15110 PRINCIPLES OF COMPUTING – EXAM 1A- Fall 2014

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!

1	[8]	______________
2	[20]	______________
3	[20]	______________
4	[20]	______________
5	[16]	______________
6	[10]	______________
7	[6]	______________
TOTAL	[100]	__________

1. [8 pts] History of computing devices

1a. [3 pts] Moore's Law says that the number of integrated circuit chips in a computer doubles every 2 years, which implies that computers become twice as powerful every 2 years. According to Moore's Law, 10 years from now, computers will be ________ times as powerful as they are now. Express the answer as a power of 2.

10/2 = 5 doublings
Answer: 2^5

1b. [2pts] Describe one important effect World War II had on computing. Limit your answer to 2 sentences.

World War II gave rise to new computing needs, for example, for computation of artillery tables, keeping track of troop deployment, breaking ciphers. This is in turn lead to increased funding for efforts to build electronic computers.

1c. [3 pts] A Gigabyte (GB) is 2^{30} Bytes and a Kilobyte (KB) is 2^{10} Bytes. If you have a storage device with a capacity of 16 GB, how many 2KB files can you fit in that device? Express the result as a power of 2, without converting it to decimal.

$2^{\text{------------}}$

16 GB = 2^34 Bytes

2^34 / 2^11 = 2^23

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

2a. [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. Recall that // is used for integer division.

```
26 // 5 _____5______                 6 / 4     _____1.5_______

12 + 5 * 4 - 1 ____31________       24 % 3     ______0______

10 != 3 _____True_______            "15110 " * 2 ____"1511015110"________
```

2b. [4 pts] Suppose that we type the following assignments in a Python shell **in the given order.**

```
>>> x = 0
>>> y = 10
>>> z = x + y
>>> x = x + 1
```

For each of the expressions below write down the value that will be output if the expression is evaluated by a Python interpreter after making the assignments above.

```
>>> y
____10______

>>> x
____1_____

>>> z
____10_____

>>> (y < x) or (y < z)
____False_______
```

2c. [6 pts] Assume the following list definition in Python.

```
>>> fruits = ["banana", "orange", "cherry", ["pear", "apple"]]
```

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(fruits)
```

_________4_____________

```
>>> fruits[len(fruits)]
```

_________Error_____________

```
>>> fruits[0] < fruits[1]
```

_________True____________

```
>>> a = fruits[3]
```

nothing is displayed after this step

```
>>> len(a)
```

_________2____________

```
>>> fruits[3][1]
```

______"apple"_________(OK if " is omitted)_______

```
>>> fruits[3] = "kiwi"
```

nothing is displayed after this step

```
>>> fruits[2] + fruits[3]
```

___"cherrykiwi"______(OK if " omitted)_____________

2d. [2 pts] Show how to create a list of every integer between 1 and 15110, inclusive, named `lst1` using Python, sorted in increasing order.

```
lst1 = list(range(1,15111))
```

[2 pts] Let lst2 and lst3 be two non-empty lists. Show how to append the first element of lst2 to the end of lst3 using Python.

lst3.append(lst2[0]) / lst3.extend([lst2[0]]) / lst3 + [lst2[0]] or anything else that is correct gets full credit

3. [20 pts] This question focuses on the basics of Python functions and tracing.

3a. [5 pts] The distance *d* in meters from the ground when an object is dropped from a height *h* in meters after *t* seconds is described by the formula $d = h - \frac{1}{2}(9.8t^2)$.

Write a Python function distance_from_ground(h, t) that has two parameters representing the initial height h of the object in meters and the number of seconds t that an object has fallen from the given height. This function should return the distance from the ground in meters (as a floating point number) for this object given its initial height and the number of seconds that the object has fallen. The returned result should be a floating point number.

