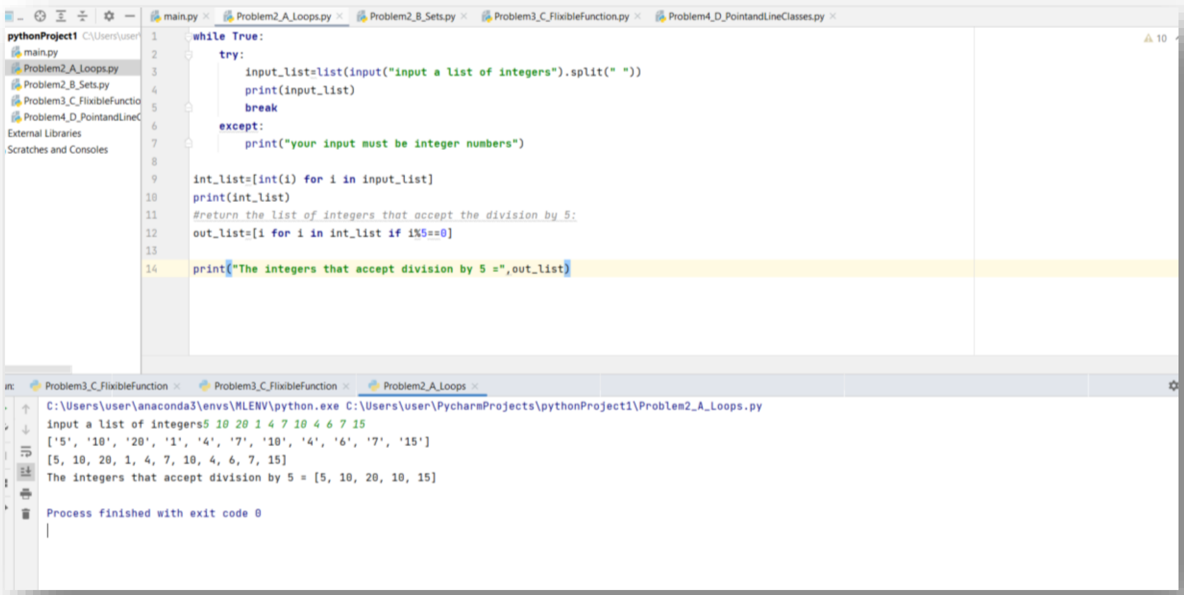
HomeWork_2 (02/Nov./2022)

1. Problem1_A_Loops.py

```python
while True:
    try:
        input_list=list(input("input a list of integers").split(" "))
        print(input_list)
        break
    except:
        print("your input must be integer numbers")

int_list=[int(i) for i in input_list]
print(int_list)
#return the list of integers that accept the division by 5:
out_list=[i for i in int_list if i%5==0]

print("The integers that accept division by 5 =",out_list)
```



