CS 115 Exam 1 (Section 1) Fall 2017 Thu. 10/21/2017

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.
- Clearly box or mark your final answer for each question.
- 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.

Problem	**Max Score**	**Your Score**
Problem 1: Trace Code	40	
Problem 2: Write snippets of code	30	
Problem 3: Write a complete program	30	
Total	100	

Problem 1: **Trace Code (40 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.

Be very clear with spacing, line breaks, etc.
Treat each sub-problem as an independent question.
All questions in this section are worth 5 points.

Problem 1A

```
dozen = 12
print(2*dozen + 1)
```

Problem 1B

```
z = 5 ** 2
print(z)
z = z - 3
print(z)
```

Problem 1C

```
a = 13
b = 5
c = a % b
d = a // b
print(c,d)
```

Problem 1D

```
print(2, 2, '=', 4, sep='*')
```

Problem 1E

```
pizza = 'pie'
print(pizza + 'pie')
print('pizza', pizza)
```

Problem 1F

```
k = 20
for j in range(2,5):
    print(j)
    k = k + j
print(k)
```

Problem 1G

```
for j in range(2,5):
    for k in range(2):
        print(j,k)
    print('!')
```

Problem 1H

```
x = 10
print("I'd like to order")
if (x < 100):
    print(100-x, "more")
if (x < 50):
    print(50-x, "more")
elif (5 < x < 20):
    print("a few more")
else:
    print("no more")
print("pancakes")
```

Problem 2: **Write snippets of code (30 points)**

Write snippets of code to do the following. Your code should only print the requested output. You will lose points for printing additional output.

You can assume that all your snippets are enclosed within a main function and that any necessary libraries have been imported. You only need to write the specific lines of code that accomplish each task.

Problem 2A (8 points)

Painting a column: You are painting a column for the scenery of a play.

Prompt the user to input the radius r (in feet) and height h (in feet) of the column, then calculate the total surface area of the column (in square-feet) and print the total surface area. Do not round any values or calculations.

The total surface area to be painted is given by the formula $A = 2\pi rh + 2\pi r^2$. (The column gets moved during the play, so its total area—including top and bottom—will get painted.) For example, using a radius of 1.0 feet and a height of 1.5 feet yields a total paintable area of about 15.708 square feet.

Problem 2B (10 points)

Ask the user to enter 100 integers, one at a time. After the user has entered all the integers, print the total number of *negative integers*, print the total number of *odd integers* and then print the total number of integers that were *divisible by 5*.

For example, if the sequence 5, -2, 4, -15, 6, 10 was entered, the program should first output 2 (since -2 and -15 are negative), then output 2 (since 5 and -15 are both odd) and then output 3 (since 5, -15, 10 are divisible by 5).

Problem 2C (12 points)

Sum of cubes: write a calculator that computes the sum of the cubes of a set of sequences, where each sequence is just the first n integers.

- Prompt the user for the number of sequences to be calculated.
- For each sequence:
  - Prompt the user for the upper-bound of the sequence, n. (You may assume they always enter an integer $n \geq 1$.)
  - Calculate the sum of the cubes from 1 to n, as $1^3 + 2^3 + \ldots + n^3$.
  - Print this sum.

Below is sample input/output (user input is underlined).

```
Enter total number of sequences: 4
Enter bound: 1
Sum of cubes from 1 to 1 = 1
Enter bound: 2
Sum of cubes from 1 to 2 = 9
Enter bound: 3
Sum of cubes from 1 to 3 = 36
Enter bound: 100
Sum of cubes from 1 to 100 = 25502500
```

You do *not* need to match the example output (above) exactly — it is just to explain the problem better.

Problem 3: **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.

Read the instructions carefully before you start coding! If you get stuck, try to maximize your partial credit. Your program is a diary that prints some statistics on bird watching trips. The program should do the following:

- Prompt the user to enter the total number of days of the trip.
- For each day:
  - Prompt for the location's name, exactly following the example below.
  - For that day, prompt for the total waterfowl, songbirds and parrots seen.
- After the user has entered all of their data:
  - Print the total birds seen across the whole trip.
  - Print the average birds seen per day for the trip.
  - Print the location with the most Songbirds seen on a single day.
  - Print the location with the most birds seen on any single day.
  - If only one parrot was seen for the whole trip, print "Cool, you saw a parrot." If more than one parrot was seen, print "Wow, you saw X parrots!" (where X is the total parrots seen across the whole trip). If no parrots were seen, print "No parrots? Thats sad."

Your program should match the sample output below, *exactly*. User input is underlined.

```
* My Bird Watching Diary *
Enter number of days: 2

Location on Day 1: Golden Gate Park
Total Waterfowl: 5
Total Songbirds: 30
Total Parrots: 5

Location on Day 2: Telegraph Hill
Total Waterfowl: 15
Total Songbirds: 25
Total Parrots: 10

* Summary *
You saw 90 total birds.
Average birds per day: 45.0
Location with most songbirds: Golden Gate Park
Location with most birds: Telegraph Hill
Wow, you saw 15 parrots!
```

Problem 3, continued . . .

Extra Page