Week-10, Practice Programming

Problem 1

Question

Create a Class `Calculator` that has the following methods:

- `Sum(a,b)` that returns `a + b`
- `Multiply(a, b)` that returns `a * b`
- `Subtraction(a, b)` that returns `a - b`
- `Division(a, b)` that returns `a / b`
- `Remainder(a, b)`that returns `a % b`
- `Power(a, b)` that returns `a ** b`
- `Quotient(a, b)` that returns `a // b`

Consider a and b to be positive integers.

Answer

```python
# Create class Calculator
class Calculator:
    # Method for sum
    def Sum(self,a,b):
        return a + b
    # Method for Multiply
    def Multiply(self,a, b):
        return a * b
    # Method for Subtraction
    def Subtraction(self,a, b):
        return a - b
    # Method for Division
    def Division(self,a, b):
        return a / b
    # Method for Remainder
    def Remainder(self,a, b):
        return a % b
    # Method for Power
    def Power(self,a, b):
        return a ** b
    # Method for Quotient
    def Quotient(self,a, b):
        return a // b
```

Suffix visible

```python
# Get input from user
a = int(input())
b = int(input())
# Create object for class Calculator
x = Calculator()
# Call method of x and print
print(x.Sum(a,b))
print(x.Multiply(a, b))
print(x.Subtraction(a, b))
print(x.Division(a, b))
print(x.Remainder(a, b))
print(x.Power(a, b))
print(x.Quotient(a, b))
```

Testcases

Public

Input

```
67
8
```

Output

```
75
536
59
8.375
3
406067677556641
8
```

Private

Input

```
253
5
```

Output

```
258
1265
248
50.6
3
1036579476493
50
```

Problem 2

Question

Create a class `StudentResult` based on the following table's data where column name represents the object's attribute in the class `StudentResult`. `Email_id` is an optional field (default value = None) and other fields are mandatory for each student:

Roll_no	Student_name	Math	Physics	Chemistry	Computer	English	Email_id
1001	Amit	60	70	60	55	75	amit@gmail.com

- Marks are out of 100

`StudentResultClass` has one method named `Display` that prints the data in the following format:

Input

```
s1=StudentResult(1001,'Amit',60,70,60,55,75,'amit@gmail.com')
s1.display()
```

Output

```
1001 Amit 60 70 60 55 75 amit@gmail.com
```

You only need to create the class, the object will be created internally to verify the answer.

Answer

```python
class StudentResult:
    # Create constructor for class StudentResult
    def __init__(self,Roll_no,Student_name,Math,Physics,Chemistry,Computer,English,Email_id=None):
        # Assign parameters value to instance variable
        self.Roll_no = Roll_no
        self.Student_name = Student_name
        self.Email_id = Email_id
        self.Math = Math
        self.Physics = Physics
        self.Chemistry = Chemistry
        self.Computer = Computer
        self.English = English
    # Create method to print object variable value
    def Display(self):
        print(s1.Roll_no,s1.Student_name,s1.Math,s1.Physics,s1.Chemistry,s1.Computer,s1.English,s1.Email_id)

```

Suffix Code block(Hidden)

```
# Get input from user for object creation
a = input()
b = input()
c = input()
d = input()
e = input()
f = input()
g = input()
h = input()
# Create object for class StudentResult
s1=StudentResult(a,b,c,d,e,f,g,h)
s1.Display()
```

Testcases

Public

Input

```
1001
Amit
60
70
60
55
75
amit@gmail.com
```

Output

```
1001 Amit 60 70 60 55 75 amit@gmail.com
```

Private

Input 1

```
1002
Rahul
55
45
69
85
78
rahul@gmail.com
```

Output 1

```
1002 Rahul 55 45 69 85 78 rahul@gmail.com
```

Input 2

```
1003
Anjali
85
78
98
85
96
anjali@gmail.com
```

Output 2

```
1003 Anjali 85 78 98 85 96 anjali@gmail.com
```

Problem 3

Question

Create a class `StudentResult` based on the following table's data where column name represents the object's attribute in the class `StudentResult`. `Email_id` is an optional field (default value = None) and other fields are mandatory for each student. In addition, create a class variable `Count` that contains the total number of objects created and create the following methods inside the class:

- `Average_marks` : That returns the average marks of the student.
- `Total_marks`: That returns `total_marks` out of 500 of the student.
- `Max_marks`: That returns maximum marks of the student.
- `Min_marks`: That returns minimum marks of the student.

