**Question 1** Given a variable text below, write a Python program to find answers to following sub- questions.

*Text = "Harry lay in his dark cupboard much later, wishing he had a watch. He didn't know what time it was and he couldn't be sure the Dursleys were asleep yet. Until they were, he couldn't risk sneaking to the kitchen for some food. He'd lived with the Dursleys almost ten years, ten miserable years, as long as he could remember, ever since he'd been a baby and his parents had died in that car crash. He couldn't remember being in the car when his parents had died. Sometimes, when he strained his memory during long hours in his cupboard, he came up with a strange vision: a blinding flash of green light and a burning pain on his forehead. This, he supposed, was the crash, though he couldn't imagine where all the green light came from. He couldn't remember his parents at all. His aunt and uncle never spoke about them, and of course he was forbidden to ask questions. There were no photographs of them in the house.*

(1) Calculate the number of words that start with a vowel (a, e, i, o, and u). Print out a list of these words, along with their frequency (number of times they appear in the text). The list should be in decreasing order of frequency.

**PROGRAM**

# define the text to be analyzed

text = "Harry lay in his dark cupboard much later, wishing he had a watch. He didn't know what time it was and he couldn't be sure the Dursleys were asleep yet. Until they were, he couldn't risk sneaking to the kitchen for some food. He'd lived with the Dursleys almost ten years, ten miserable years, as long as he could remember, ever since he'd been a baby and his parents had died in that car crash. He couldn't remember being in the car when his parents had died. Sometimes, when he strained his memory during long hours in his cupboard, he came up with a strange vision: a blinding flash of green light and a burning pain on his forehead. This, he supposed, was the crash, though he couldn't imagine where all the green light came from. He couldn't remember his parents at all. His aunt and uncle never spoke about them, and of course he was forbidden to ask questions. There were no photographs of them in the house."

# define a function to determine if a word starts with a vowel

def starts\_with\_vowel(word):

vowels = set(['a', 'e', 'i', 'o', 'u']) # define a set of vowels

return word[0].lower() in vowels # return True if the first letter of the word is a vowel, otherwise False

# create a dictionary to store the frequency of each word that starts with a vowel

vowel\_words = {}

# loop through each word in the text

for word in text.split():

if starts\_with\_vowel(word): # check if the word starts with a vowel

if word not in vowel\_words: # check if the word is already in the dictionary

vowel\_words[word] = 1 # if not, add it with a frequency of 1

else:

vowel\_words[word] += 1 # if it is, increment its frequency

# sort the dictionary by frequency in descending order

sorted\_vowel\_words = sorted(vowel\_words.items(), key=lambda x: x[1], reverse=True)

# print the list of words that start with a vowel and their frequency

print("Words that start with a vowel and their frequency:")

for word, frequency in sorted\_vowel\_words:

print(word, frequency)

**OUTPUT**

Graphical user interface

Description automatically generated

2) Print out words that have an apostrophe (‘) in the middle of the word. Examples include English contractions such as “didn’t”, “wasn’t”, “couldn’t", or "she'd".

**PROGRAM**

# Create an empty list to store words with apostrophes

apostrophe\_words = []

# Split the text into individual words

words = text.split()

# Loop through each word in the text

for word in words:

# Check if the word contains an apostrophe

if "'" in word:

# If it does, add the word to the apostrophe\_words list

apostrophe\_words.append(word)

# Print the list of words with apostrophes

print("Words with apostrophes:")

print(apostrophe\_words)

**OUTPUT**

**Graphical user interface, text, application

Description automatically generated**

(3) Remove all stopwords, and words with an apostrophe in the text and print it out. You can find a list of English stopwords here: <http://www.ranks.nl/stopwords>.

**PROGRAM**

# define the text to be analyzed

text = "Harry lay in his dark cupboard much later, wishing he had a watch. He didn't know what time it was and he couldn't be sure the Dursleys were asleep yet. Until they were, he couldn't risk sneaking to the kitchen for some food. He'd lived with the Dursleys almost ten years, ten miserable years, as long as he could remember, ever since he'd been a baby and his parents had died in that car crash. He couldn't remember being in the car when his parents had died. Sometimes, when he strained his memory during long hours in his cupboard, he came up with a strange vision: a blinding flash of green light and a burning pain on his forehead. This, he supposed, was the crash, though he couldn't imagine where all the green light came from. He couldn't remember his parents at all. His aunt and uncle never spoke about them, and of course he was forbidden to ask questions. There were no photographs of them in the house."

