SI 206 WINTER 2017, MIDTERM EXAM

Name: _____

UMID: _____

(2 points) Your UMID is the 8-digit number on your M-Card.
Please write it at the top of every page of the exam.
You will earn 15 points by writing it on all 14 pages.

If you do not know your UMID, write your unique name (what comes before your @umich.edu e-mail address)

This exam is worth **2850 points**. There are 22 questions.

We suggest that you take a quick pass to look through the entire exam and answer questions you are confident about before coming back to questions that will take you more time.

This exam is closed book. You are allowed one page of notes (double-sided). If a question seems unclear - please write down any assumptions you feel are needed. If you think that there is a just-plain mistake/typo, check with an instructor.

Anywhere we ask you what will be printed out, if you think an error will be generated, you may write "error". You do not need to write out what the whole error message would be.

Please write all of your answers inside the provided answer boxes, or on the answer lines, and **circle** multiple choice answers.

DO NOT WRITE ANSWERS ON DIFFERENT PAGES FROM THE ANSWER SPACE, OR ON THE BACK OF PAGES, because we will not be able to grade those.

The last page of this exam is a provided Regular Expression character "cheat sheet". You may tear it off the exam if you want to refer to it. Do not write any answers on it!

1. What is the type of `btz` after the following executes? (50 points)

```
btz = ["a","b","c"].split()
```

   a. String

   b. List

   c. Dictionary

   d. Tuple

   e. None of the above, there is an error

2. What is the output after the following code has been executed? (50 points)

```
diction = {"d":4,"e":{"g":4}}
print(diction["e"]["g"])
```

answer

3. What is the output after the following code has been executed? (100 points)

```
def func(x):
   for item in x:
        return x

print(func([1,2]))
```

answer

The following few questions are based upon the following selection of code, similar to the cards.py code you have seen before. (The code is on the next page.)

cards.py

```
 5  import random
 6
 7  class Card(object):
 8      suit_names =  ["Diamonds","Clubs","Hearts","Spades"]
 9      rank_levels = [1,2,3,4,5,6,7,8,9,10,11,12,13]
10      faces = {1:"Ace",11:"Jack",12:"Queen",13:"King"}
11
12      def __init__(self, suit=0,rank=2):
13          self.suit = self.suit_names[suit]
14          if rank in self.faces: # self.rank handles printed representation
15              self.rank = self.faces[rank]
16          else:
17              self.rank = rank
18          self.rank_num = rank # To handle winning comparison
19
20      def __str__(self):
21          return "{} of {}".format(self.rank,self.suit)
```

4. Which lines of code in **cards.py** show assignment of values to **class variables**? (100 points)

answer

5. Which lines of code in **cards.py** show assignment of values to **instance variables**? (100 points)

answer

6. Which of the following statements are valid to create an instance of class **Card** and save it in the variable **x**? Circle any that apply. (150 points)

```
a. x = Card.create()

b. x = Card(2.3)

c. x = Card(5,3)

d. x = Card(2)

e. x = Card.__init__

f. x = Card(1,"Queen")
```

7. (250 points) Write an additional method for the **Card** class. You can assume the method is correctly placed inside the class, just like the methods you see in the code above. This method should be called `compare_rank`. It should accept an integer that represents a card rank (1-13) as input.

   The method should return a Boolean value **True** if the input rank is larger than this Card's rank, and should return the Boolean value **False** if the input rank is smaller or equal to this Card's rank.

   So, for example, the following code should work correctly when placed after the Card class definition, if the **compare_rank** method is defined correctly:

```
c = Card(2,11)
print(c.compare_rank(4)) # should print False
print(c.compare_rank(13)) # should print True
```

answer

____End questions that focus on the above cards.py code._____

8. What should the value of `var` be **so that all of the print statements are printed** to the console? (75 points)

```
var = ____???_____


if var[1:3] == "py":
   print("Yup!")
if var[-1] == 'x':
   print("Nice")
if len(var) == 4:
   print(False)
if 'm' in var:
   print("Done")
```

answer

9. **Given the code in #8**, is there only one possible **type** that the answer to question 8 must be? Or are there multiple possibilities? **If** there is only one possible type for that answer, <u>what type</u> must it be? **If** there are multiple possibilities, <u>what types are possible</u>? (75 points)

answer

10. What will print when the following code runs? If there will be an error, say so. (100 points)

```
stuff = {'purple': [9, 106, 506, 'SI', 'Yu-Jen', 'Mauli'],
'blue': "Cornflower", 42: 'green'}

print(stuff["purple"][3] + stuff["blue"][0])
```

answer

11. **Matching.** Below are several regular expressions and several strings. Next to each string, write the letter(s) corresponding to at least one regular expression which would match that string.