```
def distance_from_ground(h, t):
    return (h – 0.5 * 9.8 * (t**2))
```

3b. [5 pts] Consider the following Python function where n is assumed to be a positive integer:

```
def mystery(n, m):
    p = 1
    e = m
    while e > 0:
        p = p * n
        e = e - 1
    return p
```

p	e
1	3
4	**2**
16	**1**
64	**0**
____	____

Trace this function for n = 4, m = 3, showing the value of e and p in the table above at the end of each iteration of the loop. The initial values of p and e are given for you in the table. Use as many spaces as you need.

3c. [3 pts] Which of the following expressions is being computed by `mystery` above? Circle your answer.

nm   n+m   [✓ n^m] m^n nm/m   none of these

All or none

3d. [2 pt] Suppose that the return statement was indented as below. What would `mystery(4, 3)` return in this case?

```
def mystery(n, m):
    p = 1
    e = m
    while e > 0:
        p = p * n
        e = e - 1
        return p
```

4 since return stops the execution of the function.

All or none.

3e. [5 pts] Consider the following recursive function below that computes the sum of the first n positive integers:

```
def sum(n):
    return sum_helper(n, 0)


def sum_helper(n, subtotal):
     if n == 0:
         return subtotal
     else:
         return sum_helper(n-1, subtotal+n)
```

Show how the sum of the first 4 positive integers is computed by listing the sequence of function calls that lead to the answer and write what value is finally is returned. The first two calls are given for you.

`sum(4) --> sum_helper(4,0) -->` **sum_helper(3,4) → sum_helper(2,7) → sum_helper(1,9) → sum_helper(0,10) which returns 10**

4. [20 pts] This question focuses on searching.

4a. [6 pts] Below is a Python function that takes an integer list `lst` and an integer `num` as inputs, and searches for the last number in the list that is greater than or equal to `num`. It returns the index of that number, or None if there is no such number. For example, when the function is called with [100, 45, 12, 24] for `lst` and 40 for `num` it should return 1. This is because 45 is the last item in the list that is greater than or equal to 40. Complete the missing parts of the function.

```
def  last_greater(lst,num):
    last_index = None

    for i in range(0, _ len(lst)_):

        if _ lst[i] >= num _:

          last_index = ____ i ______

    return _ last_index _
```

4b. [4 pts] Write the output from each of the following calls to `last_greater`.

```
>>> last_greater([10, 20, 30, 11, 13], 14)

2


>>> last_greater([], 1)

None
```

4c. [2 pts] How many times would the for loop iterate if we ran `last_greater(list(range(1,100)), 50)`? Answer: _____**99**_______

4d. [5 pts] We could search the list backwards, looking for an integer that is greater than or equal to `num`, and return its index as soon as we find one. Write a Python function called `last_greater_backwd(lst,num)` that outputs the same result as `last_greater(lst,num)` for the same inputs but works as described above.

Answer 1:

```
def  last_greater_backwd(lst,num):

    for i in _ range(0,len(lst))__:

        if  lst[len(lst) - i - 1] >= num:

            return len(lst) - i - 1

    return ____None_____
```

Answer 2:

```
def  last_greater_backwd(lst,num):

    for i in _ range(0,len(lst))__:
        k = len(lst) - i - 1

        if ____ lst[k]_____ >= num:

            return _ k__

    return ____None_____
```

Answer 3:

```
def  last_greater_backwd(lst,num):

    for i in _ range(len(lst)-1, -1, -1)_:

        if ______lst[i] _______ >= num:

