COSC 101, Exam #2 Practice

Name & Section: _____

Please write your name & section above. Do not start the exam until instructed to do so.

You have 50 minutes to complete this exam.

There are 4 questions and a total of 39 points available for this exam. Don't spend too much time on any one question.

Since indentation is important in Python, please be sure that your use of indentation is obvious for any code you write.

If you want partial credit, show as much of your work and thought process as possible.

If you run out of space for answering a question, you can continue your answer on one of the blank pages at the end of the exam. If you do so, be sure to indicate this in two places: (1) below the question, indicate which blank page contains your answer, and (2) on the blank page, indicate which question you are answering.

Question	Points	Score
1	13	
2	11	
3	5	
4	10	
Total:	39	

1. **While loop tracing**

   (a) (3 points) The following function is meant to calculate and return the sum of all integers between 1 and n (inclusive), but there are lines missing from the function. As a result, the function has a runtime error (variables not defined) and a semantic error (it does not calculate the sum correctly). Write three lines of code to fix the errors (please be careful about your indentation).

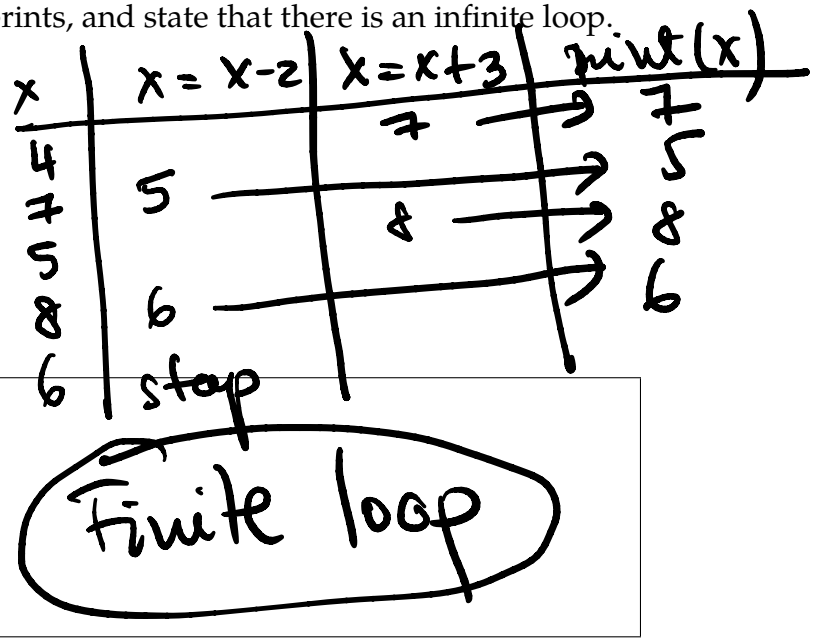
```
def sum_of_n(n):



    while 0 <= num <= n:
            sum += num



    return sum
```

Solution:

```
def sum_of_n(n):
    sum = 0
    num = 1
    while 0 <= num <= n:
        sum += num
        num += 1
    return sum
```

   (b) (5 points) What does this program print? If there is an infinite loop, indicate the first 4 lines of what the program prints, and state that there is an infinite loop.

```
x = 4
while x != 6:
        if x > 6:
                x = x - 2
                print(x)
        else:
                x = x + 3
                print(x)
```



Solution:

