NYU – Tandon School of Engineering
Brooklyn, NY 11201

CS-UY 1114 / Python

First Midterm Exam – 19 November 2019

Name: _____

NetID (first part of email):_____

- Duration: 1 hour, 15 minutes
- **DO NOT WRITE ON THE BACK OF ANY PAGE!**
- Do not separate any page.
- **Please do not use pencil**, if you must, write darkly, these pages will be scanned
- If you write an answer other than in the space provided, please indicate in the space provided, where we can find that answer
- This is a closed book exam, no calculators are allowed
- You can expect that the user inputs the appropriate values (int/float/etc where required).
- Comments are not required
- Anyone found cheating on this exam will receive a zero for the exam.
- Anyone who is found writing after time has been called will receive a zero for this exam.
- Do not open this test booklet until you are instructed to do so.
- If you have a question please ask the proctor of the exam!
- Descriptive variable names are required
- Answers that are unnecessarily convoluted will result in a point deduction.
- **Global variables are NOT PERMITTED on this exam.**
- You may use ONLY the following Python constructs and functions!

math operators: `+`, `-`, `/`, `//`, `%`, `*`, `**`	conditional statements: `if`, `elif`, `else`
boolean operators: `and`, `or`, `not`	`while` and `for` loops, `def`, and `return`
all functions in modules `random`, `turtle`, `math`, and `string`	coercion functions: `int`, `str`, `float`
assignment operator (=) and augmented assignment operators: `+=`, `-=` , `*=`, `/=`, `//=`, `%=`, `**=`	All string methods (such as `s.find`, `s.split`, `s.upper`, `s.isupper`, `s.isalpha`, `s.join`, etc)
comparison operators: `==`, `!=`, `<`, `<=`, `>`, `>=`	slice and indexing operators
All list methods (such as `lst.pop`, `lst.sort`, `lst.append`, `lst.copy`, etc) and `len`	

Name _____ Net ID: _____

Question 1 (20 Pts; 2 pts each)

For each of the following, show the final value of "var" at the end of executing the expressions, or write ERROR if there is an error. Make sure to show EXACTLY what Python would produce if we ran the code "`print(var)`" after the statements below. **Each block is separate and the value of "var" is reset each time.**

	Code Fragment	Output of print(var) after the code fragment runs or ERROR
a)	`var = [1, 2, 3]` `var[1] = var[2]+5`	
b)	`var = 'Exam ' + ('i' *2)`	

c)	`var = 'Exam '`
	`number = 2`
	`var = var + (number*2)`	

d)	`st = 'abc'` `var=st.find('bc')`	
e)	`var = 'Exam 1'` `var[5] = '2'`	

f)	`var = [1, 2, 3]`
	`other_var = [4, 5]`
	`var.append(other_var)`	

g)	`ls = ['a','b','c']` `var = ls.find('b')`	
h)	`st = 'aaabbcccd'` `var = st.find('ac')`	
i)	`st = 'aaabbcccd'` `var = st[1::3]`	
j)	`var = [1, 2, 3]` `var.extend([20, 30]);`	

Name _____ Net ID: _____

Question 2 (15 points):

Given the following definitions, what would be printed when calling **`main()`**? If the code produces an error, then your output should simply be "ERROR".

```
def func1(n):
    if n == 5:
        print("func1 msg1: n =", n)
    else:
        print("func1 msg2: n =", n)
        func2(n - 2)
        print("func1 msg3: n =", n)

def func2(n):
    if n == 2 or n == 6:
        print("func2 msg1: n =", n)
    else:
        print("func2 msg2: n =", n)
        n -= 2
        print("func2 msg3: n =", n)

def main():
    n = 10
    print("main msg1: n =", n)
    func1(n - 1)
    print("main msg2: n =", n)
```

OUTPUT:

Name _____ Net ID: _____

Question 3 (10 points):

Given the following definitions, what would be printed when calling **`main()`**? If the code produces an error, then your output should simply be "ERROR".

```
def func1(char_list):
    chars = '!$%\t\t&*()_+=-}{[]|."'
    char_list[0] = chars[10:14]
    char_list.append(chars[-6:-9])
    char_list.append(chars[3:4])
    char_list.append(chars[12])
    char_list[0] = chars[-2:-1]
    char_list.insert(1, chars[-3])

def func2(chars, some_str):
    some_str = some_str.replace('$', chars[3])
    some_list = [some_str[16],  some_str[10], some_str[7]];
    some_str = '-'.join(some_list)
    return func3(some_str.upper())

def func3(input):
    output = ''
    for value in input:
        output = value + output
    return output

def main():
    char_list = ['']
    func1(char_list)
    print(func2(char_list, 'ot$frozn$iyo$q2wu'))
```