# create a list of words that do not appear in the stop\_words set and do not contain an apostrophe

words = [word for word in text.split() if word not in stop\_words and "'" not in word]

# print the list of words without stopwords and words with an apostrophe

print("Words without stopwords and words with an apostrophe:")

print(words)

**OUTPUT**

A picture containing text

Description automatically generated

**Question 2.**  I am interested in knowing how the climate changes in terms of temperature and precipitation. The U.S. climate data site contains climate data for Denton, Texas since 2007. I would like you to do some calculation and comparison between the data in 2011 and 2018.

1. Create two files based on data published at U.S. climate data (<http://www.usclimatedata.com/climate/denton/texas/united-states/ustx0353>):

* File A (should be called 2011-Jan-June.txt, or 2011-Jan-June.csv) contains daily weather data from January 1, 2011 to June 30, 2011;
* File B (called 2018-Jan-June.txt, or 2018-Jan-June.csv) contains daily weather data from January 1, 2018 to June 30, 2018;

To find the data, go to “History” tab of the above page, select the right year and month. You will see the data being presented to you.

The final format of each result file should look like the following:

<Date-month>, <High>, <Low>, <Precip>

1-Jan,55,33,0.08

2-Jan,55,33,0.12

……

1-June,80,56,0,15

…

The delimiter can be comma (,) or whitespace. Make sure you round the numbers for the temperature so there is no decimal points.

(2) Write a program to calculate the mean, median, and standard deviation of high temperature, low temperature and precipitation of each file, and output the results in the following format:

File name mean median standard deviation

2011-Jan-June.txt ----- ----- -----

2018-Jan-June.txt

**PROGRAM**

import csv

import numpy as np

# Define function to calculate mean, median, and standard deviation

def calc\_stats(data):

mean = round(np.mean(data), 2)

median = round(np.median(data), 2)

std\_dev = round(np.std(data), 2)

return mean, median, std\_dev

# Define function to read in data from file

def read\_data(filename):

with open(filename, 'r') as file:

reader = csv.reader(file)

next(reader) # skip header row

dates = []

highs = []

lows = []

precip = []

for row in reader:

date = row[0]

high = float(row[1])

low = float(row[2])

prec = float(row[3])

dates.append(date)

highs.append(high)

lows.append(low)

precip.append(prec)

return dates, highs, lows, precip

# Read in data from 2011-Jan-June.csv

filename = '2011-Jan-June.csv'

dates, highs, lows, precip = read\_data(filename)

# Calculate statistics for 2011 data

mean\_highs\_2011, median\_highs\_2011, std\_dev\_highs\_2011 = calc\_stats(highs)

mean\_lows\_2011, median\_lows\_2011, std\_dev\_lows\_2011 = calc\_stats(lows)

mean\_precip\_2011, median\_precip\_2011, std\_dev\_precip\_2011 = calc\_stats(precip)

# Print results for 2011 data

print('File name\tmean\t\tmedian\t\tstandard deviation')

print('2011-Jan-June.csv\t{}\t{}\t{}'.format(mean\_highs\_2011, median\_highs\_2011, std\_dev\_highs\_2011))

print('\t\t{}\t{}\t{}'.format(mean\_lows\_2011, median\_lows\_2011, std\_dev\_lows\_2011))

print('\t\t{}\t{}\t{}'.format(mean\_precip\_2011, median\_precip\_2011, std\_dev\_precip\_2011))

# Read in data from 2018-Jan-June.csv

filename = '2018-Jan-June.csv'

dates, highs, lows, precip = read\_data(filename)

