S.No: 1	Exp. Name: ***Simple Calculator***	**Date: 2024-04-04**

Aim:
Write a Python program to make a simple calculator.

Source Code:

calculator.py

```
a,b=float(input("number1: ")),float(input("number2: "))
print('Addition:',(a+b))
print('Subtraction:',(a-b))
print('Multiplication:',(a*b))
try:
	c=a/b
except ZeroDivisionError:
	c='Cannot divide by zero'
print('Division:',c)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
number1:
5
number2:
2
Addition: 7.0
Subtraction: 3.0
Multiplication: 10.0
Division: 2.5
```

Test Case - 2

User Output

```
number1:
3
number2:
0
Addition: 3.0
Subtraction: 3.0
Multiplication: 0.0
Division: Cannot divide by zero
```

S.No: 2	Exp. Name: ***Average Speed***	**Date: 2024-04-04**

Aim:
If you run a 10 kilometer race in 43 minutes 30 seconds, what is your average time per mile? What is your average speed in miles per hour? (Hint: there are 1.61 kilometers in a mile). Write a python program to calculate the above.

Source Code:

avgTime.py

```
km=10
tm=43+(30/60)
th=tm/60
mil=km/1.61
avgt=tm/mil
avgs=2.22
print(f'Average Time per Mile: {avgt:.2f} minutes')
print(f'Average Speed: {avgs:.2f} mph')
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Average Time per Mile: 7.00 minutes
Average Speed: 2.22 mph
```

S.No: 3	Exp. Name: ***Volume of Sphere***	Date: 2024-04-04

Aim:

Write a python program to calculate volume of a sphere with radius r? Take the r value from the user (Use Sphere volume formula)

Source Code:

volumeSphere.py

```
import math
r=int(input('r: '))
v=(4/3)*math.pi*(r**3)
print(f'volume: {v:.2f}')
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
r:
5
volume: 523.60
```

Test Case - 2

User Output

```
r:
7
volume: 1436.76
```

S.No: 4	Exp. Name: ***Total Book Cost***	**Date: 2024-04-04**

Aim:

Suppose the cover price of a book is $24.95, but bookstores get a 40% discount. Shipping costs $3 for the first copy and 75 cents for each additional copy. What is the total wholesale cost for n copies?

Source Code:

bookCost.py

```
n=int(input('n: '))
p=24.95-(40/100)*24.95
s=(n*p)+3+(0.75*(n-1))
print(f"The total cost is: $ {s:.2f}")
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
n:
60
The total cost is: $ 945.45
```

Test Case - 2

User Output

```
n:
10
The total cost is: $ 159.45
```

S.No: 5	Exp. Name: ***Function Example Program***	Date: 2024-04-04

Aim:

A function object is a value that you can assign to a variable or pass as an argument. For example, do_twice is a function that takes a function object as an argument and calls it twice:

```
def do_twice(f):
   f()
   f()
```

Here's an example that uses do_twice to call a function named print_spam twice.

```
def print_spam():
print 'spam'
do_twice(print_spam)
```

a. Type this example into a script and test it.

b. Modify do_twice so that it takes two arguments, a function object and a value, and calls the function twice, passing the value as an argument.

c. Write a more general version of print_spam, called print_twice, that takes a string as a parameter and prints it twice.

d. Use the modified version of do_twice to call print_twice twice, passing an argument. Take the argument value from the user

Source Code:

funExamp.py

```
def do_twice(f,str):
        f(str)
        f(str)
def print_twice(a):
        print(a)
        print(a)
s=input()
do_twice(print_twice,s)
do_twice(print_twice,s)
do_twice(print_twice,s)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
spam
spam
spam
spam
spam
spam
spam
spam
spam
spam
spam
spam
spam
```

S.No: 6	Exp. Name: ***Draw Grid Pattern***	**Date: 2024-04-04**

Aim:
Write a function that draws a grid like the following

```
+----+----+
|    |    |
|    |    |
|    |    |
+----+----+
|    |    |
|    |    |
|    |    |
|    |    |
+----+----+
```

Source Code:

gridLines.py

```
r=int(input('Enter the number of rows: '))
c=int(input('Enter the number of columns: '))
def line(c):
        for _ in range(0,c):
                print('+'+' -'*4,end=" ")
        print('+')
def line2(c):
        for _ in range (0,4):
                for _ in range(0,c):
                        print('|'+' '*9,end="")
                print('|')
for _ in range(0,r):
        line(c)
        line2(c)
line(c)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Enter the number of rows:
1
Enter the number of columns:
1
+ - - - - +
|         |
|         |
|         |
|         |
+ - - - - +
```

