Q. 1	[5]	____________
Q. 2	[20]	____________
Q. 3	[23]	____________
Q. 4	[21]	____________
Q. 5	[18]	____________
Q. 6	[13]	____________
TOTAL	[100]	____________

15110 PRINCIPLES OF COMPUTING – EXAM 1A- Summer 2015

Name: _________________________________ **Section:** ______

Andrew Id: _____________________________

Directions: Answer each question neatly in the space provided.
Please read each question carefully. You have 50 minutes for this exam.
No electronic devices allowed. Good luck!

Q.1 History of Computing Devices [5 points]

(a) [1 pt] Although it was never built, the design of Charles Babbage's Analytical Engine is considered to be the foundation of modern computing. Why?

(b) [1 pt] How did the Jacquard's Loom improve the knitting process and make it possible to complete complex weaves at a much faster rate? What did this show people about the relationship between machines and human beings?

(c) [1 pt] Moore's Law says that computers become twice as powerful every 2 years. According to this, 12 years from now, computers will be ________ times as powerful as they are now.

(d) [2 pt] A Terabyte (TB) is 2^{40} Bytes and a Gigabyte (GB) is 2^{30} Bytes. If you have a storage device with a capacity of 4TB and a movie takes 2GB place. How many movies can you fit in that device? (You can express the result as a power of 2, without converting it to decimal).

2. This problem focuses on expressions, data types, and variable assignments. [20 points]

(a) [6 pts] For each of the following Python expressions, write down the value that is output when the expression is evaluated using a python3 interpreter. Write Error if you think the expression will raise an error.

```
              22 // 7  ____________      5 / 2  _____________

  20 + 3 * 4 - 2 ** 3  ____________    18 % 12  _____________

             3 >= 3.1  ____________   "25" * 2  _____________
```

(b) [4 pts] Suppose that we type the following assignments in a Python shell in **the given order**.

```
>>> salary = 4000
>>> rate = 0.1
>>> increase = salary * rate + 100
>>> salary = salary + increase
>>> wageCut = 1 / 3 * salary
```

Write the value that will be output if the expression was evaluated by a Python Interpreter

```
>>> salary                             →
>>> increase                           →
>>> wageCut                            →
>>> (wageCut>1000) and (salary==4000)  →
```

(c) [4 pts] Assume the following list definition in Python.

```
>>> cars = ["Mazda", "Mercedes", "Toyota", "Audi", "Chrysler"]
```

What would be displayed in a Python shell for each of the following expressions if they are evaluated in the given order? If it would give an error then write Error.

```
>>> len(cars)                          →
>>> cars[4]                            →
>>> cars[2] < cars[3]                  →
>>> cars[len(cars) - 4]                →
>>> "Maz" in cars            (+bonus)  →
```

(d) [6 pts] Show how to create lists according to the descriptions below

I) Create a list named *tens* consisting of integers from 10 to 99 inclusive in an increasing order.

II) Create a list named *numbers* consisting of 30 copies of the number 1.

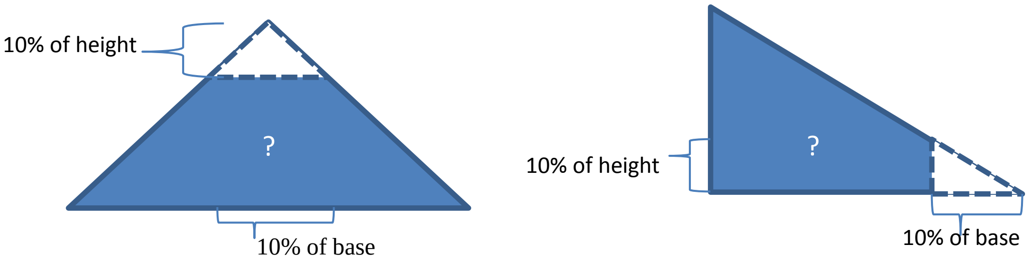
III) Let *lstTeamA* and *lstTeamB* be two non-empty lists. Show how to insert the first element of *lstTeamA* into *lstTeamB* so that it will be 3rd item of *lstTeamB*.

