CSCI 1100 Exam #3

Aaron Taylor

TOTAL POINTS

88 / 100

QUESTION 1

1 Question 1 12 / 13

- 0 Correct
- 3 Does not define function correctly (def, function name, 4 arguments)
- 1 Does not open file correctly (or at all)
- 2 Does not loop over file correctly
- 2 Does not keep track of line number correctly
- 1 Does not return None when line number is less than file length
- 2 Does not return empty string when line has fewer than end characters
- 2 Does not return the specified string correctly
- **1 Leaves off last character of string or adds an extra character**
- 1 syntax error

QUESTION 2

2 Question 2 12 / 13

- 0 Correct
- 3 Does not define function correctly (def, function name, 1 argument)
- 2 Modifies the input list (does not create a copy or new set)
- 2 Does not iterate through the list correctly (or at all)
- 3 Does not find a repeated number at all
- **1 Does not find the first repeated number**
- 1 Does not return None if there are no repeated numbers
- 1 Does not return the value (prints instead)
- 1 Syntax error, or similar interpreter error
- 1 Does not return a single integer (instead returns str, list, tuple, or set)
- 2 Modifies the input list by sorting

QUESTION 3

3 Question 3 15 / 15

- **0 Correct**
- 3 Does not loop over actors correctly
- 3 Does not compare actor's set to the best5 set correctly (using intersection or checking with if/in)
- 3 Does not print histogram correctly for each actor
- 2 Does not sort the names correctly
- 2 Does not right-justify the names correctly
- 2 Length of name column is not equal to the longest name or does not find the max length correctly or hardcodes the value

QUESTION 4

4 Question 4 15 / 15

- **0 Correct**
- 3 Does not loop over keys correctly
- 2 Does not read values correctly
- 4 Does not find top 3 values correctly
- 1 Finds highest values instead of lowest
- 2 Does not format values to 1 decimal place
- 3 Does not print year and value pairs
- 2 More than three items found

QUESTION 5

5 Question 5 13 / 16

- 0 Correct
- 0.5 Does not run simulation for multiple time steps
- 1 Does not run exactly 10 time steps of simulation
- 1 Does not run simulation for every bird
- 0.5 Does not ensure that only active birds moved
- 0.5 Prints multiple stop messages for same bird (e.g., prints new stop message every time step after bird moves out of bounds)
- **1 Does not use move or __str__ methods, from Bird class, wherever possible**
- 1 Does not move bird to correct location (i.e. values

for x and y of Bird class are not changed correctly)

- **0.5** Does not perform bounds check and related tasks after every move (including move during last time step)
- **1 Does not perform bounds check correctly**
- **1 Does not set active flag of bird to False after bounds check fails**
- **1** Does not print stop message after bounds check fails
- **1** Does not print correct time step or position in stop message
- **1** Does not print "Remaining active birds" line or individually prints it for every remaining active bird
- **0.5** Does not ensure that information of only the active birds after final time step is printed under "Remaining active birds"
- **1** Does not correctly print information of each bird under "Remaining active birds"
- **4** Not a decent attempt at answering the question
- **1** Has significant syntax error or missing/misplaced syntax not taken into account by above rubric items

QUESTION 6

6 Question 6 13 / 15

- **0** Correct
- **2** Does not open the file correctly
- **2** Does not loop over the file correctly
- **2** Does not parse top level of line correctly ('|' char)
- **1 Does not parse second level of line correctly (','-separated)**
- **2** Does not create an Agent object for the parsed data
- **1** Does not keep all agents in a list
- **1** Does not use raw_input to ask user for a language
- **1** Does not loop over agents correctly
- **1** Does not find agents who speak the language correctly
- **1** Does not print agent name correctly
- **1** Does not print number of skills correctly
- **1 Syntax error**
- **1** Extra prints

