**1) What is the output for the commands below? (5points)**

string = "aaaaccddeebbbff"

(a) pattern = "a?b"

match = re.search(pattern, string)

print match.group()

Output = b

(b) pattern="eb?b"

match = re.search(pattern, string)

print match.group()

Output = ebb

(c) pattern = "a{2}c"

match = re.search(pattern, string)

print match.group()

Output = aac

(d) pattern = "[^a-c]"

print re.findall(pattern, string)

Output = ['d', 'd', 'e', 'e', 'f', 'f']

(e) pattern = "^d"

print re.findall(pattern, string)

Output = []

**2) Please transform the imaginary phone list (input) using regular expressions to the output format (shown below) (5points)**

Input:

"555-8396 Neu, Allison",

"Burns, C. Montgomery",

"555-5299 Putz, Lionel",

"555-7334 Simpson, Homer Jay"

Output:

Allison Neu 555-8396

C. Montgomery Burns

Lionel Putz 555-5299

Homer Jay Simpson 555-7334

**import** re

Input **=** **[**"555-8396 Neu, Allison"**,** "Burns, C. Montgomery"**,** "555-5299 Putz, Lionel"**,**"555-7334 Simpson, Homer Jay"**]**

string **=** str**(**Input**)**

pattern **=** r'(?P<phone>\d{3}-\d{4})?\s?(?P<last>\w+),\s(?P<first>\w\*\.?\s?\w\*)'

output1 **=** re**.**finditer**(**pattern**,**string**)**

**for** i **in** output1**:**

phone\_nub **=** i**.**group**(**'phone'**)**

last\_name **=** i**.**group**(**'last'**)**

first\_name **=** i**.**group**(**'first'**)**

**print(**first\_name**,**last\_name**,**phone\_nub**)**

**3) Using the class definition below, which line creates a new Car object with 4 wheels and 2 doors? (5points)**

class Car(object):

def \_\_init\_\_(self, w, d):

self.wheels = w

self.doors = d

self.color = ""

(A) Car(mycar, 4, 2)

(B) mycar = Car(4, 2, "white")

**(C) mycar = Car(4, 2)**

(D) mycar = Car(2, 4)

**4) Which of the following methods changes the color of the car, based on the definition below? (5points)**

class Car(object):

def \_\_init\_\_(self, w, d):

self.wheels = w

self.doors = d

self.color = ""

(A) def paint(c):

color = c

(B) def paint(self, c):

color = c

(C) def paint(c):

self.c = c

**(E) def paint(self, c):**

**self.color = c**

**5) You create a car with mycar = Car(4, 2). Which is a line of code to change the color of mycar to "red"? (5points)**

class Car(object):

def \_\_init\_\_(self, w, d):

self.wheels = w

self.doors = d

self.color = ""

def paint(self, c):

self.color = c

(A) Car.paint("red")

(B) mycar.paint(red)

**(C) mycar.paint("red")**

(D) mycar.paint(Car, "red")

**6) With the code below, what does the line print(mycar == yourcar)print? (5points)**

class Car(object):

def \_\_init\_\_(self, w, d):

self.wheels = w

self.doors = d

self.color = ""

def paint(self, c):

self.color = c

def \_\_eq\_\_(self, other):

if self.wheels == other.wheels and \

self.color == other.color and \

self.doors == other.doors:

return True

else:

return False

mycar = Car(4, 2)

mycar.paint("red")

yourcar = Car(4,2)

print(mycar == yourcar)

(A) True

**(B) False**

(C) An error

**7) Assume AGUACACUGGU is a RNA sequence, which of the following operations in Biopython can you perform on it (1point)**

1. Transcribe
2. Translate
3. Back-Transcribe
4. Transcribe: RNA cannot be transcribed!
5. Translate: Seq('STL', IUPACProtein())
6. Back-Transcribe: Seq('AGTACACTGGT', IUPACUnambiguousDNA())

**Problems (8) and (9) are 10 points each**

**Students should enclose the program (separately) they have used to answer these two problems along with the answer as a separate file titled Problem#\_StudentName.py. Use Biopython and regular expression where ever applicable. You cannot use an alternative way when Biopython modules or regular expression modules exist to solve this problem.**

**8)** Please read the fast sequence from the file Input.fasta using Biopython.

**(a)** Please compute their GC content using the appropriate utility module of Biopython and output your results in tab-delimited format with the sequence-id and GC content in a .txt file with name Problem8\_yourname.txt.

**(b)** Please transcribe and translate this using Biopython, by using (a) Standard codon table (id: 1) and (b) Mitochondrial codon table (id: 2). Please print out the transcribed and translated sequence as appended to the same .txt file (output of (a)).

**(c)** Also please print all the stop codons and their positions on the sequence from the two-codon tables (id: 1 and id: 2). The output should be appended to the same .txt file (output of (b)) as a table of the form:

Codon-table-name (id)\tCodon\tPosition-1, Position-2, …,Position-n

**Note**: \t denotes tab-delimited format.

**9)** Please read the fast sequence from the file Final\_exam\_Regex\_input.fasta using Biopython.

Use regular expression to find the position of the pattern “ggatcc”, “cggccg”, “gccggc”, “gaattc”. Please write the output into a .txt file(Problem9\_yourname.txt).

The output should be a tab-delimited table of the form:

Pattern\tPosition-1, Position-2, …,Position-n

**Note**: \t denotes tab-delimited format