This test is the test from fall 2011, but question 3 is modified to cover topics we discussed. Note that not all topics we discussed are on this test, don't just study this test.

PROBLEM 1 : (*What is the output? (10 points)*)

A. (5 pts) What is the output of the following code segment? Write the output to the right. Note that there is only output for the print statements.

Output:

```
x = 5
y = 3.0
print 2 + x * 4
print x * 4 + 2
print 12 / x
print 10 / y
print x > 2
```

B. (5 pts) What is the output of the following code segment? Write the output to the right. Note that there is only output for the print statements.

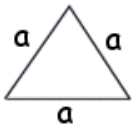
Output:

```
sport = "cross country"
print sport[2]
print sport[2:4]
print sport[sport.find('t'):]

spices = ['basil', 'dill', 'chickory', 'parsley', 'sage']
print spices[2]
print spices[2:4]
```

PROBLEM 2 : (***Triangles and Dinner - Simple Functions (14 points)***)

A. (6 pts) An *equilateral* triangle has all three sides of the same length. The *area* of an equilateral triangle with a the length of a side is $\frac{\sqrt{3}}{4} * a^2$.



Write the function `areaEquilateralTriangle` that has one float parameter `side` representing one side of an equilateral triangle and returns the area of the triangle.

call	**returns**
areaEquilateralTriangle(4.2)	7.63834406138
areaEquilateralTriangle(2.0)	1.73205080757
areaEquilateralTriangle(1.5)	0.974278579257

```
def areaEquilateralTriangle(side):
    '''
    given one float parameter side representing the length of
    one of the sides,
    returns a float that is the area of the equilateral triangle
    '''
```

B. (8 pts) Ellen and Oscar want an easy way to decide who will cook dinner each night. They decided that Oscar will cook if it is an odd day and Ellen will cook if it is an even day. But then Oscar realized that many times he will cook two days in a row, on the 31st and the 1st, but that Ellen would never cook two days in a row. They then agreed in addition that if the day was the 31st day of the month then if the month was even, Ellen would cook and if the month was odd, Oscar would cook that day.

Write the function `whoseNightToCook` that has two int parameters `day` and `month` and returns the string 'Ellen' or 'Oscar', the name of the person who should cook dinner that night following the rules above. Assume the arguments are correct. That is, you do not need to know or verify how many days in a month.

call	**returns**	**comment**
whoseNightToCook(13, 4)	'Oscar'	day is odd
whoseNightToCook(16, 3)	'Ellen'	day is even
whoseNightToCook(31, 8)	'Ellen'	day is 31, month is even
whoseNightToCook(31, 3)	'Oscar'	day is 31, month is odd

```
def whoseNightToCook(day, month):
    '''
    returns "Ellen" if an even day, "Oscar" if an odd day
    except when day is 31, returns "Ellen" if month is even,
        and "Oscar" if month is odd
    '''
```

PROBLEM 3 : (*It's a mystery (12 points)*)

Consider the following `mystery` function with one parameter `animals` which is a list of strings.

```
def mystery(animals):                          # line 1
    ''' animals is a list of strings '''
    x = [ ]                                    # line 2
    for w in animals:                          # line 3
        x += [len(w)]                          # line 4
    amount = max(x)                            # line 5
    y = [ ]                                    # line 6
    for w in animals:                          # line 7
        if len(w) == amount:                   # line 8
            y += [w]                           # line 9
    return y[0]                                # line 10
```

A. (4 pts) Consider making the call `mystery(animals)` with the value of `animals` below. Answer the following questions about tracing what happens with this call

```
animals = ['cat', 'mouse', 'snake', 'chicken', 'fish']
```

A1. What is the value of `x` after line 5 executes?

A2. What is the value of `amount` after line 5 executes?

A3. What is the value of `y` before line 10 executes?

A4. What value is returned from the call `mystery(animals)`?

B. (8 pts) Consider making the call `mystery(zoo)` with the value of `zoo` below. Answer the following questions about tracing what happens with this call

```
zoo = ['lion', 'rhino','bear','zebra']
```

B1. What value is returned from the call `mystery(zoo)`?

B2. Explain in words what `mystery` does.

B3. In the original code, if the if statement in line 8 was changed to if len(w) <= amount: , explain what `mystery` would do now.

B4. In the original code, if line 10 was changed to `return y[-1]`, explain in words what `mystery` would now do.

PROBLEM 4 : (*How many teens? How many boomers? (16 points)*)

A. (8 pts) Write the function `getAges` which has one parameter `data` that is a nonempty list of strings in the format 'firstname:lastname:age' and returns a list of ints of the ages from `data`.

call	**returns**
getAges(['Lisa:Johnson:22', 'Xiao:Xue:34', 'Raj:Reddy:21', 'Bo:Moe:16'])	[22,34,21,16]
getAges(['A:A:8', 'B:B:3', 'C:C:17', 'D:D:42', 'E:E:20'])	[8,3,17,42,20]
getAges(['Barack:Obama:50']	[50]

```
def getAges(data):
```

B. (8 points) Write the function `howManyInRange` which has three parameters, `data` that is a list of strings in the format 'firstname:lastname:age', and two int parameters `start` and `end`. This function returns the number of people in the age range from start to end including the start and end ages. In writing `howManyInRange` you may call `getAges` that you wrote in Part A. Assume `getAges` works correctly.

call	**returns**
howManyInRange(['Lisa:Johnson:22', 'Xiao:Xue:34', 'Raj:Reddy:21', 'Bo:Moe:16'],20,29)	2
howManyInRange(['A:A:8', 'B:B:3', 'C:C:17', 'D:D:42', 'E:E:20'],20,29)	1
howManyInRange(['Barack:Obama:50',30,39)	0

```
def howManyInRange(values, start, end):
```

PROBLEM 5 : (*Talk like a Pirate (14 points)*)

There are three simple rules for talking like a pirate.

1. The word 'Hello' (capitalized or not) becomes 'Ahoy' (always capitalized)
2. 'ar' not starting a word becomes 'arrr' (replace only the first occurrence)
3. For any word of length greater than 7 that does not contain 'ar' inside the word, remove all occurrences of lowercase o's and u's

Write the function `convertWord` that takes a word and returns the pirate equivalent of that word following the rules above.

call	**returns**	comment
convertWord('yesterday')	'yesterday'	no changes
convertWord('boing')	'boing'	word too short, no o replaced
convertWord('are')	'are'	'ar' starts a word, no change
convertWord('gargargantuan')	'garrrgargantuan'	only first 'ar' replaced
convertWord('purposefully')	'prpseflly'	o's and u's removed
convertWord('starboard')	'starrrboard'	no 'o' removed since 'ar' in word

```
def convertWord(word):
```

This page intentially left blank.