QUESTION 7

7 Question 7 8 / 13

- **0** Correct
- **3 Does not find intersection of sets correctly**
- **2** Does not subtract the sets correctly (find differences)
- **3** Does not union the differences together correctly to find all items in exactly one meal.
- **3** (^)XORs or uses symmetric_difference on all sets for items in exactly one meal & produces slightly incorrect result Or other slightly incorrect result.
- **2 Does not sort the items in both cases or error in sorting**
- **2** Print - Syntax or other error or large difference with given format in printing in both lines Examples - no delimiter between items, the word 'set' is printed
- **1** Does not print according to format or other minor error.
- **1** Syntax error or other error or missing code
- **1** Does not sort the items or error in sorting in one case

li = #some_list
li.sort() --> NoneType

E

Computer Science 1 — CSci 1100
Exam 3
November 23, 2015

RCS ID: tayloa5 @rpi.edu **Name:** Aaron Taylor

RIN # : 661537404

Circle your lab section

Sec. 1	T 10 (Low 3112), Jeramey
Sec. 2	T 10 (Walker 5113), Rahul
Sec. 3	T 12 (Sage 2715), Rahul
Sec. 4	W 10 (Sage 3101), Jeramey
Sec. 5	W 12 (J-Rowl 2C30), Dean
Sec. 6	W 12 (J-Rowl 2C06), Naveen
Sec. 7	W 12 (Eaton 215), Rhaad
Sec. 8	W 2 (Eaton 215), Rhaad

[✓ Sec. 9]	W 2 (J-Rowl 2C14), Naveen
Sec. 10	W 2 (J-Rowl 2C22), Young
Sec. 12	W 2 (Sage 2704), Jassiem
Sec. 13	W 4 (Eaton 215), Jassiem
Sec. 14	W 4 (Lally 104), Young
Sec. 15	W 6 (Lally 102), Partha
Sec. 16	T 12 (Darrin 235), Partha

Problem	Points	Score
1	13	
2	13	
3	15	
4	15	
5	16	
6	15	
7	13	
Total	100	

Instructions:

- You have 90 minutes to complete this test.
- You may use only one double-sided crib sheet. Otherwise, put away all books, laptop computers, and electronic devices.
- Please write using a dark pencil and write legibly. Exams are being scanned and we need your cooperation.
- Please read each question carefully several times before beginning to work.
- In all the questions, your output must match exactly the given output (do not insert additional spaces if they are not in the output).
- We generally will not answer questions except when there is a glaring mistake or ambiguity in the statement of a question.
- Unless otherwise stated, you may use any technique we have covered thus far in the semester to solve any problem.
- Please state clearly any assumptions that you have to make in interpreting a question.
- In all questions, assume the user enters a valid input unless otherwise stated.
- There are **7 questions** on this test totaling 100 points.
- When you are finished with this test please turn it in to the proctor and show him or her your student id. You will then be free to leave the exam room.

Write your answers in the box below only. Do not write on the back or outside the box.

1. (**13 points**) Write a function called `get_line(fname,lno,start,end)` that takes as input a file name `fname`, a line number `lno`, and starting and end points (`start,end`) on the given line. The function should return the string containing all the characters from the starting point up to but not including the end point on that line (same as it would be with string slicing!).

   Line numbers start at 1; characters in a line are counted starting with zero and include the new line character at the end. If there are fewer than `lno` lines, your function should return None. If the line at `lno` has fewer than end characters, return an empty string.

   Given the following contents of file `hpss.txt`:

```
Nearly ten years had passed since the Dursleys had
woken up to find their nephew on the front step.

Privet Drive had hardly changed at
all.
```

   The following program:

```
print '1:', get_line('hpss.txt', 2, 9, 15)
print '2:', get_line('hpss.txt', 5, 5, 9)
print '3:', get_line('hpss.txt', 5, 0, 5)
print '4:', get_line('hpss.txt', 8, 0, 10)
```

   should print (notice for 3, the newline is also included in the returned string):

```
1: to fin
2:
3: all.

4: None
```

```
def get_line(fname,lno,start,end):
    f= open(fname)
    lines=[]
    for line in f:
        lines.append(line)
    if len(lines)<lno:
        return None
    else:
        if len(lines[lno-1])<end:
            return ''
        else:
            return lines[lno-1][start:end-1]
```

Write your answers in the box below only. Do not write on the back or outside the box.

2. (**13 points**) Write a function `find_repeated(L)` that returns the first repeated value in a list of integers `L`. Your function must not change the list `L`. If there is no repeated value, your function should return None. (Hint: Keep track of the set of values seen so far!)

