement_incomplete_word
            print('Your letter is correct. The word so far is: ',incomplete_word)
        else:
            incorrect_letters += 1
            print(letter, 'is not in the word. You have', incorrect_letters, 'incorrect guesses remaining')
            complete_word = incomplete_word.replace('_', letter)
            print('The word is: ', complete_word)
incorrect_letters = incorrect_letters+1
if incomplete_word == word:
    print(’You Win’)
    return(’Hurray’)
else:
    print_body_part_message(incorrect_letters)
    print(’You Lose’)

hang_man(’chicken’)

Question 3a: What does the function guess_a_letter do with the variable guessed_letters?

Question 3b: What are the 2 possible conditions in which the game ends?

Question 3c: Write down the user input and what the system prints out for a possible game in which the word being guessed is “chicken” and the user guesses at least 3 incorrect letters and at least 3 correct letters.

Question 4

output = 50
    for number in range(6):
        output = output - number

Question: What does output equal after the loop executes?

Answer:
def vending_machine(money, price):
    if (money < price):
        print('Not enough money')
    elif (money == price):
        print('There is no change.')
    else:
        dollars = 0
        quarters = 0
        dimes = 0
        nickels = 0
        pennies = 0
        change = money - price
        final_output = change
        dollars = change // 1
        change = change - dollars
        change = change * 100  ## converting to cents
        ## intial amount of money is in dollars
        ## for coins smaller than a dollar,
        ## is is more convenient to calculate
        ## in terms of cents.
        if change > 0:
            quarters = change // 25
            change = change - (quarters * 25)
            if change > 0:
                dimes = change // 10
                change = change - (dimes * 10)
                if change > 0:
                    nickels = change // 5
                    change = change - (nickels * 5)
                    pennies = change
        print('Your change is: ')
        if dollars > 0:
            print(dollars, 'dollars')
        if quarters > 0:
            print(quarters, 'quarters')
        if dimes > 0:
            print(dimes, 'dimes')
        if nickels > 0:
            print(nickels, 'nickels')
        if pennies > 0:
            print(pennies, 'pennies')
        return(final_output)

output = vending_machine(5.00, 4.34)

Question 5a: What would be printed by the print statements produced by the code above?
Answer:

Question 5b: What would does the global variable output equal after the code above executes?
Answer:
Section 3: Write Programs as specified.

Question 6: This program calculates the amount of fabric needed to create a sweater for a dog. For simplicity, the program calculates the amount of material required to make a hollow cylinder, closed at the top and bottom, based on some measurements of the dog. Of course when the sweater is made, holes will be cut into the cylinder for the dog's head and legs, and other adjustments will be made so it fits properly.

The height of the cylinder is the distance from the base of the neck to the base of the tail. The diameter is the height of the thickest portion of the dog's chest. The formula for the area of the surface of a cylinder is \((\text{height} \times \pi \times \text{diameter}) + (2 \times \pi \times \text{radius}^2)\). The program should ask the user for the appropriate input and print the appropriate output in a way the user can understand. The function should return the total amount of fabric in square inches or square centimeters.
**Question 7:** This program will center three lines of text as follows. The user is asked to provide three sentences of no more than 65 characters. The program then centers each line by adding between 0 and 32 spaces to the beginning and then printing it. For example, if one of the sentences was ‘Hello World’, the system would add 27 spaces to the beginning of the string before printing. Note that the function: `len(string)` returns the number of characters in a string. Furthermore, it may be necessary to consistently round the number of spaces added either up or down in order to center the strings consistently. Thus the centering will be as close as possible, but not perfect.
Question 8: Write a program that will ask a user several questions about how nice they are. The program will then calculate how nice they are in Standard Niceness Units (SNUs), according to guidelines devised by the World Council on Niceness (WCN). These guidelines are represented in the flowchart below. The flowchart indicates when questions should be asked and how many SNUs are added or subtracted to the total SNU score based on the answers. The questions are given as global variables. You should use these variables instead of writing out all the questions (it will save a lot of time). At your option, you may also use (without writing it yourself), the function `yes_or_no`. You can vary from the flowchart if it doesn’t matter logically. For example, you could ask the user questions 1, 2, 3, 4, 5 and 6 and use the answers (stored as variables) in the if/else statements of your decision tree if this seems easier to you.

The `yes_or_no` function and global variables:

```python
def yes_or_no(question):
    answer = ''
    while not (answer in ['Yes','yes','No','no']):
        print('Please answer the following question "yes" or "no"."
        answer = input(question)
        if answer in ['Yes','yes']:
            return(True)
        else:
            return(False)
```

```python
question1 = 'Do you always share your candy? '
question2 = 'Do you clean up after yourself most of the time? '
question3 = 'Do you give money to charity? '
question4 = 'Do you spend some of your time doing charitable work? '
question5 = 'Do you care about non-human animals? '
question6 = 'Would you commit a crime against another person for personal gain? '
question1a = 'Do you always share your food? '
question2a = 'Do you clean up after other people? '
question3a = 'Would you give up all your money to help a friend in serious need? '
question3a1 = 'Would you do the same for a stranger? '
question4a = 'Would you dedicate your life to a charitable cause and be poor yourself? '
question5a = 'Do you believe in eating all non-human animals? '
question5a1 = 'Do you believe in eating some non-human animals? '
question6a = 'Would you steal so much that your victim becomes poor? '
question6b = 'Would you murder another person for profit? '
```

Some hints for following the flow chart: questions 1,2,3,4,5,6 are mandatory. Regardless of whether question 1 is answered with 'yes' or 'no', question 2 must be asked. Regardless of whether question 2 is answered 'yes' or 'no', question 3 must be asked, etc. However, other questions are only asked under certain circumstances. In most cases a positive answer yields more questions, but not always. Usually, positive answers cause points to be added to the SNU (total Standard Niceness Units), but occasionally, positive answers cause points to be subtracted.

The flow chart is on the following page.

For extra credit, you can complicate the decision tree further by adding five additional questions. Only add questions that would be asked if previous questions are answered No..