Roll_no	Student_name	Math	Physics	Chemistry	Computer	English	Email_id
1001	Amit	60	70	60	55	75	amit@gmail.com

- Marks are out of 100

Object creation format

```
s1=StudentResult(1001,'Amit',60,70,60,55,75,'amit@gmail.com')
```

Output:

```
Amit 320/500 64.0 75 55
Total Students = 1
```

- **You only need to create the class.Do not create an object for the class. It will be created internally to verify the answers.**

Answer

```python
class StudentResult:
    Count = 0
    # Create constructor for class StudentResult
    def __init__(self,Roll_no,Student_name,Math,Physics,Chemistry,Computer,English,Email_id=None):
        # Assign parameters value to instance variable
        self.Roll_no=Roll_no
        self.Student_name=Student_name
        self.Email_id=Email_id
        self.Math=Math
        self.Physics=Physics
        self.Chemistry=Chemistry
        self.Computer=Computer
        self.English=English
        StudentResult.Count += 1
    # Create Total_marks method
    def Total_marks(self):
        return(str((self.Math+self.Physics+self.Chemistry+self.Computer+self.English))+'/500')
     # Create Average_marks method
    def Average_marks(self):
        return(str((self.Math+self.Physics+self.Chemistry+self.Computer+self.English)/5))
    # Create Max_marks method
    def Max_marks(self):
        return(max(self.Math,self.Physics,self.Chemistry,self.Computer,self.English))
    # Create Max_marks method
    def Min_marks(self):
        return(min(self.Math,self.Physics,self.Chemistry,self.Computer,self.English))
```

Suffix Code block(Hidden)

```python
# Get input from user for object creation
a=input()
b=input()
c=int(input())
d=int(input())
e=int(input())
f=int(input())
g=int(input())
h=input()
# Create object for StudentResult
s1=StudentResult(a,b,c,d,e,f,g,h)
# Call method of object s1 and print return value
print(s1.Student_name, s1.Total_marks(),s1.Average_marks(),s1.Max_marks(),s1.Min_marks())
# Call class variable count and print
print('Total Students =',StudentResult.Count)
```

Testcases

Public

Public

Input

```
1001
Amit
60
70
60
55
75
amit@gmail.com
```

Output

```
Amit 320/500 64.0 75 55
```

Private

Input 1

```
1002
Rahul
55
45
69
85
78
rahul@gmail.com
```

Output 1

```
Rahul 332/500 66.4 85 45
```

Input 2

```
1003
Anjali
85
78
98
85
96
anjali@gmail.com
```

Output 2

```
Anjali 442/500 88.4 98 78
```

Problem 4

Question

Create a class `StringManipulation` that receives a list of words `wlist` at the time of object creation. The class must have the following methods:

- `Words_of_length(length)` — returns a list of all the words of length `length` in `wlist`
- `Words_starts_with(char)` — returns a list of all the words that start with `char` in `wlist`
- `Words_ends_with(char)` — returns a list of all the words that end with `char` in `wlist`
- `Palindromes` — returns a list of all the words that are palindromes in `wlist`
- `Total_words` — returns the number of words in `wlist`
- `Longest_word` — that returns the longest length word in `wlist`. if list `wlist` has more than one longest word then return the first one.
- `Smallest_word` that returns the smallest length word in `wlist`. if list `wlist` has more than one smallest word then return the first one.
- `Count(word)` that returns the total number of occurrences of `word` in `wlist`