   For example, the following program:

```
print '1:', find_repeated([1,3,4,3,5,6,1])
print '2:', find_repeated([1,3,4,5,6])
print '3:', find_repeated([1,1,3,4,5,6])
```

   should print:

```
1: 3
2: None
3: 1
```

```
def find-repeated (L):
    nums = L[:]
    repeated = [ ]
    for i in nums:
        if L.count(i) > 0:
            rep = i, L.count(i)
            repeated.append(rep)
    if len(repeated) == 0:
        return None
    else:
        return repeated[0][0]
```

3. (**15 points**) Suppose you are given a set containing the best five movies of 2015 according to experts in a variable called `best5` and a dictionary `actors` containing the names of actors as keys and a set of movies for each actor as values.

   Assuming that both variables are defined in your program, print the name of each actor in sorted order, and stars to represent the number of movies for the actor that are in the `best5` set. Right justify the names by the length of the longest name (Hint: remember `"abc".rjust(5)` will return `"  abc"`).

   For example given:

```
best5 = set([ 'Mad Max', 'Inside Out', 'Selma', 'Interstellar', 'The Martian'])
actors = { 'Tom Hardy': set(['Mad Max', 'Legend', 'London Road']),
           'Charlize Theron': set(['Mad Max', 'Dark Places']),
           'Russell Crowe' : set(['Fathers and Daughters', 'Noah']),
           'David Oyelowo': set(['Selma', 'Interstellar']),
           'Matt Damon': set(['The Martian', 'Interstellar']),
           'Jessica Chastain': set(['The Martian', 'Interstellar']) }
```

   Your program should output:

```
 Charlize Theron: *
   David Oyelowo: **
Jessica Chastain: **
      Matt Damon: **
   Russell Crowe:
       Tom Hardy: *
```

```
names = [ ]
for actor in actors',
    for movie in actors[actor]!
        common = best5 & movie
        names.append(actor, len(common))
names = names.sort()
long = 0
for name in names!
    if len(name[0]) > long'.
        long = len(name[0]
for name in name!
    print name[0].rjust(long - len(nam[0]), ":", * * name[1]
```

4. **(15 points)** Suppose you are given the mean temperature for Troy in the month of December in a dictionary `temp` as shown below. The keys of the dictionary are years and the values are the mean temperature for that year. Write a piece of code that finds and prints the top three coldest years according to this dictionary. Note: If there are ties in values, any ordering of years is acceptable.

   For example, given the dictionary below:

```
temp = { 2001: 36.4, 2002: 27.4, 2003: 29.3, 2004: 28.6, 2005: 27.8,
         2006: 37.3, 2007: 28.1, 2008: 30.2, 2010: 26.0, 2011: 35.4,
         2012: 33.8, 2013: 27.9, 2014: 32.8}
```

   Your program should output:

```
2010: 26.0
2002: 27.4
2005: 27.8
```

```
cold=[ ]
for year in temp:
    for temps in year:
        temptup= temps,year
        cold.append(temptup)
cold = set(cold)
cold = list(cold)
cold = cold.sort()

for i in range(3):
    if cold[i][1] == cold[i-1][1]:
        continue
    else:
        print str(cold[i][1])+":", cold[i][0]
```

5. (**16 points**) Suppose you are writing a simulation in which multiple birds move in a board. The board has dimensions `(0,0)-(100,100)` with `(0,0)` representing the upper left corner and `(100,100)` representing the lower right corner.

   The birds are represented as a single `x,y` point (no length or radius). They move with a single increment added to their `x,y` coordinates. We do not worry about collisions between them in this question.

   Write a simulation that goes for exactly 10 steps. At each step, each active bird is moved once. If any bird goes outside the board or is touching one of the edges of the board, it is stopped (the active flag is set to False) and a message is printed. At the end of the simulation, the location of all the active birds is printed.

   You must implement the simulation using the `Bird` class and its methods, and the list variable called `birds` initialized as shown below.

```
class Bird(object):
    def __init__(self, name, x, y, inc):
        self.name = name
        self.x = x
        self.y = y
        self.inc = inc
        self.active = True
    def __str__(self):
        return '%s: (%d,%d)' %(self.name, self.x, self.y)
    def move(self):
        self.x += self.inc
        self.y += self.inc

if __name__ == "__main__":
    birds = [Bird('Wolf',10,30,-10), Bird('Tweety', 40,25,5), Bird('Fluff',95,95,-15)]
    ## add the simulation code that will go here in the box below
```