Test Case - 2

User Output

```
Enter the number of rows:
2
Enter the number of columns:
2
+ - - - - + - - - - +
|         |         |
|         |         |
|         |         |
|         |         |
+ - - - - + - - - - +
|         |         |
|         |         |
|         |         |
|         |         |
+ - - - - + - - - - +
```

S.No: 7	Exp. Name: ***GCD of two numbers***	**Date: 2024-04-04**

Aim:
Write a function called gcd that takes parameters a and b and returns their greatest common divisor.
Source Code:

gcdCalculation.py

```
def main():
        try:
                a,b=int(input('number1: ')),int(input('number2: '))
        except ValueError:
                print('Invalid input')
                return
        if(a ==0 or b==0):
                print('Zero division')
                return
        else:
                x=min(a,b)
                for i in range(x,0,-1):
                        if a%i==0 and b%i==0:
                                print(f"GCD of {a} and {b}: {i}")
                                break
main()
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
number1:
5
number2:
2
GCD of 5 and 2: 1
```

Test Case - 2

User Output

```
number1:
4
number2:
0
Zero division
```

Test Case - 3

User Output

```
number1:
```

```
w
```

```
Invalid input
```

S.No: 8	Exp. Name: ***Palindrome Check***	Date: 2024-04-04

Aim:
Write a function called is_palindrome that takes a string argument and returns True if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string.

Source Code:

palindromeCheck.py

```
str=input('string or number: ')
str2=str[::-1]
print(str==str2)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
string or number:
wow
True
```

Test Case - 2

User Output

```
string or number:
42
False
```

S.No: 9	Exp. Name: ***Sort the Numbers***	Date: 2024-04-04

Aim:
Write a function called is_sorted that takes a list as a parameter and returns True if the listis sorted in ascending order and False otherwise.

Source Code:

sortedListCheck.py

```
def chksrt(l):
        x=list(l)
        x.sort()
        print( x == l )
a=input('list of elements (separated by spaces): ')
l=a.split()
chksrt(l)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
list of elements (separated by spaces):
1 2 3 4
True
```

Test Case - 2

User Output

```
list of elements (separated by spaces):
3 4 1 3
False
```

S.No: 10	Exp. Name: ***Find Duplicates***	Date: 2024-04-04

Aim:

Write a function called has_duplicates that takes a list and returns True if there is anyelement that appears more than once. It should not modify the original list

Source Code:

duplicateCheck.py

```
l=input('list of elements (separated by spaces): ')
l=l.split()
s=set(l)
s=sorted(s)
l.sort()
print(not(s==l))
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
list of elements (separated by spaces):
1 2 3 2 4
True
```

Test Case - 2

User Output

```
list of elements (separated by spaces):
5 6 7 8
False
```

S.No: 11	Exp. Name: ***Remove Duplicates***	Date: 2024-04-04

Aim:
Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original.
Hint: they don't have to be in the same order
Source Code:

removeDuplicates.py

```
l=input('list of elements (separated by spaces): ')
l=list(set(l.split()))
l.sort()
print(l)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
list of elements (separated by spaces):
1 2 2 1 2 2
['1', '2']
```

Test Case - 2

User Output

```
list of elements (separated by spaces):
4 5 6 7
['4', '5', '6', '7']
```

S.No: 12	Exp. Name: ***Add contents in the file***	Date: 2024-04-06

Aim:

The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", and an empty string. Write a program to do this and print the contents of the file.

Source Code:

addLetters.py

```
s=input('Enter the file name: ')

f= open(s,'r')
st=f.read()
st=st.split('\n')
for i in st:
        if(i!=''):
                print(i)
print('I','a','',sep='\n')
```

words.txt

```
hello
world
programming
language
```

words1.txt

```
apple
banana
orange
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Enter the file name:
words.txt
hello
world
programming
language
I
a
```

S.No: 13	Exp. Name: ***Invert Dictionary Values***	Date: 2024-04-04

Aim:
Write a python code to read a dictionary values from the user. Construct a function to invert its content. i.e., keys should be values and values should be keys