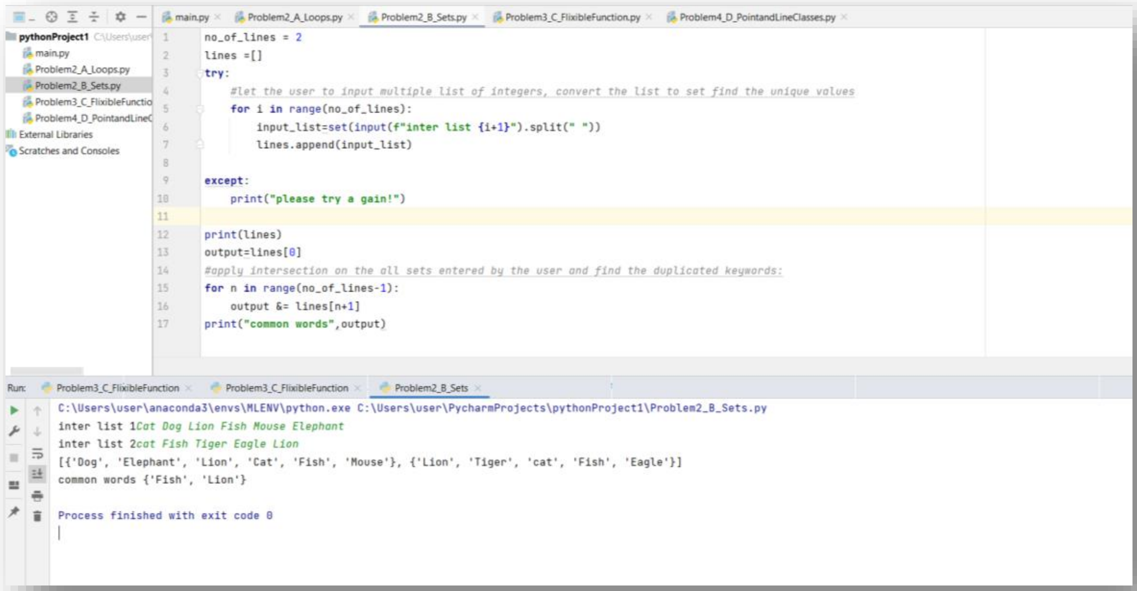
```python
while True:
    try:
        input_list=list(input("input a list of integers").split(" "))
        print(input_list)
        break
    except:
        print("your input must be integer numbers")

int_list=[int(i) for i in input_list]
print(int_list)
#return the list of integers that accept the division by 5:
out_list=[i for i in int_list if i%5==0]

print("The integers that accept division by 5 =",out_list)
```

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem2_A_Loops.py
input a list of integers5 10 20 1 4 7 10 4 6 7 15
['5', '10', '20', '1', '4', '7', '10', '4', '6', '7', '15']
[5, 10, 20, 1, 4, 7, 10, 4, 6, 7, 15]
The integers that accept division by 5 = [5, 10, 20, 10, 15]

Process finished with exit code 0
```

Figure 1: Output Sample of Problem1_A

2. Problem2_B_Sets.py

```python
no_of_lines = 2
lines =[]
try:
    #let the user to input multiple list of integers, convert the list
to set find the unique values
    for i in range(no_of_lines):
        input_list=set(input(f"inter list {i+1}").split(" "))
        lines.append(input_list)

except:
    print("please try a gain!")

print(lines)
output=lines[0]
#apply intersection on the all sets entered by the user and find the
duplicated keywords:
for n in range(no_of_lines-1):
    output &= lines[n+1]
print("common words",output)
```



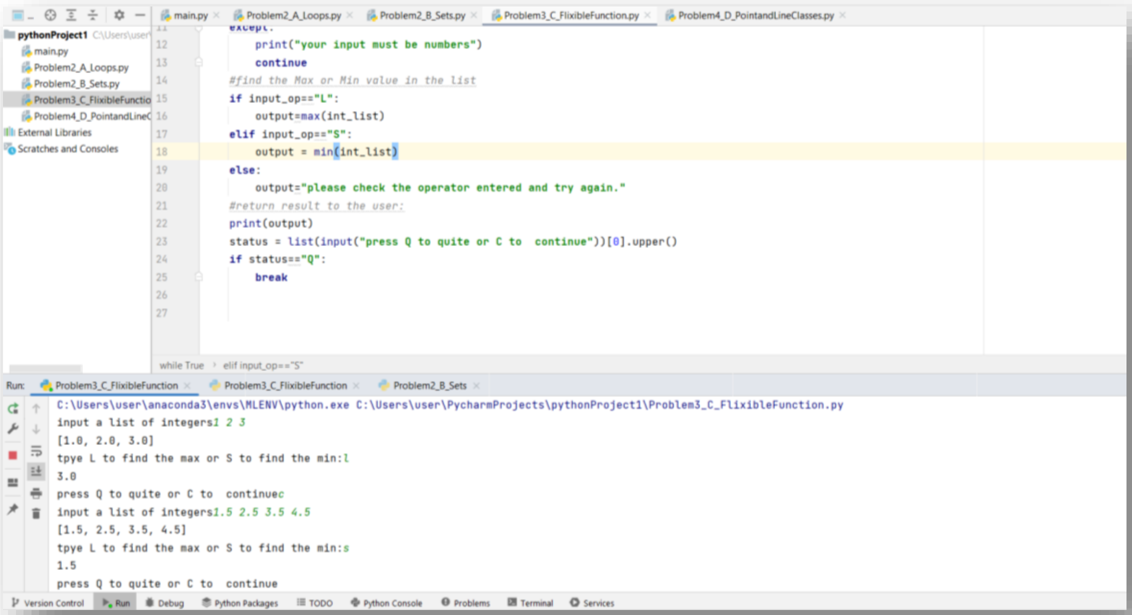
```python
no_of_lines = 2
lines =[]
try:
    #let the user to input multiple list of integers, convert the list to set find the unique values
    for i in range(no_of_lines):
        input_list=set(input(f"inter list {i+1}").split(" "))
        lines.append(input_list)