OUTPUT: (Write your solution of the output ONLY in this box)

Name _____ Net ID: _____

Question 4 (55 Points, multiple parts, partial credit):

In the Pokemon trading card game some cards are “Basic,” some are “Stage 1,” and some are “Stage 2.” A Stage 1 card can only be used if it has evolved from the Basic card associated with it, and a Stage 2 card can only be used if it evolves from a Stage 1 card associated with it. For example, The Basic card, “Pichu” can only evolve into the Stage 1 card “Pikachu”, and “Pikachu” can only evolve into “Raichu”. As a result, having a “Pikachu” without a “Pichu” would be useless because the “Pikachu” could not be played without first playing a “Pichu”. (i.e. `Pichu -> Pikachu -> Raichu )`

A card collector has a very large deck of cards and knows that there are some Stage 1 or 2 cards for which he does not have the Stage 1 or Basic and so cannot play that card. They have asked us to write a program which can determine which of the cards in his deck can be played and which cannot. Basic cards can always be played; Stage 1 cards can only be played if the player also has the Basic card; Stage 2 cards can only be played if the player also has the Basic card and Stage 1 card. All card names are unique.

Nintendo, the company that makes Pokemon, has provided us with a Python list of lists of strings with the names of the cards and their associations with respect to Basic, Stage 1, or Stage 2. For example, if we have the following associations:

```
Pichu -> Pikachu -> Raichu
Abra -> Kadabra -> Alakazam
Oshawott -> Dewott -> Samurott
```

The list of lists would be:

```
PokeWorld = [[‘Pichu’, ‘Pikachu’, ‘Raichu’], [‘Abra’, ‘Kadabra’, ‘Alakazam’], [‘Oshawott’, ‘Dewott’, ‘Samurott’]]
```

(Question continued on next page)

Name _____ Net ID: _____

Please write the following functions in the spaces provided on the next pages, or, in those spaces, indicate where you have written your answers.

A. (5 pts) Write a function, “cardInDeck” which takes a list of strings (not the PokeWorld above) and a given card and determines if the card is in the list. This function should return True or False.
```
#sig : list (string) -> boolean
```
B. (20 pts) Write a function, “getEvolutionSet” which, given the PokeWorld and the name of a Pokemon card, returns a list as follows:
    a. If the Pokemon is a Basic card, this returns an empty list (i.e. given ‘Pichu’ it will return [])
    b. If the Pokemon is a Stage 1 card, this returns a list with the name of the associated Basic card (i.e. given ‘Pikachu’ it will return [‘Pichu’])
    c. If the Pokemon is a Stage 2 card, this returns a list with the names of the Basic and Stage 1 cards (i.e. given ‘Raichu’ it will return [‘Pichu’, ‘Pikachu’])
```
#sig : list(list(string)), string -> list(string)
```
C. (15 pts) Write a function “canPlayCard” which takes the PokeWorld (list of lists of strings), the player’s deck (list of strings) and a single card (guaranteed to be in the deck), and determines if the card can be played based on the rules given above.
```
#sig : list(list(string)), list(string), string -> boolean
```
D. (15 pts) Write a function “checkDeck” which takes the PokeWorld (list of lists of strings), a player’s deck (list of strings) and returns a list of cards in the deck that cannot be played due to the rules above.
```
#sig : list(list(string)), list(string) -> list(string)
```

Note, you are NOT being asked to write all functions (i.e. main) but can use any of the functions above in answering a question for another function.

Write your answer below the function headers, lines are provided for convenience (N.B.: Your answer may be shorter or longer than the number of lines provided)

```
def cardInDeck(deck, card):
```

_____

_____

_____

_____

_____

_____

Name _____ Net ID: _____

```
def getEvolutionSet(world, card):
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

```
def canPlayCard(world, deck, card):
```

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Name _____ Net ID: _____

```
def checkDeck(world, deck):
```

Name ____ Net ID: ___________________

(additional space for answers)

(Additional space for answers)