3. This question focuses on the basics of Python functions and tracing. [23 points]

(a) [2 pts] Write a function ***triangle_area*** which takes two parameters (*height* and *base)* to calculate and **return** the area of the triangle. The area of triangle is calculated with the formula below:

$$Area = \frac{height * base}{2}$$

(b) [2 pts] Write a function ***truncated_triangle*** which also takes height and base parameters (h and b) as input, and computes the area of a triangle that the tip cut off. The tip is also a triangle; its height and base are 10% of the height and base, respectively, of the larger triangle, as shown in the figure. **Use the *triangle_area* function in your solution**.



(c) [4 pts] Consider the following Python function where m and n are assumed to be positive integers. Trace this function for m = 2 and n = 5 and show the output of it.

```
def mystery1(m, n):
   i = 0
   while i < n:
      i = i + 1
      print(i ** m, end=" ")
```

*First, call the function **mystery1** for **m = 2 and n = 5***
Second, go to the next line and show the output

>>>

(d)[2 pts] Which of the following is the output of the function ***mystery1*** expressed in terms of m and n? Circle your answer?

a) 0^m 1^m 2^m 3^m 4^m … n^m	e) m^0 m^1 m^2 m^3 m^4 … m^n
b) 0^m 1^m 2^m 3^m 4^m … $(n-1)^m$	f) m^0 m^1 m^2 m^3 m^4 … $m^{(n-1)}$
c) 1^m 2^m 3^m 4^m … n^m	g) m^1 m^2 m^3 m^4 … m^n
d) 1^m 2^m 3^m 4^m … $(n-1)^m$	h) m^1 m^2 m^3 m^4 … $m^{(n-1)}$

(e)[4 pts] If the print statement was taken outside of the while loop to occur right after the while statement, as shown below (mystery2), what would be printed by the call of mystery2 function with parameters **3 and 10**?

```
def mystery2(m, n):
   i = 0
   while i < n:
      i = i + 1
   print(i ** m, end=" ")
```

*First, call the function **mystery2** for **m = 3 and n =10***
Second, go to the next line and show the output

>>>

(f) [3 pts] Consider the ***mystery2*** function above and show the output of the program segment below.

```
mysteryVal = mystery2(2, 3)

if mysteryVal == 9:
   print("I'm here:", mysteryVal)
else:
   print("Here I'm:", mysteryVal)
```

Output

(g) [6 pts] Consider the following recursive function below that computes the sum of a set of pairs of integers.

```
def mySum(a, b):
   if a <= 0:
      return 0
   elif a == b:
      return a + mySum(a-1, b+1)
   else:
      return a + b + mySum (a-1, b+1)
```

Show how the calculation is completed by listing the sequence of function calls that lead to the answer. The first two calls are given for you.

`mySum(4, 4)` → `mySum(   ,   )` → ______________________________

____

____

Write the final values returned from:

`>>> mySum(2, 3)` ______________________

`>>> mySum(4, 4)` ______________________

4. This question focuses on loops and searching. [21 points]

(a) [7 pts] A function is needed to determine whether the numbers in a given list is ordered or not. This function takes a list as an input parameter and, it returns 'True' if the items of the list are sorted in an ascending way. If it is not sorted the function returns False. For example the function will return True for the list [23, 34, 40, 60] but it will return False for the list [23, 34, 33, 60].Complete the missing parts of the function. Do not use built-in functions used for sorting like *sorted*.

```
def isSorted(myList):

    index = _______________

    while index < __________________________ :

        if _____________________________________ :

            return _____________________

        index = _________________________

    return _____________________
```

(b) [2 pts] Give an example of a five element list that would be a worst case input for the isSorted function.

____

(c) [2 pts] What is the big O worst case complexity of **isSorted** function? Indicate what *n* refers to.

____

(d) [2 pts] Show the output generated from each of the following calls to **isSorted** function

>>> isSorted([16, 64, 128, 256, 512]) **Answer:** ______________________

>>> isSorted([64, 128, 512, 510]) **Answer:** ______________________

(e) [3 pts] See the ***isSorted*** function above. How many times would the while loop iterate when we call `isSorted(list(range(10,100)))`?

