CSCI 1100 Exam #2 Correct

Aaron Taylor

TOTAL POINTS

63.5 / 102

QUESTION 1

1 Question 1 6.5 / 12

- 0 Correct
- 1 Does not read input from user with raw_input
- 1 Does not convert inputs to integer
- **4 Does not add the days for past months correctly (bad loop or incorrect slicing)**
- 1 Loop ending or subsetting for the months list is off by 1 or 2
- **1 Does not add 1 for leap year correctly (incorrect or no mod) or other error**
- 1 If adding 1 for leap year or not, does not check for current month (should be after February) or checks incorrectly.
- **0.5 The date in the final print statement is incorrect or absent. It should be "1/1/user_input_year"**
- 2 Print statement is absent or throws syntax error or other error.

> List subset is created, but not summed over - syntax error on trying to add list subset to an integer
>
> Even if that worked
> This code adds the remaining days of the year, not the required days.

QUESTION 2

2 Question 2 11 / 13

- 0 Correct
- 1 Incorrect function definition
- 1 Does not return result (prints or no return)
- 1 Does not handle correct spelling (may add other penalty on top)
- **2 Does not handle swapping of last two vowels**
- 2 Does not handle the length not being equal to 10 characters
- 2 Does not index or slice the three parts of the string correctly
- 2 Does not add penalties for the last three slices
- 2 If statements are incorrect (not checking for equality)
- 2 Syntax error
- 2 Penalty summed incorrectly
- 1 Extra condition
- 1 Using global variables
- 1 Incorrect indentation

> When checking the last two vowels the rest of the words needs to be correct as well.

QUESTION 3

3 Question 3 3 / 13

- 0 Correct
- 1 Incorrect function definition
- 2 Does not return result (prints or no return)
- 1 Function does not return value on some inputs
- **1 String is not initialized correctly**
- 2 String append is not correctly formatted
- 3 If statements are incorrect
- **3 For loop over the lists is incorrect (indexing will cause error)**
- 3 For loop over the lists is incorrect (NO indexing)
- 3 For loop over the lists is incorrect (uses a double loop)
- **3 Arbitrary syntax error, or similar interpretor error**
- **3 Output string is incorrect for other reasons**
- 3 Infinite loop
- 2 No output at all
- 5 Does not write a function at all
- 3 Uses global variable
- 3 No loop used

- 3 For loop over the lists is incorrect (incorrect indexing)
- 3 Uses multiple functions

QUESTION 4

4 Question 4 5 / 13

- 0 Correct
- 2 Incorrect function definition
- 1 Does not return result (prints or there is no return)
- **2 Length is checked incorrectly (only -1 if check is invalid for a single length)**
- **2 Capitalization is checked incorrectly (-1 for declaring that a caps letter has been found after checking word[i] == word[i].upper(), because this check passes for any non-alphabetic word[i])**
- **2 Punctuation character is checked incorrectly**
- **2 Combines checks incorrectly, or is missing either a decent capitalization or punctuation check (-1 per missing check)**
- 2 Absence/misplacement of return statements or of variable used to maintain return value

#4: should be 'if len(word) < 16'

QUESTION 5

5 Question 5 11 / 13

- 0 Correct
- 1 Does not use raw_input correctly
- 1 Does not print "People born in" line in correctly
- 3 Does not loop through the list correctly
- 2 Does not unpack the tuple correctly
- 2 Does not check for the month correctly (split, in, endswith, find, or count)
- **2 Does not print the ordinal value correctly (e.g. 22nd)**
- 2 Does not format the rest of the print line correctly
- 1 Uses "split" incorrectly (does not store result in a variable or tries to use split off a list, not a string)

QUESTION 6

6 Question 6 12 / 13

- 0 Correct
- 1 Does not use raw_input
- 1 Input not converted to integer
- 1 File not opened correctly (or not at all)
- 2 Loop over file is incorrect
- **1 Does not convert values from file to integer**
- 2 If statement is not correct
- 3 Found values are not correctly averaged (sums and/or count are incorrect)
- 2 Found values are not correctly appended into a list (if using a list)
- 1 The list for found values is not initialized correctly (if using a list)
- 2 Output is not formatted correctly

QUESTION 7

7 Question 7 9 / 13

- 0 Correct
- 1 Does not read value with raw_input
- 1 Converts the value to integer but checks against strings
- 4 Double for loop is incorrect
- **2 If statement is not correct**
- 1 List is not initialized
- 2 List append is called incorrectly
- 1 Doesn't construct list of tuples
- 2 Does not print correctly
- 0 Global variable usage
- 2 Function Fails to Return Value
- 2 Syntax Error
- **2 Semantic Error**
- 13 Blank

i, j are not indices

QUESTION 8

8 Question 8 6 / 12

- 0 Correct
- 1 Part A - First line incorrect (str)
- 1 Part A - Second line incorrect (s)
- **1 Part A - Third line incorrect (7)**
- 1 Part A - Fourth line incorrect (first 3)
- **1 Part A - Fifth line incorrect (second 3)**