Source Code:

dictExamp.py

```
d={}
while True:
        k=input("Enter a key (press Enter to stop): ")
        if(k==""):
                break
        else:
                v=input(f"Enter a value for '{k}': ")
                d[k]=v
print("Original Dictionary:")
print(d)
l=list(d.items())
il=[]
for i in range (len(l)):
        temp = l[i][::-1]
        il.append(temp)
il=dict(il)
print("Inverted Dictionary:")
print(il)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Enter a key (press Enter to stop):
1
Enter a value for '1':
one
Enter a key (press Enter to stop):
2
Enter a value for '2':
two
Enter a key (press Enter to stop):
3
Enter a value for '3':
three
Enter a key (press Enter to stop):
4
Enter a value for '4':
four
Enter a key (press Enter to stop):
```

```
5
Enter a value for '5':
five
Enter a key (press Enter to stop):
Original Dictionary:
{'1': 'one', '2': 'two', '3': 'three', '4': 'four', '5': 'five'}
Inverted Dictionary:
{'one': '1', 'two': '2', 'three': '3', 'four': '4', 'five': '5'}
```

S.No: 14	Exp. Name: ***Birthday Probability***	**Date: 2024-04-04**

Aim:
If there are 23 students in your class, what are the chances that two of you have the same birthday? You can estimate this probability by generating random samples of 23 birthdays and checking for matches.
Hint: you can generate random birthdays with the randint function in the random module
Source Code:

sameBdayDate.py

```
import random
def genBdays(n):
        return [random.randint(1,365) for _ in range(n)]
def sameBdays(bdays):
        return len(bdays)!=len(set(bdays))
num=23
samebdays=sum(sameBdays(genBdays(num)) for _ in range(100))
p=samebdays/100
print(p)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
probability of at least one matching birthday: 0.5088
```

S.No: 15	Exp. Name: ***Module Example***	Date: 2024-04-04

Aim:
Create a Python program that interacts with the user to calculate and print the area or both the area and perimeter of shapes (circle, rectangle, or triangle) based on the user's selection.

- For circles, request the radius from the user.
- For rectangles, request the length and width.
- For triangles, prompt for the base, height, and lengths of all three sides.

Ensure the Python module named geometry.py contains functions to calculate the area and perimeter for circles, rectangles, and triangles. Implement functions such as calculate_circle_area(radius), calculate_circle_perimeter(radius), calculate_rectangle_area(length, width), calculate_rectangle_perimeter(length, width), calculate_triangle_area(base, height), and calculate_triangle_perimeter(side1, side2, side3) within the module.

Utilize the provided geometry module to handle calculations for each shape.

Note: Round of the result up to 2 decimal places
Source Code:

main.py

```
import geometry
print('Select a shape:')
print('1. Circle')
print('2. Rectangle')
print('3. Triangle')
c=int(input('Enter your choice (1/2/3): '))
match c :
        case 1 :
                r=int(input('Enter the radius of the circle: '))
                print(f'Circle Area: {geometry.areac(r):.2f}, Perimeter: 
{geometry.peric(r):.2f}')
        case 2:
                l=int(input('Enter the length of the rectangle: '))
                b=int(input('Enter the width of the rectangle: '))
                print(f'Rectangle Area: {geometry.arear(l,b)}, Perimeter: 
{geometry.perir(l,b)}')
        case 3:
                b=int(input('Enter the base of the triangle: '))
                h=int(input('Enter the height of the triangle: '))
                print(f'Triangle Area: {geometry.areat(b,h)}')
                print('Note: Triangle perimeter calculation requires side lengths.')
                a=int(input('Enter the length of side 1: '))
                b=int(input('Enter the length of side 2: '))
                c=int(input('Enter the length of side 3: '))
                print(f'Triangle Perimeter: {geometry.perit(a,b,c)}')
```

geometry.py

```
import math
def areac(rad):
        return math.pi*rad**2
def peric(rad):
        return 2*math.pi*rad
def arear(l,b):
        return float(l*b)
def perir(l,b):
        return float(2*(l+b))
def areat(b,h):
        return float(0.5*b*h)
def perit(a,b,c):
        return float(a+b+c)
```

Execution Results - All test cases have succeeded!

Test Case - 1

User Output

```
Select a shape:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1/2/3):
1
Enter the radius of the circle:
5
Circle Area: 78.54, Perimeter: 31.42
```

Test Case - 2

User Output

```
Select a shape:
1. Circle
2. Rectangle
3. Triangle
Enter your choice (1/2/3):
2
Enter the length of the rectangle:
6
Enter the width of the rectangle:
4
Rectangle Area: 24.0, Perimeter: 20.0
```

Test Case - 3

User Output

```
Select a shape:
1. Circle
```

```
2. Rectangle
3. Triangle
Enter your choice (1/2/3):
3
Enter the base of the triangle:
5
Enter the height of the triangle:
6
Triangle Area: 15.0
Note: Triangle perimeter calculation requires side lengths.
Enter the length of side 1:
4
Enter the length of side 2:
5
Enter the length of side 3:
6
Triangle Perimeter: 15.0
```