except:
    print("please try a gain!")

print(lines)
output=lines[0]
#apply intersection on the all sets entered by the user and find the duplicated keywords:
for n in range(no_of_lines-1):
    output &= lines[n+1]
print("common words",output)
```

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem2_B_Sets.py
inter list 1Cat Dog Lion Fish Mouse Elephant
inter list 2cat Fish Tiger Eagle Lion
[{'Dog', 'Elephant', 'Lion', 'Cat', 'Fish', 'Mouse'}, {'Lion', 'Tiger', 'cat', 'Fish', 'Eagle'}]
common words {'Fish', 'Lion'}

Process finished with exit code 0
```

Figure 2:Output Sample of Problem2_B

3. Problem3_C_FlixibleFunction.py

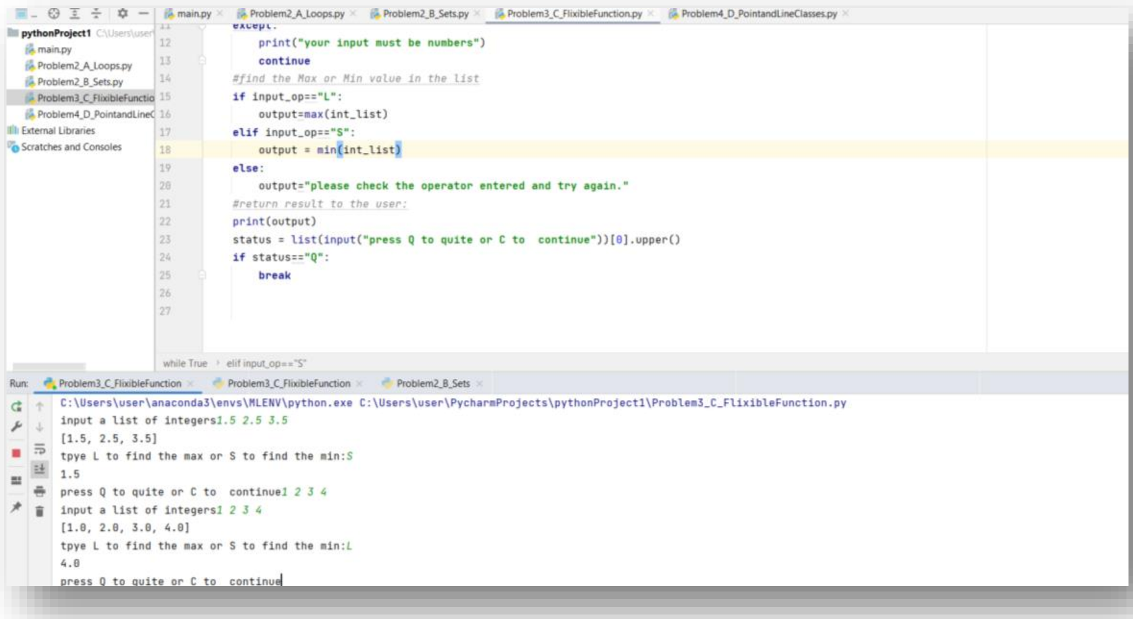
```python
while True:
    try:
        #let the user to input the list of integers:
        input_list=list(input("input a list of integers").split(" "))
        #convert the list of strings to list of floats:
        int_list = [float(i) for i in input_list]
        print(int_list)
        #let the user to choose the operant L,Large,large,l=Max and 
S,Small,small,s=Min:
        input_op=input("tpye L to find the max or S to find the 
min:")[0].upper()

    except:
        print("your input must be numbers")
        continue
    #find the Max or Min value in the list
    if input_op=="L":
        output=max(int_list)
    elif input_op=="S":
        output = min(int_list)
    else:
        output="please check the operator entered and try again."
    #return result to the user:
    print(output)
    status = list(input("press Q to quite or C to  continue"))[0].upper()
    if status=="Q":
        break
```



```python
    except:
        print("your input must be numbers")
        continue
    #find the Max or Min value in the list
    if input_op=="L":
        output=max(int_list)
    elif input_op=="S":
        output = min(int_list)
    else:
        output="please check the operator entered and try again."
    #return result to the user:
    print(output)
    status = list(input("press Q to quite or C to  continue"))[0].upper()
    if status=="Q":
        break
```