- 1 Part A - Sixth line incorrect (list)
- 1 **Part B - Has additional 11**
- 1 Part B - Has additional 1
- 1 Part B - Missing 4
- 1 **Part B - Missing 5**
- 1 **Part B - Missing 7**
- 1 **Part B - Missing 8**
- 1 Part B - Has additional 3
- 1 Part B - Prints all numbers
- 1 Part B - prints only 4
- 2 prints random numbers

  part b: we are comparing strings here

B

Computer Science 1 — CSci 1100
Exam 2
October 26, 2015

RCS ID: taylor5 @rpi.edu **Name:** Aaron Taylor

RIN # : 661637404

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	12	
2	13	
3	13	
4	13	
5	13	
6	13	
7	13	
8	12	
Total	102	

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 **8 questions** on this test totaling 102 points (2 points is bonus, we will truncate grades higher than 100 to 100).
- 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. (**12 points**) Assume that you are given the number of days in each month in a variable called `months` already defined in your program as shown below.

```
months = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
```

Write a program that asks the user for a date using `raw_input` and prints the total number of days between the beginning of the same year and the given date (January first would be zero days).

Note that in leap years, February is 29 days long (not 28 as given above). It is pretty complex to compute if a given year is a leap year. For the purpose of this exam, we will assume that if a year is divisible by 4, then it is a leap year. Make sure your program can handle this case as well.

Here is a possible execution of your program (2016 is a leap year!):

```
Day => 2
Month => 3
Year => 2016
61 days since 1/1/2016
```

```
day = int(raw_input("Day => "))
month = int(raw_input("Month => "))
year = int(raw_input("Year => "))

if months[month] % 4 == 0
    months[1] = 29
  days = months[month:12] + day
else:
    days = months[month:12] + day
print "%d days since %d/%d/%d" (days, month, day, year)
```

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

2. (**13 points**) Write a function `spell_penalty(word)` that will return an integer score representing how badly you misspelled `Rensselaer`. We will use a very simple scoring function (hint: loops are not needed to solve this, only string slicing and if statements, but you can use loops if you wish):
   - Correct spelling: 0 penalty
   - Only swapped the last two vowels, i.e., `Rensselear`: 2 points penalty
   - Word is not exactly 10 characters: 8 points penalty
   - If the above two cases fail, divide the word into three substrings: first four letters, followed by the next three letters and the final three letters. For example, `Rensselaer` will be divided to `Rens|sel|aer` and `Renssealer` will be divded to `Rens|sea|ler`.

     For each part of the input word that is different than the counterparts with the correct spelling, including the capitalization, (i.e. `Rens`,`sel`,`aer`), give a 3 point penalty (regardless of how bad the spelling is).

Here are example runs of the function:

```
>>> spell_penalty('Rensselaer')
0
>>> spell_penalty('Renssealer')
6
>>> spell_penalty('Rensselear')
2
>>> spell_penalty('Rensselrrr')
3
```

```
def spell_penalty(word):
    a = False
    penalty = 0
    if len(word) != 10
        penalty += 10
        a = True
    elif word[7:9] != "ae"
        penalty += 2
        a = True
    elif a = False:
        p1 = word[0:4]
        p2 = word[4:7]
        p3 = word[7:10]
        if p1 != "Rens"
            penalty += 3
        elif p2 != "sel"
            penalty += 3
        elif p3 != "aer"
            penalty += 3
    return penalty
```

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

3. (**13 points**) Write a function `compare_lists(L1,L2)` which takes in two lists `L1,L2` of the same length as parameters and returns a string containing a character for each position in list `L1`. The output string has a `1` in a given index `i` when `L1[i]>L2[i]`, `2` when `L1[i]<L2[i]`, and `~` otherwise.

   Here are example runs of the function:

```
>>> compare_lists([1,2,3],[4,5,6])
'222'
>>> compare_lists([4,5,6],[2,5,8])
'1~2'
>>> compare_lists([2,4,7,3,4,9,7],[3,2,9,-1,-2,9,9])
'21211~2'
```

```
def compare_lists(L1, L2)
    output = []
    for i in L1:
        if L1[i] > L2[i]:
            output.append(1)
        elif L1[i] < L2[i]:
            output.append(2)
        else:
            output.append(~)
    output.join()
    Finaloutput = str(output)
    return Finaloutput
```

4. (**13 points**) Write a function called `good_password` that takes as input a word and returns `True` if the word contains a good password, and `False` otherwise.

