Test 1: Compsci 101

Kristin Stephens-Martinez

September 27, 2018

Name: _____ (1/2 pt)

NetID/Login: _____ (1/2 pt)

Section Number: _____

Honor code acknowledgment (signature) _____ (1 pt)

	Value	Grade
Front Page	2 pts.	
Problem 1	30 pts.	
Problem 2	9 pts.	
Problem 3	12 pts.	
Problem 4	18 pts.	
Problem 5	4 pts.	
Problem 6	15 pts.	
TOTAL:	90 pts.	

This test has 12 pages be sure your test has them all. Do NOT spend too much time on one question — remember that this class lasts 75 minutes.

In writing code you do not need to worry about specifying the proper `import statements`. Don't worry about getting function or method names exactly right. Assume that all libraries and packages we've discussed are imported in any code you write.

Be sure your name and net-id are legible on this page and that your net-id appears at the top of every page.

There are two blank pages at the end of the test for extra work-space.

PROBLEM 1 : (*Name/Type/Value (30 points)*)

Part A (26 points)

Consider the following variables and their values in answering the questions below.

```
words = ['potato', 'wasabi', 'oregano', 'jicama', 'onion', 'lettuce', 'sage']
say = 'Hey, diddle, diddle, the cat and the fiddle.'
```

Each variable in the left column is assigned a value. Provide the type and value of the variable after the assignment. The first two lines have been filled in. Types you can use are **int, float, list, string, boolean**.

Assignment	**Type**	**Value**
`a = 7`	int	`7`
`b = words[4]`	string	`'onion'`
`c = say[1] == 'H'`		
`d = len(words)`		
`e = 14 // 4`		
`f = 5 % 2`		
`g = say[:3]`		
`h = 1.0`		
`i = 4 != 4`		
`j = words[0][3]`		
`k = say[-2]`		
`l = 'S' in 'awesome'`		
`m = 3 / 2`		
`n = 'j' in words`		
`o = words[1:2]`		

Part B: What will Python display? (6 points)

Below is a function that returns the largest number in the list. Notice that it has a print statement on line 6. Write the output for the lines of code below. Also, notice both part 1 and part 2 have print statements.

```
1 def max(numbers):
2     ret = 0
3     for n in numbers:
4         if ret < n:
5             ret = n
6             print(n)
7     return ret
```

Part 1

```
m = max([2, 5, 2, 8, 13])
print(m)
```

Part 2

```
print(max([14, 7, 7, 10, 13]))
```

PROBLEM 2 : (*Spot the Bug (9 points)*)

Below is a function similar to the one you saw in lab that returns the fee for a speeding ticket based on the speed. Specifically, the fee should be 60 if the speed is greater than 75, 40 if greater than 50 and less than or equal to 75, 20 if greater than 35 and less than or equal to 50, and zero if less than or equal to 35. But it's buggy! Answer the following questions about this code.

```
1  def speedingTicket(speed):
2      if speed > 35:
3          fee = 20
4      elif speed > 50:
5          fee = 40
6      elif speed > 75:
7          fee = 60
8      else:
9          fee = 0
10     return fee
```

Part A (2 points)

In the first cell below, write a function call for `speedingTicket` that returns a *correct* fee. In the second cell, write your code's return value.

call	return value

Part B (3 points)

In the first cell below, write a function call for `speedingTicket` that returns a *wrong* fee. In the second cell, write your code's return value. In the third cell, write the value it should return.

call	return value	correct value

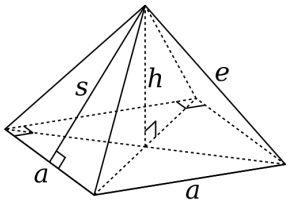
Part C (4 points)

Implement a non-buggy version of the function.

```
def speedingTicket(speed):
```

PROBLEM 3 : (*Formulas (12 points)*)

Part A (6 points)



A square pyramid is a solid made of one square and four triangles where the square is the base with one side of length a and the height of the pyramid is h. See the diagram above.

The slant height of a square pyramid, s in the diagram, in terms of only a and h is:

$$\sqrt{h^2 + \frac{1}{4}a^2}$$

The total surface area of a square pyramid using only a and h is:

$$a(a + \sqrt{a^2 + 4h^2})$$

Implement the following functions that return the pyramids slant height and total surface area. Assume that the math module has been imported and you must use the `math.sqrt` function to calculate square roots (e.g. `math.sqrt(4)` evaluates to 2).

```
import math

def slantHeight(a, h):
    """
    return the slant height of a square pyramid where the parameter a
    has the length of one side of the square and the parameter h has
    the height of the pyramid.
    """
```

```
def totalSurfaceArea(a, h):
    """
    return the total surface area of a square pyramidwhere the
    parameter a has the length of one side of the square and the
    parameter h has the height of the pyramid.
    """
```

Part B (6 points)

At the bazaar, where bargaining is allowed, you want to only pay in prime numbers and you are willing to haggle for it. At the same time, you know that merchants are not necessarily willing to go down in price too far. So instead you decide to either round up or down to the closest prime number when making a purchase. If there is a tie, you will round down.