   When executed on the above list of birds, you will get the following output:

```
Time 1: Stopped: Wolf: (0,20)
Time 7: Stopped: Fluff: (-10,-10)
Time 10: Remaining active birds
Tweety: (90,75)
```

   Please write your solution on the next page, you can use the space on this page for scratch work.

```
birdobjs = []
for bird in birds:
    bird = Bird(bird)
    birdobjs.append(bird)
for i in range(10):
    for bird in birdobj:
        bird.move()
        if (bird.x, bird.y) == (0,0) or ((bird.x, bird.y) == (100,100))
            / (bird.x <0) or (bird.y <0) or (bird.x >100) or
          / (bird.y > 100):
            print "Time: %s stopped: %s: (%s, %s)
                / %(i+1, bird.name, bird.x, bird.y)
            bird.remove(bird)
    if i == 9:
        print "Time %s: remaining active birds" /
              %(i+1)
        for bird in birdobj:
            print "%s: (%s, %s)". %(bird.name, bird.x

                    / , bird.y)
```

6. (**15 points**) Suppose you are given a file named `fieldagents.txt`. Each line of this file contains the name of a Secret Intelligence Service field agent followed by their special skills and languages that they speak. The three fields are separated by `'|'` and individual languages and skils are separated by a comma.

   Write a program that reads this file and creates a list of `Agent` objects using the `Agent` class definition below. Your program then should ask the user for a language and it should print the name and the number of skills of all the agents who speak this language using this list. If no agent speaks the searched language, then nothing is displayed.

   You do not need to add any methods to the class in your solution.

```
class Agent(object):
    def __init__(self, n, s, l):
        self.name = n
        self.skills = s
        self.languages = l
```

   Given the following `fieldagents.txt` file:

```
Sterling Archer|Asset acquisition,Undercover,Krav Maga,Bullet Counting|English,Russian,French
Lana Kane|Counter Sniper,Hand-to-hand combat|English,French
Ray Gillette|Bomb Specialist,Firearms,Bionic legs|English,German,Latin
```

   The following is a possible output of your program (agents are output in the order they appear in the file):

```
Enter a language => French
Agents speaking French:
Sterling Archer (4 skills)
Lana Kane (2 skills)
```

   A second possible output:

```
Enter a language => Turkish
```

   Please write your solution on the next page, you can use the space on this page for scratch work.

Write your answer to Question 6 in the box below only.

```
f= open(fieldagents.txt)
agents= [ ]
for line in f:
        agent = {}
        skills = []
        nline = line.strip().split("|")

            agent["n..."] = nline[0]
            agent["s"] = nline[s]
            agent["L"] = nline[L)
            agents.append(agent)
agentobjs=[]
input= raw_input("Enter a language =>  ")
for agent in agents:
        a = Agent(agent["n"],agent["s"],agent["L"]
        agentobjs.append(a)
print "agents speaking", input
for agent in agentobjs:
    if input in agent.languages:
            print "%s (%s skills)" %(agent.name,
                len(agent.skills))
```

Write your answers in the box below only. Do not write on the back or outside the box.

7. (**13 points**) Suppose you are given three variables `t1,t2,t3` in your program. Each variable is a set containing the menu for a different Thanksgiving dinner you are invited to.

   First, print items that are in the menu for all three dinners. Then, print the items that are in the menu for exactly one dinner. All items should be listed in alphabetical order.

   For example, if you are given menus:

```
t1 = set(['Turkey', 'Potatoes', 'Green Beans', 'Cranberry', 'Gravy'])
t2 = set(['Turkey', 'Yams', 'Stuffing', 'Cranberry', 'Marshmallows'])
t3 = set(['Turkey', 'Gravy', 'Yams', 'Green Beans', 'Cranberry', 'Turducken'])
```

   your program must print the following (your output should match ours):

```
Items in all three dinners: Cranberry, Turkey
Items in exactly one dinner: Marshmallows, Potatoes, Stuffing, Turducken
```

```
three = list((t1 & t2) | (t2 & t3) | (t1 & t3))
one = list((t1-(t2|t3)) | (t2-(t1|t3)) | (t3-(t1|t2)))
three = three.sort()
one = one.sort()
print "Items in all three dinners:",
for i in three:
    print i+", ",
    if i == three[-1]:
        print i, "\n"
print "Items in ~~exactly~~ one dinner:",
for i in one:
    print i + ", ",
    if i == one[-1]:
        print i, "/n"
```