   A good password should have at least 16 characters. It should contain at least one upper case letter, and one character from the list `[',','.',':',';']`. The remaining characters can be anything.

   Here are example runs of the function:

```
>>> good_password('gazorpazorp.Field')
True
>>> good_password('gazor:pazorp.Field')
True
>>> good_password('gazorpazorp.field')
False
>>> good_password('gazorpazorpField')
False
```

```
def good_password(word):
    a = True
    If len(word) != 16:
        a = False
    i = 0
    while i <= len(word):
        if word[i] is word.uppercase() == False
            a = False
        elif word.count(list[i]) == 0:
            a = False
    else:
        a = True

    return a
```

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

5. (**13 points**) Suppose your program already has a list called `records` containing names and birthdays of different people.

   Write a program that reads the name of a month using `raw_input` and prints the names of all the people born on that month and the day they were born, in the same order as they appear in the list.

   Given the following list,

```
records = [ ('Snow White', '22nd of May'), ('Prince Charming', '18th of December'),
            ('Red Riding Hood', '20th of August'), ('Evil Queen', '15th of May'),
            ('Huntsman', '1st of May'), ('Jiminy Cricket', '12th of February') ]
```

   Here is an example run of your program:

```
Month => May
People born in May
Snow White, on the 22nd
Evil Queen, on the 15th
Huntsman, on the 1st
```

```
month = raw_input("Month => ")
print "People born in", month
for i in records:
    if month in i[1]:
        print i[0]+", on the", i[1][0:4]
```

6. (**13 points**) Suppose you are given a file named `temp.txt` that contains nothing but integers on each line. Write a program that reads from the user a low and high value, then finds all values in the file between these two values (including the low and the high value), and prints their average.

   For example, given the contents of the file `temp.txt` below:

```
1
4
2
8
16
6
```

   The output of the program will look as follows:

```
Low => 3
High => 8
Average of values between 3 and 8 => 6.0
```

```
l=int(raw_input("low => "))
h=int(raw_input("high => "))
f=open("temp.txt")
lines=f.read()
temps=[]
for i in lines:
    if l<i<h:
        temps.append(i)
mean = sum(temps)/len(temps)
mean = float(mean)
print "Average of values between %d and %d => %d"
    %(l,h,mean)
```

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 a 9 by 9 Sudoku board represented as a list of lists stored in a variable named `board` as shown below.

   Write a program that reads a value using `raw_input`, constructs and prints a list of all locations in the board that contain the input value as tuples.

   Given the following board:

```
board = [ [ '1', '.', '.', '.', '2', '.', '.', '3', '7'],
          [ '.', '6', '.', '.', '.', '5', '1', '4', '.'],
          [ '.', '5', '.', '.', '.', '.', '.', '2', '9'],
          [ '.', '.', '.', '9', '.', '.', '4', '.', '.'],
          [ '.', '.', '4', '1', '.', '3', '7', '.', '.'],
          [ '.', '.', '1', '.', '.', '4', '.', '.', '.'],
          [ '4', '3', '.', '.', '.', '.', '.', '1', '.'],
          [ '.', '1', '7', '5', '.', '.', '.', '8', '.'],
          [ '2', '8', '.', '.', '4', '.', '.', '.', '6'] ]
```

   Your program should work as follows:

```
Enter a value => 2
Locations with 2: [(0, 4), (2, 7), (8, 0)]
```

```
value = raw_input ("Enter a value => ")
coordinates = []
for i in board:
    for j in board:
        if value in board:
            tup = i, j
            coordinates.append(tup)
print "Locations with", value+":", coordinates
```

8. (**12 points total; 6 points each**) Assume each of the following is an individual program. Write the **exact** output of these programs in the area provided. Show your work at each step in the scratch area provided for partial credit. Note: there are no syntax errors in the code provided in this question.

Part a

```
bag = [ ['int', 'float', 'str', 'object'],
        ['class', 'function', 'variable'],
        ['parameter', 'argument'],
        [],
        ['raw_input', 'len', 'range'],
        ['for', 'while', 'if', 'elif', 'else'],
        [] ]
print bag[0][2]
print bag[1][0][3]
print len(bag)
print len(bag[4])
print len(bag[5][0])
print sorted(bag[0])
```

Output:

str
3
6
3
0
['float', 'int', 'object', 'str']

Scratch area:

Part b

```
vals = "4,3,5,1,11,7,8".split(",")
lastval = vals[0]
print lastval
for i in range(1,len(vals)):
    if lastval > vals[i]:
        continue
    print vals[i]
    lastval = vals[i]
```

Output:

4

Scratch area:

[4, 3, 5, 1, 11, 8]
lastval = 4
1:7 so 1-6
is 4 > vals[i]

vals[1] ≤ 4
so do nothing?