

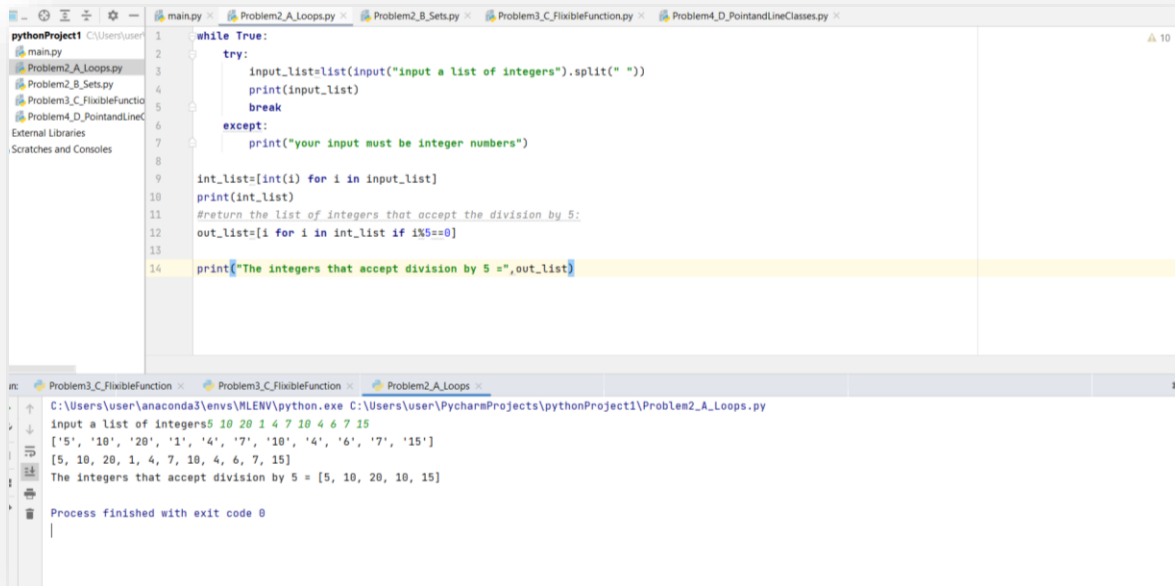
## HomeWork\_2 (02/Nov./2022)

### 1. Problem1\_A\_Loops.py

```
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)
```



The screenshot shows the PyCharm IDE with the file `Problem1_A_Loops.py` open. The code is as follows:

```
1 while True:
2     try:
3         input_list=list(input("input a list of integers").split(" "))
4         print(input_list)
5         break
6     except:
7         print("your input must be integer numbers")
8
9 int_list=[int(i) for i in input_list]
10 print(int_list)
11 #return the list of integers that accept the division by 5:
12 out_list=[i for i in int_list if i%5==0]
13
14 print("The integers that accept division by 5 =",out_list)
```

The console output shows the following sequence of events:

```
C:\Users\user\anaconda3\envs\MLENV\python.exe C:\Users\user\PycharmProjects\pythonProject1\Problem2_A_Loops.py
input a list of integers 5 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
```

