

Name: \_\_\_\_\_

**Rules and Hints**

- You may use one handwritten 8.5 x 11" cheat sheet (front and back). This is the only additional resource you may consult during this exam. *No calculators.*
- When you write code, be sure that the indentation level of each statement is clear.
- Explain/show work if you want to receive partial credit for wrong answers.
- As long as your code is correct, you will get full credit. No points for style.
- As always, the SSU rules on academic integrity are in effect.

<b>Problem</b>	<b>Max Score</b>	<b>Your Score</b>
<i>Problem 1:</i> Trace Code	20	
<i>Problem 2:</i> Trace Functions	20	
<i>Problem 3:</i> Define functions	30	
<i>Problem 4:</i> Write a complete program	30	
<b>Total</b>	100	

**Problem 1: Trace Code (20 points)**

Write what will be printed to the screen when each of the following snippets of code is executed in PyCharm or in the Online Python Tutor.

Write your final solution in the box provided

Do not write any scratch-work in the solution box.

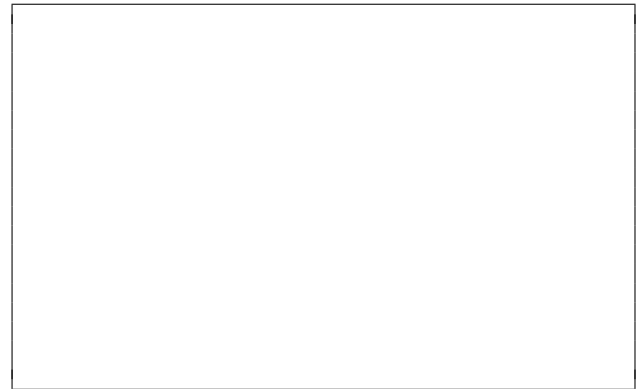
In your solution, be very precise with spacing, line breaks, etc.

Treat each sub-problem as an independent question.

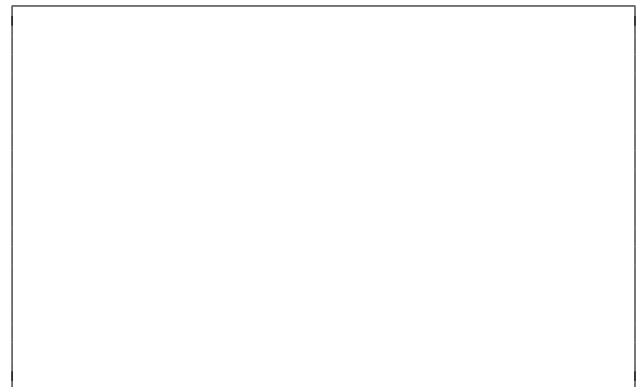
All questions in this section are worth 4 points.

**Problem 1A**

```
x = -5
y = 5
while x + y < 10:
    print(x, y)
    x = x + 3
```

**Problem 1B**

```
s = 'Noma Nation'
print(len(s))
print(s[len(s)-1])
print(s[:2]*3)
print(s[-8:-1])
```



**Problem 1C**

```
s = ['Snoopy', 'Lucy',  
     'Linus', 'Marcie',  
     'Peppermint Patty']  
print(len(s))  
print(s[2])  
print(len(s[2]))  
print(s[2][3])
```

**Problem 1D**

```
A = [1,2,3,4,5]  
B = A  
C = A[:]  
B[0] = -5  
C[2] = min(A)  
print(B)  
print(C)
```

**Problem 1E**

```
L = []  
for i in range(4):  
    L.append(i*2)  
print(L)
```

**Problem 2: Trace Functions (20 points)**

Write what will be printed to the screen when each of the following snippets of code is executed in PyCharm or in the Online Python Tutor.

Write your solution in the box provided.

Do not write any scratch-work in the solution box.

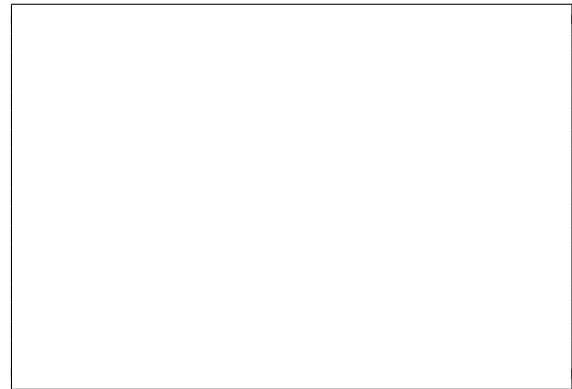
In your solution, be very precise with spacing, line breaks, etc.

Treat each sub-problem as an independent question.

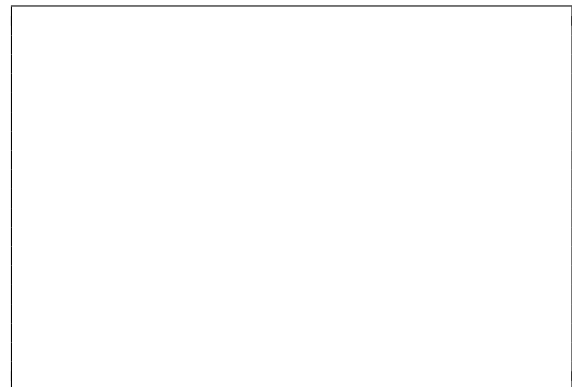
All questions in this section are worth 5 points.