# Calculate statistics for 2018 data

mean\_highs\_2018, median\_highs\_2018, std\_dev\_highs\_2018 = calc\_stats(highs)

mean\_lows\_2018, median\_lows\_2018, std\_dev\_lows\_2018 = calc\_stats(lows)

mean\_precip\_2018, median\_precip\_2018, std\_dev\_precip\_2018 = calc\_stats(precip)

# Print results for 2018 data

print('2018-Jan-June.csv\t{}\t{}\t{}'.format(mean\_highs\_2018, median\_highs\_2018, std\_dev\_highs\_2018))

print('\t\t{}\t{}\t{}'.format(mean\_lows\_2018, median\_lows\_2018, std\_dev\_lows\_2018))

print('\t\t{}\t{}\t{}'.format(mean\_precip\_2018, median\_precip\_2018, std\_dev\_precip\_2018))

**OUTPUT**

**Graphical user interface, text, application, email

Description automatically generated**

**Question 3.** We have discussed how we could create a Python class, and how to define the properties, functions, and data types it would require. For this question, you are required to create and implement the class *Student* in the “<LastName>\_assognment-three-code.ipynb” file*.* Your class should be able to do the following:

1. Every instance of the *Student* class should have the following attributes:

* First Name
* Last Name
* Middle Name
* euid, by default euid is blank.
* GPA, by default the GPA is set to 0
* Classes taken. This object should store the class number (such as INFO 5717), the semester it was taken it (such as "Fall 2019"), and the grade the student obtained. HINT: use a collection data type.

1. Write a program in the “<LastName>\_assognment-two-code.ipynb” file that asks the user to enter information for three fictional students. The program should prompt the user to ask for the student's first, middle, and last names. This information should be stored in an object of type Student.

**PROGRAM**

class Student:

def \_\_init\_\_(self, first\_name, last\_name, middle\_name=''):

self.first\_name = first\_name

self.last\_name = last\_name

self.middle\_name = middle\_name

self.euid = ''

self.gpa = 0

self.classes\_taken = []

# Create three instances of the Student class by prompting user for input

students = []

for i in range(3):

first\_name = input(f"Enter the first name of student {i+1}: ")

middle\_name = input(f"Enter the middle name of student {i+1} (press enter if none): ")

last\_name = input(f"Enter the last name of student {i+1}: ")

student = Student(first\_name, last\_name, middle\_name)

students.append(student)

print(f"Student {i+1} added successfully!\n")

# Print the student information

for i, student in enumerate(students):

print(f"Student {i+1} Information:")

print(f"First Name: {student.first\_name}")

print(f"Middle Name: {student.middle\_name}")

print(f"Last Name: {student.last\_name}\n")

**OUTPUT**

**Graphical user interface, text, application

Description automatically generated**

1. Write a function for the *Student* class that assigns each student a euid. The euid should be composed of three letters (the first three letters of the student's first, middle, and last names) and then 4 digits chosen randomly. For example, if a student's name is Jennifer Lynn Meyers her euid could be "jlm8940". Use this function to print out the names and euids of the students you created in (2).

**PROGRAM**

import random

class Student:

def \_\_init\_\_(self, first\_name, middle\_name, last\_name):

self.first\_name = first\_name

self.middle\_name = middle\_name

self.last\_name = last\_name

self.euid = ""

self.gpa = 0

self.classes\_taken = []

def add\_class(self, class\_number, semester, grade):

self.classes\_taken.append((class\_number, semester, grade))

def generate\_euid(self):

# Get the first three letters of the student's first, middle, and last names

first\_initial = self.first\_name[:1]

middle\_initial = self.middle\_name[:1]

last\_initial = self.last\_name[:1]

# Generate a random 4-digit number

random\_number = random.randint(1000, 9999)

# Concatenate the letters and numbers to create the euid

self.euid = first\_initial + middle\_initial + last\_initial + str(random\_number)

# Create an empty list to store our students

students = []

# Prompt the user to enter information for three students

for i in range(3):

first\_name = input("Enter the student's first name: ")

middle\_name = input("Enter the student's middle name: ")

last\_name = input("Enter the student's last name: ")

# Create a new Student object and add it to the list of students

student = Student(first\_name, middle\_name, last\_name)

students.append(student)

# Generate euids for each student and print them out

for student in students:

student.generate\_euid()

print(f"{student.first\_name} {student.middle\_name} {student.last\_name}: {student.euid}")

**OUTPUT**

**Graphical user interface, text, application

Description automatically generated**

1. Write a function for the *Student* class called "register". This function asks the user to enter a class number, a semester, and a grade for each student. Then this information is added to the *Student* object. Call this function for the students you created in part (2). Each student should be registered for at least two courses but can have more than.