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem3_C_FlixibleFunction.py
input a list of integers1 2 3
[1.0, 2.0, 3.0]
tpye L to find the max or S to find the min:l
3.0
press Q to quite or C to  continuec
input a list of integers1.5 2.5 3.5 4.5
[1.5, 2.5, 3.5, 4.5]
tpye L to find the max or S to find the min:s
1.5
press Q to quite or C to  continue
```

Figure 3:Output Sample of Problem3_C (Case_0)



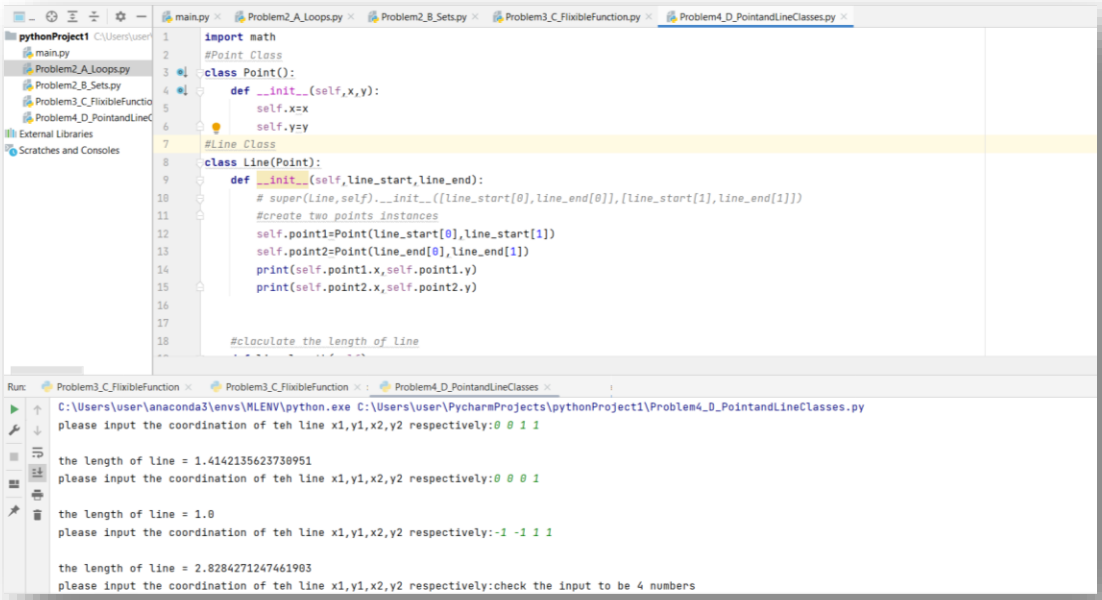
```
    except:
        print("your input must be numbers")
        continue
    #find the Max or Min value in the list
    if input_op=="L":
        output=max(int_list)
    elif input_op=="S":
        output = min(int_list)
    else:
        output="please check the operator entered and try again."
    #return result to the user:
    print(output)
    status = list(input("press Q to quite or C to  continue"))[0].upper()
    if status=="Q":
        break
```

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem3_C_FlixibleFunction.py
input a list of integers1.5 2.5 3.5
[1.5, 2.5, 3.5]
tpye L to find the max or S to find the min:S
1.5
press Q to quite or C to  continue1 2 3 4
input a list of integers1 2 3 4
[1.0, 2.0, 3.0, 4.0]
tpye L to find the max or S to find the min:L
4.0
press Q to quite or C to  continue
```