    (Recommendation: take each string, and then go through each regular expression with the provided RegEx "cheat sheet" and decide which match it, then same for the next string, etc.) (150 points)

REGULAR EXPRESSION PATTERNS

A. `text`

B. `t*n`

C. `^P[^a-z]*`

D. `\$$`

E. `.*!`

F. `.+`

STRINGS TO MATCH WITH REGULAR EXPRESSION PATTERNS

a) text, man. _____

b) tnnnnn _____

c) Python _____

d) 0123abcd! _____

e) Hey! _____

12. Which of the following English descriptions is true of this regular expression: **`[^0-9]+[A-z]!$`** Select all that apply. (150 points)

    a. It describes a string that starts with any digit 0-9

    b. It describes a string that starts with any character that is NOT a digit 0-9

    c. It describes a string that must have at least 2 characters

    d. It describes a string that ends with an exclamation point !

    e. It describes a string that has no alphabetic characters (A-Z, a-z) in it

13. True or False: functions and methods are required to have `return` statements written out inside the function body. (50 points)

    a. True

    b. False

14. Given the below description of a class and its methods, which of the following tests would help a programmer test whether the code for the class definition works correctly, according to the description, if they were run in a code file? Circle all tests that apply. (You can assume that the **unittest** module is imported at the top of the file and there is no additional unittest code.) (200 points)

The class Student takes a name as input, which gets assigned to the instance variable name **when an instance is created. When an instance of Student is created, an instance variable** number_papers **is initialized with the variable 0.**

The class Student has a method write_paper**, which takes no additional input, but when it is invoked, the number of papers in the** number_papers **instance variable should increase by 1. The class Student also has a special string method (__str__) which returns a string with the name and the number of papers written, e.g. "Aya has written 12 papers."**

a.
```
class StudentTests(unittest.TestCase):
    def test_name(self):
        self.assertEqual(s.name,"Aya")
```

b.
```
class StudentTests(unittest.TestCase):
    def test_name(self):
        self.assertEqual(Student("Aya"),("Aya"))
```

c.
```
class StudentTests(unittest.TestCase):
    def name(self):
        s = Student("Lauren")
        self.assertEqual(s.name, "Lauren")
```

d.
```
class StudentTests(unittest.TestCase):
    def test_name(self):
        s = Student("Ken")
        self.assertEqual(s.name,"Ken")
```

e.
```
class StudentTests(unittest.TestCase):
    def test_str(self):
        s2 = Student("George")
        self.assertEqual(print(s2),"George   has   written   0
  papers."
```

15. Write two more tests for class Student. You should create a new test class in which to do so.

    - one test for the `__str__` method of the class Student
    - AND another test for the Student class constructor that tests something NOT tested in the options for Question 14

    to test whether their code works correctly according to the descriptions above.

    (250 points)

answer

15. Which of the following first lines of function definitions are acceptable for defining a function that takes in one required parameter and one optional parameter? Select all that apply. (100 points)

a. `def funct(z, y):`

b. `def funct(y = 7, z):`

c. `def funct(y = 1, z = 0):`

d. `def funct(y, z = 10):`

e. `def funct(z, y = 9):`

Next problem on the next page for space.

16. Given the following string formatted in an HTML-specific way, write code to load it into BeautifulSoup and print the phrase **Hello, 206!** (You can assume that the correct import statement, `from bs4 import BeautifulSoup`, has been typed at the top of your code file.) .(135 points)

REMEMBER:

- to create a BeautifulSoup object, you need to pass both the html string you want to create a BeautifulSoup object out of, and the string value `"html.parser",` to the BeautifulSoup constructor.
- The **.find_all(…)** method on a BeautifulSoup object takes two inputs: first, a required tag name, and second, an optional dictionary describing unique properties of the elements you want to find. It returns a list: either an empty list, or a list of BeautifulSoup objects that match the characteristics you input.
- The **.text** attribute on a BeautifulSoup object access the text (the plain string) in the middle of all the tags in that BeautifulSoup object, if there is any text there.

```
htmlstr = """<html>
   <body>
        <div id="main" class="section">
             <span><h2>Hello, 206!</h2></span>
        </div>
        <div id="secondary" class="section">
             <ol>
                  <li><a href="http://umich.edu">UM</a></li>
                  <li>This one is just text.</li>
             </ol>
        </div>
   </body>
</html>"""
```

answer

17. Given a similar string formatted in an HTML-specific way (shown below), write code to load it into BeautifulSoup and print each URL inside the HTML on a separate line. (You can assume that the correct import statement, `from bs4 import BeautifulSoup`, has been typed at the top of your code file.)