**PROGRAM**

class Student:

def \_\_init\_\_(self, first\_name, middle\_name, last\_name):

self.first\_name = first\_name

self.middle\_name = middle\_name

self.last\_name = last\_name

self.euid = ""

self.gpa = 0

self.classes\_taken = []

def register(self):

class\_number = input("Enter the class number: ")

semester = input("Enter the semester (e.g. Fall 2022): ")

grade = input("Enter the grade (A, B, C, D, F): ")

self.classes\_taken.append((class\_number, semester, grade))

# create three student objects

student1 = Student("John", "Doe", "Smith")

student2 = Student("Jane", "Doe", "Johnson")

student3 = Student("Bob", "Jones", "Brown")

# register each student for at least two courses

student1.register()

student1.register()

student2.register()

student2.register()

student3.register()

student3.register()

# print out the classes taken for each student

print(student1.classes\_taken)

print(student2.classes\_taken)

print(student3.classes\_taken)

**OUTPUT**

Graphical user interface, text, application, Word

Description automatically generated

1. Write a function for the *Student* class that calculates a student's GPA and prints it out. Use it to print out the names and GPAs of the student's you created in (2).

**PROGRAM**

class Student:

def \_\_init\_\_(self, first\_name, last\_name, middle\_name=''):

self.first\_name = first\_name

self.last\_name = last\_name

self.middle\_name = middle\_name

self.euid = ''

self.GPA = 0

self.classes\_taken = []

def assign\_euid(self):

self.euid = self.first\_name[:3].lower() + self.middle\_name[:3].lower() + self.last\_name[:3].lower() + str(random.randint(1000,9999))

def register(self, class\_number, semester, grade):

self.classes\_taken.append({'class\_number': class\_number, 'semester': semester, 'grade': grade})

def calculate\_gpa(self):

total\_points = 0

total\_credits = 0

for course in self.classes\_taken:

if course['grade'] == 'A':

total\_points += 4.0

elif course['grade'] == 'B':

total\_points += 3.0

elif course['grade'] == 'C':

total\_points += 2.0

elif course['grade'] == 'D':

total\_points += 1.0

elif course['grade'] == 'F':

total\_points += 0.0

total\_credits += 3.0

self.GPA = round(total\_points/total\_credits, 2)

def display\_student\_info(self):

print('Name:', self.first\_name, self.middle\_name, self.last\_name)

print('EUID:', self.euid)

print('GPA:', self.GPA)

print('Classes Taken:')

for course in self.classes\_taken:

print('Class Number:', course['class\_number'], '- Semester:', course['semester'], '- Grade:', course['grade'])

# creating instances of the Student class

student1 = Student('John', 'Doe')

student2 = Student('Jane', 'Doe')

student3 = Student('Mike', 'Smith')

# registering students for classes

student1.register('INFO 5717', 'Fall 2020', 'B')

student1.register('INFO 5502', 'Spring 2021', 'A')

student2.register('INFO 5717', 'Fall 2020', 'A')

student2.register('INFO 5502', 'Spring 2021', 'A')

student3.register('INFO 5717', 'Fall 2020', 'B')

student3.register('INFO 5502', 'Spring 2021', 'B')

# calculating GPAs for each student

student1.calculate\_gpa()

student2.calculate\_gpa()

student3.calculate\_gpa()

# displaying student information

print('Student 1 Info:')

student1.display\_student\_info()

print('------------------------')

print('Student 2 Info:')

student2.display\_student\_info()

print('------------------------')

print('Student 3 Info:')

student3.display\_student\_info()

**OUTPUT**

Graphical user interface, text, application, Word

Description automatically generated