Figure 4: :Output Sample of Problem3_C (Case_1)

4. **Problem4_D_PointandLineClasses.py**

```
import math
#Point Class
class Point():
    def __init__(self,x,y):
        self.x=x
        self.y=y
#Line Class
class Line(Point):
    def __init__(self,line_start,line_end):
        #
super(Line,self).__init__([line_start[0],line_end[0]],[line_start[1],line_end
[1]])
        #create two points instances
        self.point1=Point(line_start[0],line_start[1])
        self.point2=Point(line_end[0],line_end[1])
        print(self.point1.x,self.point1.y)
        print(self.point2.x,self.point2.y)


    #claculate the length of line
    def line_length(self):
        length=math.sqrt((self.point2.x-self.point1.x)**2+(self.point2.y-
self.point1.y)**2)
        return length
while True:
    try:
        #take the coordinates from user as input
        input_coordinate=list(input("please input the coordination of teh
line x1,y1,x2,y2 respectively:").split(" "))
        #check the length of input values(must be 4)
        if len(input_coordinate)<4:
            print("please input 4 values of type number")
            continue
        #if the input values more that 4 elements it takes the first 4
elements:
        if len(input_coordinate)>=4:
            input_coordinate=input_coordinate[0:4]
        #convert the list of strings to list of floats:
        input_coordinate=[float(i) for i in input_coordinate]
    except:
        print("check the input to be 4 numbers")
        continue
    #create new line instance from the line class and find the length by
calling line_lingth function.

new_line=Line([input_coordinate[0],input_coordinate[1]],[input_coordinate[2],
input_coordinate[3]])
    length=new_line.line_length()
    print("the length of line =",length)
```


main.py
Problem2_A_Loops.py
Problem2_B_Sets.py
Problem3_C_FlixibleFunction.py
Problem4_D_PointandLineClasses.py
pythonProject1 C:\Users\user
External Libraries
Scratches and Consoles
Run:
Problem3_C_FlixibleFunction
Problem4_D_PointandLineClasses

```python
import math
#Point Class
class Point():
    def __init__(self,x,y):
        self.x=x
        self.y=y
#Line Class
class Line(Point):
    def __init__(self,line_start,line_end):
        # super(Line,self).__init__([line_start[0],line_end[0]],[line_start[1],line_end[1]])
        #create two points instances
        self.point1=Point(line_start[0],line_start[1])
        self.point2=Point(line_end[0],line_end[1])
        print(self.point1.x,self.point1.y)
        print(self.point2.x,self.point2.y)


    #claculate the length of line
```

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem4_D_PointandLineClasses.py
please input the coordination of teh line x1,y1,x2,y2 respectively:0 0 1 1

the length of line = 1.4142135623730951
please input the coordination of teh line x1,y1,x2,y2 respectively:0 0 0 1

the length of line = 1.0
please input the coordination of teh line x1,y1,x2,y2 respectively:-1 -1 1 1

the length of line = 2.8284271247461903
please input the coordination of teh line x1,y1,x2,y2 respectively:check the input to be 4 numbers
```

Figure 5: :Output Sample of Problem4_D