Answer : ________________

(f) [5 pts] Trace the following ***youNameIt*** recursive function which takes a list as parameter. First, try to understand what it does and show the change in variables and the function calls in the table below for **listA = [2,5,7,6]** and then give an appropriate name to that function which should be related to its aim.

```
def youNameIt(myLst):
    listSize = len(myLst)
    if listSize < 2 :
        return True
    else:
        return (myLst[0] <= myLst[1]) and youNameIt(myLst[1:])
```

youNameIt(myLst)	**listSize**	**myLst[0] <= myLst[1]**	**myLst[1:]**	**Value Returned**
youNameIt ([2, 5, 7, 6])				
youNameIt ( )				
youNameIt ( )				
youNameIt ( )				
youNameIt ( )				

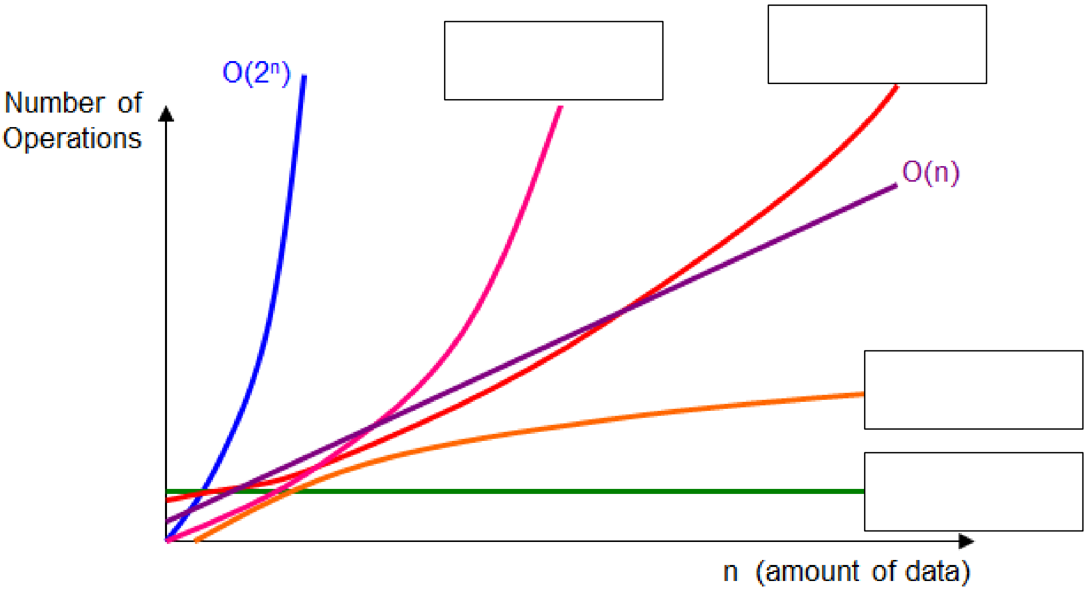
Your suggestion for the function name ____

5. This question deals with searching and sorting. **[18 points]**

(a) [4 pts] What is the big O complexity of insertion sort? ________________

What is the big O complexity of merge sort? ________________

Fill in at least two of the boxes with appropriate formulas



(b) [6 pts] Fill in the table below to show how **binary search** would locate the value “d” in the letters `["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k"]`. Use the binary search algorithm covered in lecture. Note: this table may contain extra rows.

Iteration	Lower	Upper	Mid	letters[mid]

(c) [4 pts] Which algorithm is better for the cases below

Finding an element in an unsorted list	
Finding an element in a sorted list	

(d) [4 points] Suppose you want to look up a word in a dictionary which has 500 pages. Consider only finding the correct page in the dictionary by comparing the word you are looking for to the first word on any given page.

Answer the questions below accordingly:

If you use binary search, how many name comparisons must you perform on the worst case?	
If you to use linear search, how many name comparisons must you perform on the worst case?	

