PROBLEM 2-1 (5/5 points)

What does the following code print? Assume Pylab's estimation code is perfect - that is, if you calculate that it would print `0.25`, type 0.25 into the box rather than something like 0.24999999999. You may type in strings with or without quotes.

```
a = 1.0
b = 2.0
c = 4.0
yVals = []
xVals = range(-20, 20)
for x in xVals:
    yVals.append(a*x**2 + b*x + c)
yVals = 2*pylab.array(yVals)
xVals = pylab.array(xVals)
try:
    a, b, c, d = pylab.polyfit(xVals, yVals, 3)
    print a, b, c, d
except:
    print 'fell to here'
```

0.0 2.0 4.0 8.0

Check | Save | *You have used 1 of 3 submissions*

PROBLEM 2-2 (1/1 point)

Consider the following sets of measurements and answer the following 3 questions:

A. `[0,1,2,3,4,5,6,7,8]`

B. `[5,10,10,10,15]`

C. `[0,1,2,4,6,8]`

D. `[6,7,11,12,13,15]`

E. `[9,0,0,3,3,3,6,6]`

Select the two lists that have the same mean and variance.

- [ ] A
- [ ] B
- [ ] C
- [ ] D
- [ ] E
- [x] No two sets have the same mean and variance.

You have used 1 of 1 submissions

PROBLEM 2-3 (1/1 point)

Consider following Python functions:

```
def possible_mean(L):
    return sum(L)/len(L)

def possible_variance(L):
    mu = possible_mean(L)
    temp = 0
    for e in L:
        temp += (e-mu)**2
    return temp / len(L)
```

Select the two lists that return the same values when passed into the `possible_variance` function that is defined above.

- [ ] A
- [x] B
- [ ] C
- [x] D
- [ ] E
- [ ] No two sets return the same values.

You have used 1 of 1 submissions

PROBLEM 2-4 (1/1 point)

Is the the answer to Problem 2-2 the same as the answer to Problem 2-3? If not, why are they different?

- ( ) They are the same.
- ( ) They are different because the `possible_mean` function adds up the wrong values.
- (•) They are different because of the way Python 2.7 handles division of integers. ✔
- ( ) They are different because of floating point precision issues.

You have used 1 of 1 submissions

Terms of Service and Honor Code

Privacy Policy (Revised 4/16/2014)