            return ______i_________

    return ____None________
```

4e. [3 pts] If the size of the input list is *n*

i. What is the worst-case big O complexity of the function `last_greater`? O(____***n***____)

ii. What is the worst-case big O complexity of the function `last_greater_backwd` ? O(____***n*** ______)

iii. What kind of a list would constitute the **best** case input (out of all possible lists of length *n*) for `last_greater_backwd`? **A list having any number >= the key (`num`) as its last element.**

5. [16 pts] This question deals with searching and sorting.

5a. [6 pts] Consider the behavior of merge sort on the following list of 15 elements

[25, 70, 15, 10, 40, 45, 50, 35, 60, 20, 65, 75, 55, 80, 5]

Complete the table below to show how merge sort would merge lists of increasing size until the entire list is sorted. In the first column, give the size of the largest of the lists being merged at that step– **don't give the step number!** (There may be more rows than you need in the table.)

List size	**Lists**
1	[25] [70] [15] [10] [40] [45] [50] [35] [60] [20] [65] [75] [55] [80] [5]
2	[25, 70] [10, 15] [40, 45] [35, 50] [20, 60] [65, 75] [55, 80] [5]
4	**[10, 15, 25, 70] [35, 40, 45, 50] [20, 60, 65, 75] [5, 55, 80]**
8	**[10, 15, 25, 35, 40, 45, 50, 70] [5, 20, 55, 60, 65, 75, 80]**
	[5, 10, 15, 20, 25, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80]

OR

4 [10, 15, 25, 70] [40, 45] [35, 50] [20, 60] [65, 75] [55, 80] [5]

4 [10, 15, 25, 70] [35, 40, 45, 50] [20, 60] [65, 75] [55, 80] [5]

2 [10, 15, 25, 35, 40, 45, 50, 70] [20, 60] [65, 75] [55, 80] [5]

4 [10, 15, 25, 35, 40, 45, 50, 70] [20, 60, 65, 75] [55, 80] [5]

4 [10, 15, 25, 35, 40, 45, 50, 70] [20, 60, 65, 75] [5, 55, 80]

8 [10, 15, 25, 35, 40, 45, 50, 70] [5, 20, 55, 60, 65, 75, 80]

_ [5, 10, 15, 20, 25, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80]

5b. [5 pts]

What is the big O complexity (in the worst case) of merge sort for a list of *n* elements?
O(_______***n* log *n***_____________)

What is the big O complexity (in the worst case) of insertion sort for a list of *n* elements?
O(________n^2_____________)

5c. [5 pts] As an unpaid intern for muckraker.com, you have obtained a file containing an unsorted list of the names of people on the Federal Government's no-fly list (people who are denied permission to board commercial airlines). You are curious to know whether your own name appears in the list. You could use linear search to look for your name, or you could use merge sort to sort the list, and then use binary search to look for your name. Which is likely to be faster? Explain referring to your knowledge of the big O complexity of the given search and sort methods.

1 point for "linear search". The explanation is worth 4 points and it must give the correct worst-case complexities for each method for full credit.

6. [10 points] **This question deals with correctness of functions and testing.**

6a. [2 pts] Complete the following Python function so that it computes the *integer* base 2 logarithm of its input (i.e., $\lfloor \log_2 n \rfloor$):

```
def log2(n):
    assert(n > 0)
    q = n
    i = 0
    while q > 1:
        q = q // 2

        i = ____i + 1________
    return i
```

6b. [4 pts] Explain in one sentence the purpose of the assert statement.
We want the program to crash if the input makes no sense, i.e. if n is 0 or less.

6c. [4 pts] Below is some code whose purpose is to test the `log2` function.

```
def test_log2():
    for k in range(1, 256):
        assert(2**log2(k) <= k)
```

Assuming the `log2` function is correct, will the `test_log2` function run without reporting any errors? Explain.

Yes, because the assert statement only reports an error if the condition provided is false. If the function is correct, it returns the integer log of *k*. Raising 2 to any power less than or equal to the log of *k* produces a number less than or equal to *k*, and the integer log of *k* is less than or equal to the log of *k*.

7. [6 pts] This question is based on your readings from the book *Blown to Bits*.

7a. [3 pts] If one person is sick on day 0, and each subsequent day the number of sick people doubles, on which day will there be approximately 1 million people sick?

___**20**____

7b.[3 pts] The book Blown to Bits describes why we knowingly give up our privacy in certain ways. Name three of the kinds of reasons that is discussed in the book for giving up our privacy (not specific instances).

Any three of

- **saving time**
- **saving money**
- **convenience**
- **fun**
- **can't live any other way**