    HINT: Get the URL strings into a list, using BeautifulSoup and Python, then iterate over the list to print each one… the **.find_all** method on a BeautifulSoup object may be useful! You can access the **href** attribute of a BeautifulSoup object that has an **href** attribute by indexing, e.g. `this_is_a_bs_obj['href']` (150 points)

```
html_str = """<html>
   <body>
        <div id="main" class="section">
             <span><h2>Hello, 206!</h2></span>
             <a href="http://piazza.com">Piazza</a>
        </div>
        <div id="secondary" class="section">
             <ol>
                  <li><a href="http://umich.edu">UM</a></li>
                  <li>This one is just text.</li>
             </ol>
        </div>
   </body>
</html>"""
```

answer

18. Given this following code, you want to sort `lst1` to end up in the following order: **`["happy","absence","something","synergy"]`** . Which of these 3 named functions should you pass to the **key parameter** of the **sorted** function (replacing the question marks below)? (100 points)

```
lst1 = ["happy","absence","synergy","something"]
sorted_lst1 = sorted(lst1, key= ___???____)

# Could replace the ??? there with: second_elem OR
sec_elements OR sort_by. Each of their definitions is provided
below. Which function should you use?
```

a.

```
def second_elem(x):
     return x[1]
```

b.

```
def sec_elements(lst):
     for x in lst:
          return x
```

c.

```
def sort_by(lst):
     return len(lst)
```

19. You want to define a function called `build_string` that accepts a list of tuples as input. Each tuple in the input will have 2 strings as elements. For example, `[("hi","all"),("music","play"),("celebration","station")]`. The function should return one string which is made up of the first character in each string inside the tuples. So the invocation `build_string([("hello","all"),("music","playing"),("celebration","station")])` should return the string `"hampcs"`. We have provided most of this function. Fill in the missing pieces of code on the provided lines, so that the function will work as specified. (150 points)

```
def build_string(list_of_tups):

    final_str = _______ # fill in a bit here

    for ____________________: # fill in a bit here


        for item ____________________: # fill in a bit here


            final_str  = final_str + item[0]

    return final_str
```

The next couple questions depend on the following code.

```
import requests
import json

baseurl = "https://itunes.apple.com/search"
params = {}
params["term"] = "Franz Ferdinand"
params["entity"] = "musicVideo"
r = requests.get(baseurl, params=params)
```

20. Assuming all of the information above is valid (valid base URL string, valid query parameters) for the iTunes Search API, after that code is executed, what is the type of the variable **r**? (100 points)

a. A File object

b. A BeautifulSoup object

c. A Response object

d. A string

e. A dictionary

21. If you wanted to get a Python object to process from that code above that makes a request to the iTunes Search API, and you wanted to save the Python object in a variable called `python_obj`, what should the next line of code you write in your file be? Assume this line of code will go directly after `r = requests.get(baseurl, params=params).`

(150 points)

answer

End questions based on above code.

22. Assume you have the following Python dictionary, but no other code written yet. Which below set of code should you write to save the data in this Python dictionary to a file? (100 points)

```
diction = {"python":[106,206,330],"html":[364,339],"intro":[106,110]}
```

a.

```
res = json.dumps(diction)
f.write(res)
f.close()
```

b.

```
res = json.dumps(diction)
f = open("newfile.json","w")
f.write(res)
f.close()
```

c.

```
res = json.dumps(diction)
f = open("newfile.json","w")
f.write("res")
f.close()
```

d.

```
res = json.dumps(diction)
f = open("newfile.json","r")
f.readlines() = res
f.close()
```

Python Regular Expression Quick Guide

```
^        Matches the beginning of a line
$            Matches the end of the line
.            Matches any character
\s           Matches whitespace
\S           Matches any non-whitespace character
*            Repeats a character zero or more times
*?           Repeats a character zero or more times (non-greedy)
+         Repeats a character one or more times
+?           Repeats a character one or more times(non-greedy)
[aeiou]    Matches a single character in the listed set
[^XYZ]     Matches a single character not in the listed set
[a-z0-9]   The set of characters can include a range
(              Indicates where string extraction is to start
)              Indicates where string extraction is to end
```

NOTE: please DO turn this page in!

We need all the exams to have the same number of pages in order to scan them into our grading tool.

You do not need to do any work on this page and you should not write any answers on it.