Assume that you already have the functions `nextPrime(n)` and `previousPrime(n)`, which give you the prime number that is after or before the value in the parameter `n` respectively. For example, `nextPrime(15)` returns 17 and `previousPrime(15)` returns 13.

Implement the function `primeRound(n)` that returns the closest prime number to `n`. You may assume that the parameter `n` is a number greater than or equal to 2. For example:

```
primeRound(15)  # Returns  13, closest primes to 15 are 13 and 17
primeRound(12)  # Returns  11, closest primes to 12 are 11 and 13, so it rounds down
primeRound(155) # Returns 157, closest primes to 155 are 151 and 157

def primeRound(n):
    """
    returns the closest prime number to the parameter n of type int.
    """
```

PROBLEM 4 : (*Word Lengtheners (18 points)*)

Part A (6 points)

There are certain words that have a set of middle letters repeated many times to create a greater emphasis on the word, such as bananananana or whooooooops. Notice that each of these words has a pattern of non-repeated letters on each end of the word and repeated letters in the middle.

Implement the function `wordLengthen(prefix, middle, suffix, num)` as described below that generates words like this. For example:

Function call	Return value
wordLengthen('b', 'an', 'a', 3)	'bananana'
wordLengthen('b', 'o', 'k', 10)	'boooooooook'
wordLengthen('tw', 'e', 't', 4)	'tweeeet'

```
def wordLengthen(prefix, middle, suffix, num):
    """
    The values in the parameters prefix, middle, and suffix are of type string and
    the parameter num has a value of type int.

    return a string that starts with the value in the parameter prefix. Then has the
    value in the parameter middle repeated the number of times in the parameter num.
    And finally the string returned ends with the value in the parameter suffix.
    """
```

Part B (6 points)

Implement the function `wordLengthen2(adder, middle, num)` as described below, using **a call to `wordLengthen` function above**.

```
def wordLengthen2(adder, middle, num):
    """
    adder - a function that takes a bool and returns a string. If adder's parameter
        is True it returns the prefix and if False it returns the suffix.
    middle - a value of type string
    num - a value of type int

    return a string that starts with the prefix returned by adder. Then has the value
    in the parameter middle repeated the number of times in the parameter num. And
    finally the string returned ends with the suffix returned by adder.
    """


    return wordLengthen(_____________________________, _____________________________,


        _____________________________, _____________________________)
```

Part C (3 points)

Write the `adder` function that would make the function call `wordLengthen2(myAdder1, 'o', 4)` return "whoooops".

```
def myAdder1(isPrefix):
```

Part D (3 points)

Write the `adder` function that would make the function call `wordLengthen2(myAdder2, 'na', 3)` return "bananana".

```
def myAdder2(isPrefix):
```

PROBLEM 5 : (*Crazola Crayons (4 points)*)

The Crazola factory needs to order boxes for their crayons. You have been charged to write a function that returns how many boxes they will need based on the number of crayons. A box holds 12 crayons.

Implement the function `boxesNeeded(crayons)` that returns the number of boxes needed for the number of crayons in the parameter of type int `crayons`. For example:

Function call	Return value
boxesNeeded(5)	1
boxesNeeded(13)	2
boxesNeeded(24)	2

```
def boxesNeeded(crayons):
    """
    returns the number of boxes needed to hold the number of crayons specified
    by the int parameter crayons
    """
```

PROBLEM 6 : (*Word Shaving (15 points)*)

The crazying shaving company wants to shave off an equal number of letters off each side of any word until they get to a specific length. They have asked you to write a function that does this for them. Thankfully, they have at least promised you they will never ask you to return an odd number of middle letters for a word with an even number of letters and vice versa.

Implement the function `def middlePart(word, n)` that returns the middle n letters of a word. For example:

Function call	**Return value**
middlePart('care', 0)	''
middlePart('laptop', 2)	'pt'
middlePart('closely', 3)	'ose'
middlePart('car', 3)	'car'

```
def middlePart(word, n):
    """
    The parameter word is of type string and n is of type int.

    returns the middle n letters of the string in the parameter word.

    Assume n >= 0 and (len(word) % 2) == (n % 2).
    """
```

extra page

extra page