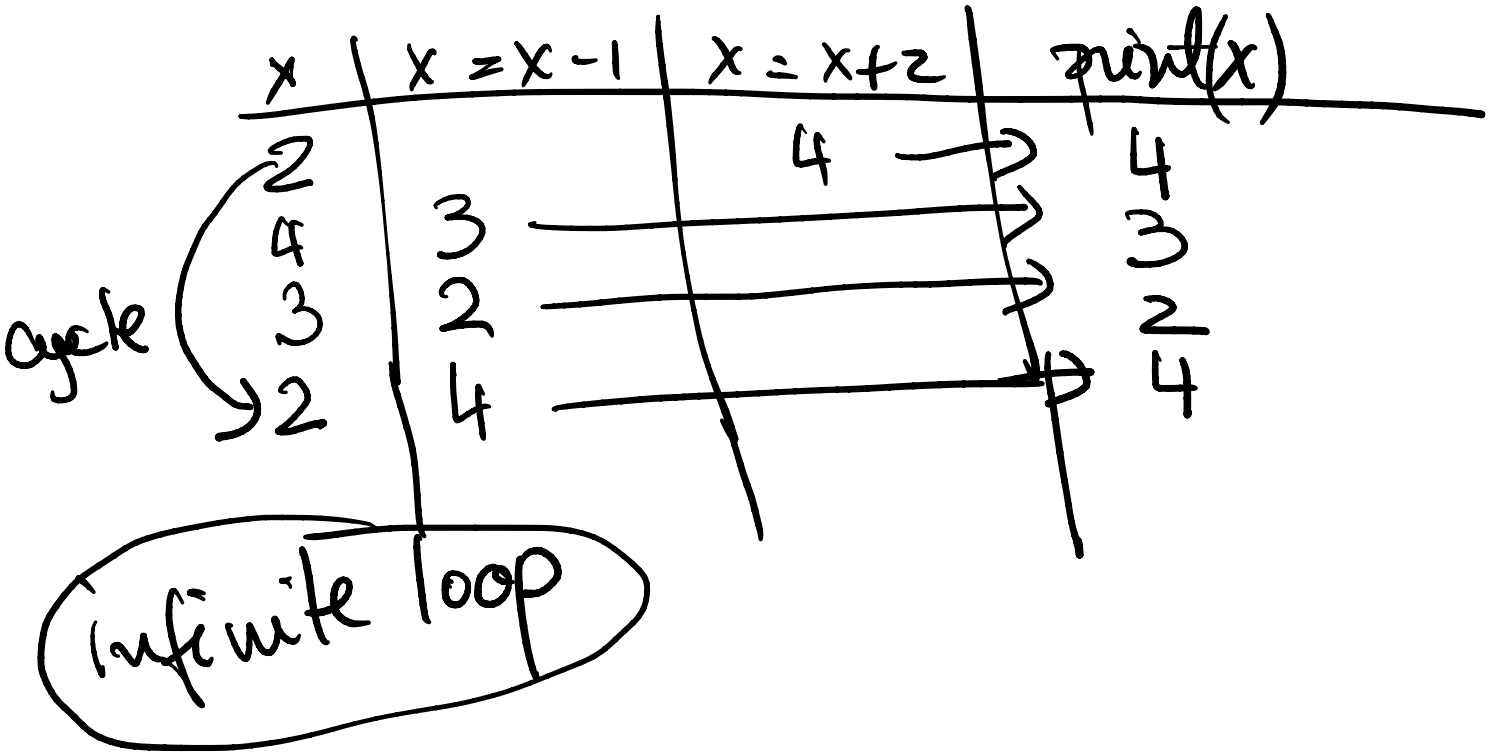
```
7
5
8
6
```

(c) (5 points) What does this program print? If there is an infinite loop, indicate the first 4 lines of what the program prints, and state that there is an infinite loop.

```
x = 2
while x != 0:
    if x > 2:
        x = x - 1
        print(x)
    else:
        x = x + 2
        print(x)
```

Solution: Infinite loop. First 4 lines of output are:

```
4
3
2
4
```



2. (11 points) For this problem, select one line of code from each of the pairs of lines of code below to solve the following problem:

   Your instructor is looking to buy a stove for the winter and she has tasked you to write a program that reads from the input names of companies that produce stoves and prices of their items, and for each company prints the average cost of a stove. Here are the statement pairs to pick from:

```
A1  return total / count
A2  return average_price

B1  total = []
B2  total = 0

C1  count = 0
C2  count = -1

D1      count = 1
D2      count += 1

E1  company_name = 'bye'
E2  company_name = input('Type company name:')

F1  price == float(input('Enter price of stove (-1 when done): '))
F2  price = float(input('Enter price of stove (-1 when done): '))

G1      average_price = average(prices)
G2      average_price = get_average_stoves()

H1  if company_name is not 'bye':
H2  while company_name != 'bye':

I1      print(f'The average cost for {company_name} is {average_price}')
I2      return('The average cost for {company_name} is {average_price}')

J1  if price != -1:
J2  while price != -1:

K1      total += price
K2      total = price
```

   Select only 11 lines of code from above, and **only one line from each pair**. You may write only the line identifiers (*e.g.*, E2) below, or write out the code. Your selections should *only* go on numbered lines below (see next page).

```
def get_average_stoves() -> float:
    '''This function gets invoked for each company and reads
all the item prices for this company from the user input (until
the user enters -1) and returns their average.'''
```

1

2

3

4

5

6

```
        price = float(input('Enter price of stove (-1 when
done): '))
```

7

```
def main() -> None:
    '''This function reads from the user input the name of each
company until 'bye' is entered and calls get_average_stoves()
to get the average for each company and then output this
information'''
```

8

9

10

11

```
        company_name = input('Type company name:')

main()
```

Solution:

- B2 `total = 0`
- C1 `count = 0`
- F2 `price = float(input('Enter price of stove (-1 when done): '))`
- J2 `while price != -1:`

- D2 `count += 1`
- K1 `total += price`

- A1 `return total / count`
- E2 `company_name = input('Type company name:')`
- H2 `while company_name != 'bye':`
- G2 `average_price = get_average_stoves()`
- I1 `print(f'The average cost for {company_name} is {average_price}')`