**Problem 2A**

```
def mod(m, n):  
    return m % n  
  
n = 10  
m = 6  
z = mod(n, m)  
print(z)
```

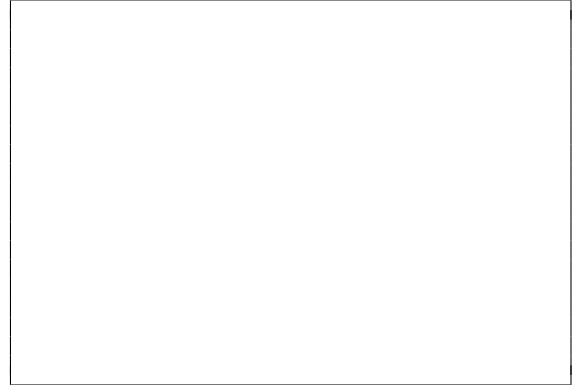
**Problem 2B**

```
def mystery(s):  
    word = ''  
    for c in s:  
        word = word+c*2  
    return word  
  
answer = mystery('312')  
print(answer)
```

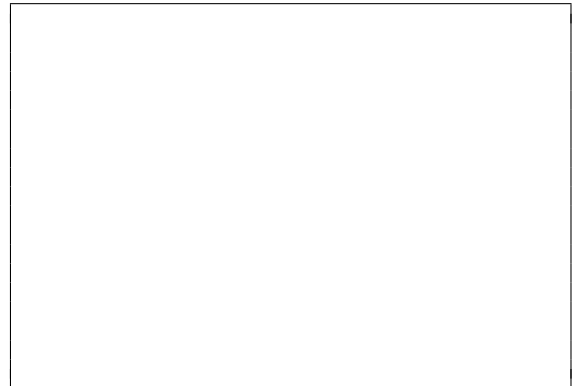


**Problem 2C**

```
def getSecret(A):  
    return A[len(A)//2+1]  
  
A = [[0, 1], [2, 3],  
      [4, 5, 6], [7, 8]]  
L = getSecret(A)  
print(len(A), len(L), L)
```

**Problem 2D**

```
def insertSum(L):  
    for i in range(len(L)):  
        L[i] = sum(L)  
  
A = [1,2,3]  
insertSum(A)  
print(A)
```



**Problem 3: Define functions (30 points)**

Define functions to perform the following tasks, obeying the stated requirements, including:

- Assume that the math library has been imported for you.
- Do not ask the user for input unless the specification explicitly requires it.
- Do not print anything unless the specification explicitly requires it.
- Do not call any function unless the specification explicitly requires it.

**Problem 3A (5 points)**

Define a function named `rectangle_perimeter` that:

- Takes 2 parameters: the width and height of a rectangle
- Returns the perimeter of the rectangle. The perimeter is equal to twice the sum of the width and the height.

Example of calling this function:

```
rectangle_perimeter(2, 3) returns 10
```

**Problem 3B (5 points)**

Define a function named `total_words` that:

- Takes 1 parameter: a string
- Returns the number of words in that string

Example of calling this function:

```
total_words("The sun also rises.") returns 4
```

**Problem 3C** (10 points)

Define a function named `getColSum` that:

- Takes 2 parameters: a nested list (i.e., a 2-dimensional list) and an integer `k`
- Returns the sum of all the elements in column `k`

You may assume that column `k` exists.

Example of calling this function:

Let `L = [[5,10,15], [1,2,3], [7,8,9]]`, then  
`getColSum(L,0)` returns 13

**Problem 3D** (10 points)

Define a function named `count_ells` that:

- Takes 1 parameter: a list of strings
- Returns the total number of times the character `L` or `l` appears in the strings

Example of calling this function:

```
Let my_list = ["Llamas across the world", "all", "say hello", "to me"]
then count_ells(my_list) returns 7
```



**Extra Page ...**

**Problem 4: Write a complete program (30 points)**

For this problem, you must write a complete program. This includes writing a docstring, logic in `def main()`, a call to `main()`, any necessary library imports, etc.

If you get stuck, try to maximize your partial credit. To get full credit, your functions should call each other where appropriate and avoid duplicating code. In all examples, user input is underlined.

Your program should contain the following functions:

- `get_names`:
  - Takes no parameters. Prompts the user for names until the user enters a blank line. Each line of user input should be considered a name. Returns a list of the names the user entered (not including the blank line).

Example: A call to `get_names` with the below sample user input

```
Enter a name: Harry James Potter
Enter a name: Hermione Granger
Enter a name:
```

should return the list `['Harry James Potter', 'Hermione Granger']`.

- `name_is_increasing`:
  - Takes 1 parameter: a string holding a person's name. Returns `True` if the person's *last* name is as long as or longer than their *first* name, or if their name is only one word long; otherwise, it returns `False`.

Example: For `'Padma P. Patil'`, the function returns `True`.

Example: For `'Parvati P. Patil'`, the function returns `False`.

Example: For `'Moaning Myrtle'`, the function returns `False`.

Example: For `'Voldemort'`, the function returns `True`.

- `main`:
  - Calls `get_names` to get a list of names from the user. Prints these names in reverse order and stating, for each name, if it has the property checked by `name_is_increasing`.

Sample input/output:

```
Enter a name: Harry J. Potter
Enter a name: Hermione Granger
Enter a name: Ronald Weasley
Enter a name:
```

```
Ronald Weasley: yes
Hermione Granger: no
Harry J. Potter: yes
```

*Start your solution on the next page...*

**Problem 4, continued ...**

**Problem 4, continued ...**