6. This question deals with recursive functions. **[13 points]**

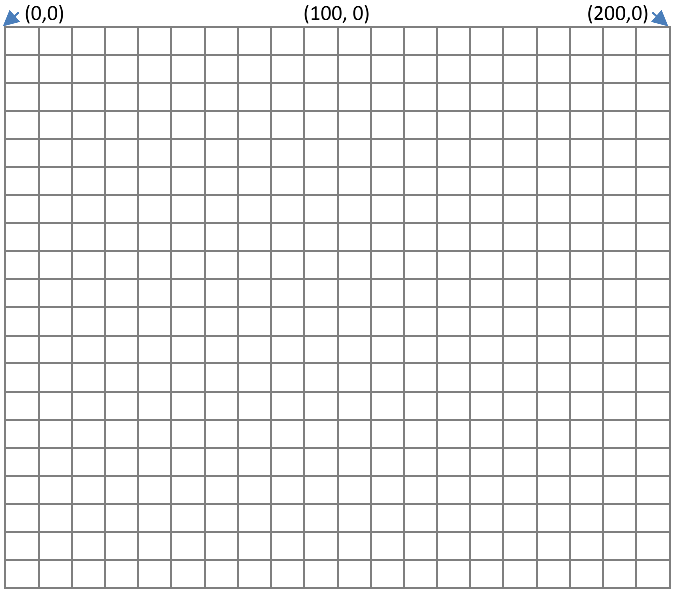
(a) [8 pts] On a computer system, a user can draw a square using the function

```
draw_square(left, top, side_length)
```

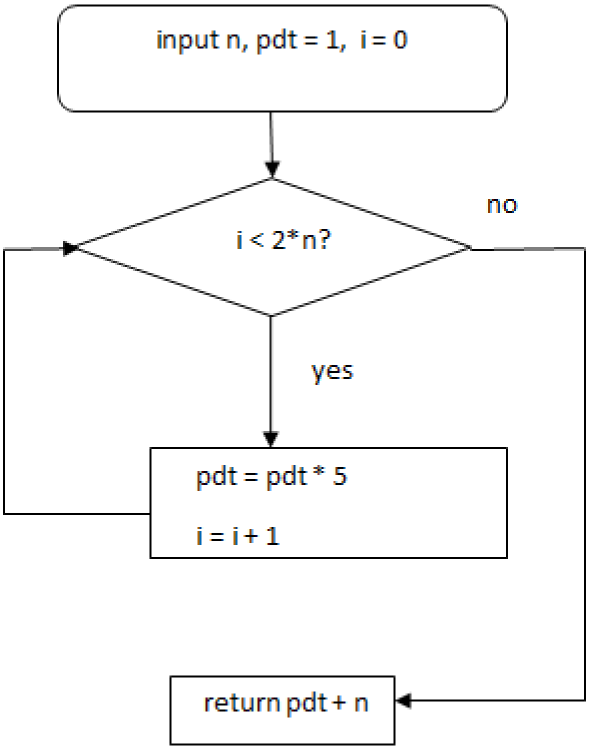
where left, top, and side_length are given in pixels.

In the 200 X 200 window shown below, draw what would be displayed by the following recursive function if it is initially called with x = 10, y = 10, and side = 10. (Grid lines are provided for you to help you draw your picture.)

```
def rec_draw(x, y, side):
   if side <= 100 and x < 100:
      draw_square(x, y, side)
      if side > 20:
         draw_square(x-20, y-20, 20)
      if side > 30:
         draw_square(x-30, y-30, 10)
      rec_draw(x+side, y+side, side*2)
```



(b) [3 pts] Consider the following flow chart. Convert the above flow chart into Python code. Use the following outline. If you can only implement the function in a different way, do it and you can receive up to half credit.



```
def mystery_func(n):

    ______________________________

    ______________________________

    while __________________________:

            ______________________________

            ______________________________

    return __________________________
```

(c) [2 pts] Explain in one sentence the purpose of this code (using n in your answer). A mathematical expression can be your explanation.