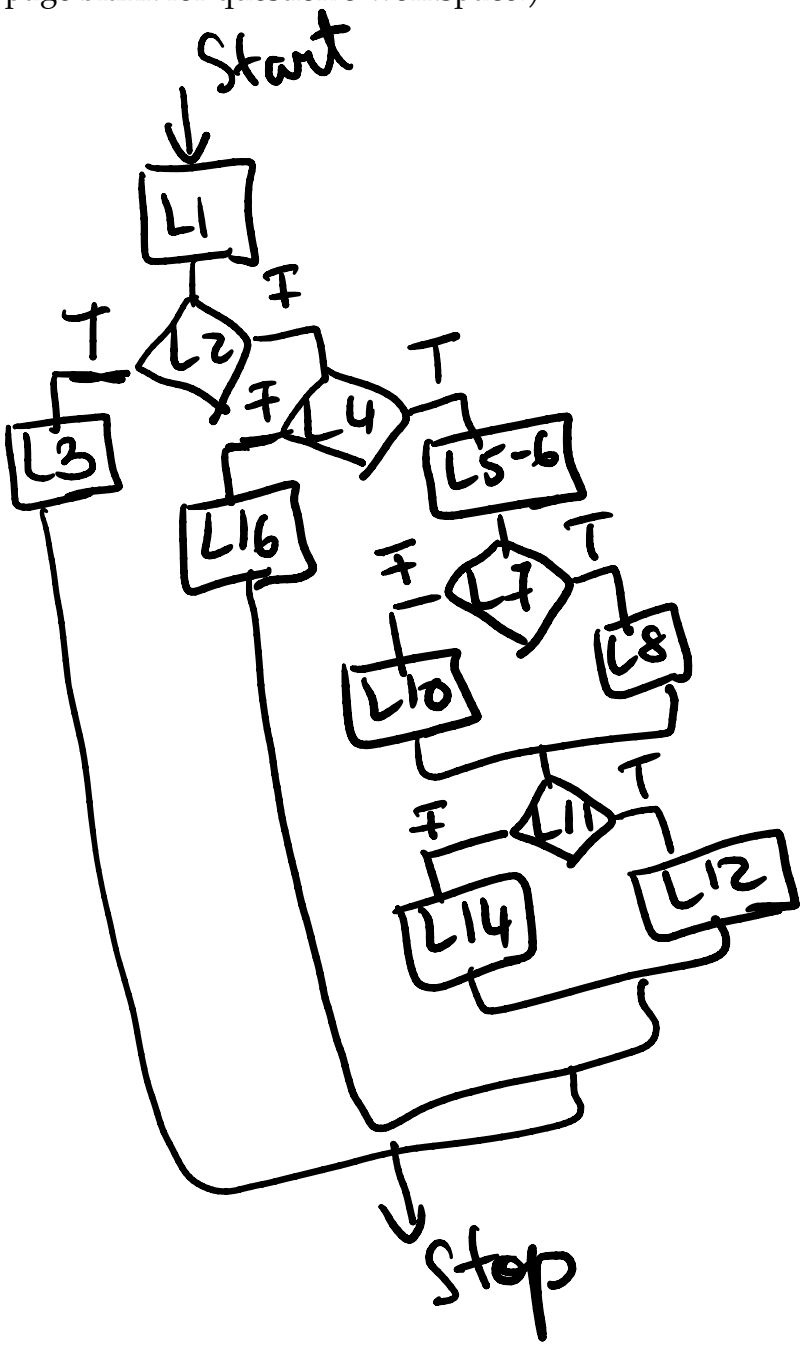
3. (5 points) What is the output of calling `mystery` function with values 240 (for pumpkins), 105 (for people) and 1 (for minperperson)? Draw the control flow chart. Provide different values to pass as arguments to the function in order to get different outcomes.

```
def mystery(pumpkins: int, people: int, minperperson: int) -> None:
    print(f"Out of {pumpkins} (pumpkins), {people} (people),
                                and {minperperson} (minperperson):")
    if (pumpkins < people * minperperson):
        print("We need more pumpkins!")
    elif (pumpkins >= people * minperperson):
        print("We have enough pumpkins.")
        pumpkinsperperson = pumpkins // people
        if (pumpkinsperperson >= 1):
            print("Each person may carve", pumpkinsperperson,
                                            "pumpkin(s).")
        else:
            print("At least", 2 * people - pumpkins, "people must share
                                            a pumpkin.")
        if (pumpkinsperperson == 1 and pumpkins % people == 0):
            print("There will be no extra pumpkins.")
        else:
            print("There may be extra pumpkins.")
    else:
        print("We have bigger problems than pumpkins!")
```

Solution:

```
Output with 240 (pumpkins), 105 (people), and 1 (minperperson):
We have enough pumpkins.
Each person may carve 2 pumpkin(s).
There may be extra pumpkins.

Three new values to get different output:
120 (pumpkins), 100 (people), 2 (for minperperson)
```

(This page blank for question 3 workspace.)



4. (10 points) Write a function called `diff_type` that takes in a list of numbers and prints + when the difference between two consecutive elements of the list is positive, = when they are equal, and −, otherwise.

   For example, `diff_type([3,4,1,7])` must output `'+−+'` because the values are increasing between 3 and 4 and 1 and 7 (which warrants a '+'), and decreasing between 4 and 1 in the middle (which warrants a '-');`diff_type([12,30,30])` must output `'+=',diff_type([32,16,8,4,2,1,0])` must output `'------'`

Solution:

```
def diff_type(lst: list) -> None:
    for i in range(len(lst)-1):
        if lst[i+1] - lst[i] > 0:
            print('+', end='')
        elif lst[i+1] - lst[i] < 0:
            print('-', end='')
        else:
            print('=', end='')
```