Answer

```python
class StringManipulation:
    # Create class constructor
    def __init__(self,wlist):
        # Assign input list data to object variable
        self.wlist=wlist[:]
    # Create class method Words_of_length
    def Words_of_length(self,length):
        # initialize empty list
        res=[]
        # Read all word from list
        for i in self.wlist:
            # Check length of each word is equal to 'length' value
            if len(i)==length:
                # Append word in res list
                res.append(i)
        return res
    # Create class method Words_starts_with
    def Words_starts_with(self,char):
        # Initialize empty list
        res=[]
        # Read all word from list
        for i in self.wlist:
            # Check first character of each word is equal to 'char' value
            if i[0]==char:
                # Append word in res list
                res.append(i)
        return res
    # Create class method Words_end_with
    def Words_end_with(self,char):
        # Initialize empty list
        res=[]
        # Read all word from list
        for i in self.wlist:
            # Check last character of each word is equal to 'char' value
            if i[-1]==char:
                # Append word in res list
                res.append(i)
        return res
    # Create class method Palindromes
    def Palindromes(self):
        # Initialize empty list
        res=[]
        # Read all word from list
        for i in self.wlist:
            # Check each word is equal to reverse of that word
            if i==i[::-1]:
                # Append word in res list
                res.append(i)
        return res
    # Create class method Total_words
    def Total_words(self):
        # Return length of list
        return len(self.wlist)
    # Create class method Longest_word
    def Longest_word(self):
        # Assume first word is maximum length word
        maxword = self.wlist[0]
        # Read all word from list one by one
        for i in self.wlist:
            # Check each word length is greater than to length of maxword
            if len(i)>len(maxword):
                # If yes then assign maxword to new word
                maxword = i
        return maxword
    # Create class method Smallest_word
    def Smallest_word(self):
        # Assume first word is minimum length word
        minword = self.wlist[0]
        # Read all word from list one by one
        for i in self.wlist:
            # Check each word length is smaller than to length of maxword
            if len(i)<len(minword):
                # If yes then assign minword to new word
                minword = i
        return minword
    # Create class method Count
    def Count(self,word):
        # Return count value of 'word' in list
        return self.wlist.count(word)
```

Suffix invisible

```python
# Get input from user and convert into list of word
word = input().split(' ')
# Create Object
s = StringManipulation(word)
# Call all method and print
print(s.Words_of_length(6))
print(s.Words_starts_with('s'))
print(s.Words_end_with('l'))
print(s.Palindromes())
print(s.Total_words())
print(s.Longest_word())
print(s.Smallest_word())
print(s.Count('it'))
```

Testcases

Public

Input

```
i hope not you might pull a muscle you need to start small in order to achieve something big like that when it comes to learning english what if i told you that you can understand big ideas with just a little bit of text you do not need to wait several years to deal with complex ideas just because you are learning a language does not mean you need to limit your thinking stories are all about going
```

Output

```
['muscle', 'little']
['start', 'small', 'something', 'several', 'stories']
['pull', 'small', 'several', 'deal', 'all']
['i', 'a', 'i', 'a', 'a']
79
understand
i
1
```

Private

Input

```
i hope not you might pull a muscle you need to start small in order to achieve something big like that when it comes to learning english what if i told you that you can understand big ideas with just a little bit of text you do not need to wait several years to deal with complex ideas just because you are learning a language does not mean you need to limit your thinking stories are all about going beyond reality it is no wonder that they let you understand big ideas with only a little bit of english reading practice
```

Output

```
['muscle', 'little', 'beyond', 'wonder', 'little']
['start', 'small', 'something', 'several', 'stories']
['pull', 'small', 'several', 'deal', 'all']
['i', 'a', 'i', 'a', 'a', 'a']
101
understand
i
2
```