Polytechnic Tutoring Center

Final Exam Review ANSWER KEY - CS 1114, Fall 2021

Disclaimer: This mock exam is only for practice. It was made by tutors in the Polytechnic Tutoring Center and is not representative of the actual exam given by the CS Department.

1. Indicate whether the Python types below are mutable or non-mutable by circling either…

Lists	**mutable**	non-mutable
Strings	mutable	**non-mutable**
Dictionaries	**mutable**	non-mutable
Integers	mutable	**non-mutable**
Tuples	mutable	**non-mutable**

2. Define a function which recursively prints out a right triangle, given the number of rows to print out and the type of character to print on each line. For example, if I call the function (say the name of my function is display_triangle, then display_triangle(5, "*") should print…

```
*
**
***
****
*****
```

```
def display_triangle(num, char):
    if num == 1:
        print(1 * char)
    else:
        display_triangle(num - 1, char)
        print(num * char)
```

3. Write a function which converts an integer n into a string which represents its binary number, note the string should go from most significant to least significant digit.

```
def convert_to_binary(num):
    acc = ''
    while (num != 0):
        acc = str(num % 2) + acc
        num //= 2
    return acc
```

4. Create the following lists using list comprehension.

a. lst1 = [1,2,2,3,3,3,4,4,4,4,5,5,5,5,5]

[i for i in range(1,6) for j in range(i)]

b. lst2 = [0, -1, 2, -3, 4, -5, 6, -7, 8]

[i * ((-1) ** i) for i in range(9)]

5. Write a function which takes in a filename and outputs a dictionary, containing all the words in the file, with the number of times each word appears in the file. Note: you may assume there is no punctuation or anything like that in the file, just words. Note as well that you should gracefully handle the possibility that the file might not exist or not be found. That is, you will need to use exception handling.

```
def count_words(filename):
    my_dict = {}

    try:
        # file_obj = open(filename, 'r')

        with open(filename, "r") as file_obj:
            for line in file_obj:
                line = line.strip().split()
                for item in line:
                    if item in my_dict:
                        my_dict [item] += 1
                    else:
                        my_dict [item] = 1

    except FileNotFoundError:
        continue

    return my_dict
```

6. For this question, you will need to write two classes: a Student class and Class class (the irony is not lost on me there). A Class must be created, knowing its instructor and its course name. Its students may be passed in as an empty list or as a list containing various Students. A Student is to be created knowing his/her name and GPA. A students classes may be created with no Classes, having an empty list of Classes, or with a list full of Classes. You will need to then write five functions. A Student has the option to drop a Class from the list of Classes he/she is taking. He/She also has the option to add a Class to his list of Classes if he/she so desires. Similarly a Class has the option of dropping a Student (I know this is weird, but we'll do it anyway), and a Class also has the option of adding a Student. Finally, the last function is that a Class should be able to calculate the average GPA of all of its Students.

```
def Students:

    def __init__(self, name, gpa, classes = []):
        self.classes = classes
        self.gpa = gpa
        self.name = name

    def drop_class(self, course_name):
        for i in range (len(self.classes)):
            if self.classes[i].course_name == course_name:
                self.classes.pop(i)

    def add_class(self, course):
        self.classes.append(course)
```

```
class Class:

    def __init__(self, course_name, instructor, students = []):
        self.course_name = course_name
        self.instructor = instructor
        self.students = students

    def drop_student(self, student_name):
        for i in range (len(self.students)):
            if self.students[i].name == student_name:
                self.students.pop(i)

    def add_student(self, student):
        self.students.append(student)

    def calculate_gpa_avg(self):
        acc = 0
        for i in range (len(self.students)):
            acc += self.students[i].gpa
        return acc/len(self.students)
```