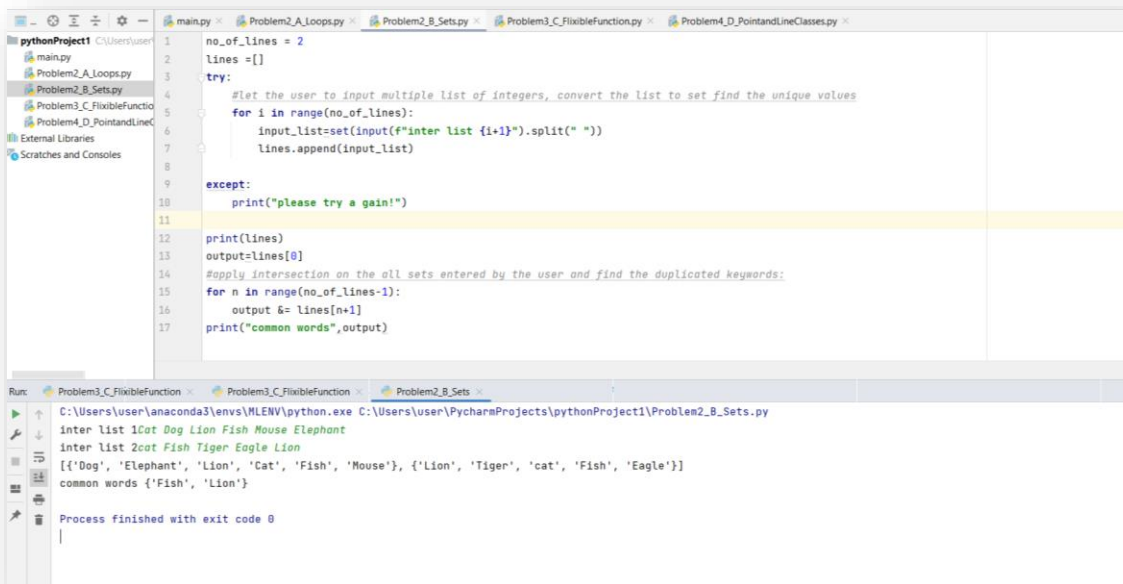
Figure 1: Output Sample of Problem1\_A

## 2. Problem2\_B\_Sets.py

```
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)
```



```
pythonProject1 C:\Users\user\
main.py Problem2_A_Loops.py Problem2_B_Sets.py Problem3_C_FixibleFunction.py Problem4_D_PointandLineClasses.py
1 no_of_lines = 2
2 lines = []
3 try:
4     #let the user to input multiple list of integers, convert the list to set find the unique values
5     for i in range(no_of_lines):
6         input_list=set(input(f"inter list {i+1}").split(" "))
7         lines.append(input_list)
8
9 except:
10    print("please try a gain!")
11
12 print(lines)
13 output=lines[0]
14 #apply intersection on the all sets entered by the user and find the duplicated keywords:
15 for n in range(no_of_lines-1):
16     output &= lines[n+1]
17 print("common words",output)

Run: Problem3_C_FixibleFunction Problem3_C_FixibleFunction Problem2_B_Sets
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\_FlexibleFunction.py

```
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
```

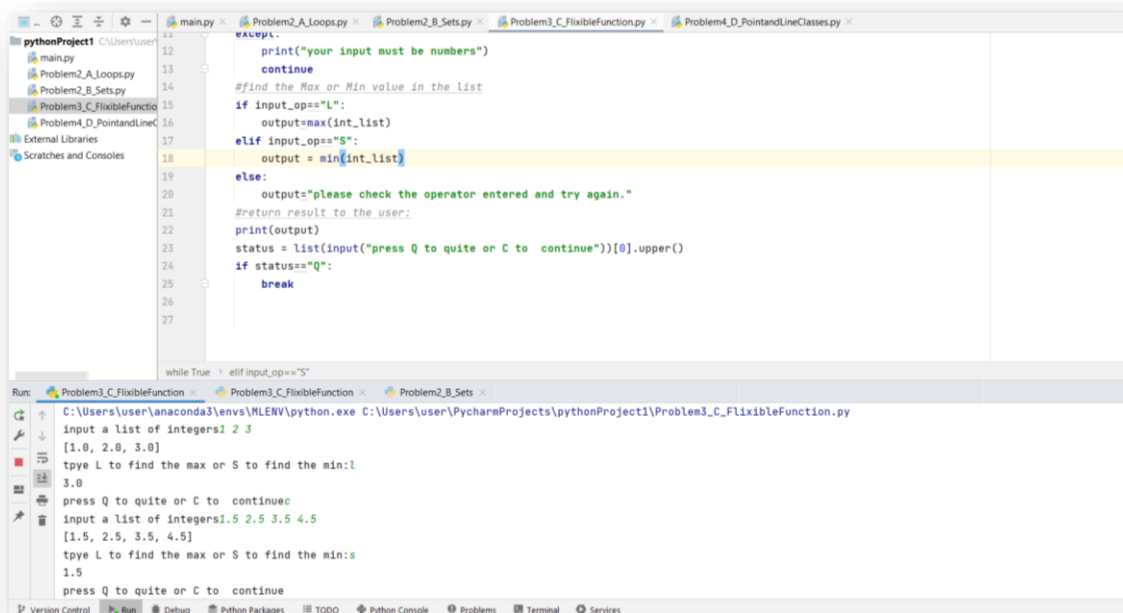


Figure 3:Output Sample of Problem3\_C (Case\_0)

The screenshot displays the PyCharm IDE interface. The top pane shows the source code for `main.py` (lines 12-27). The code is a loop that prompts the user for a list of integers and an operator ('L' for max, 'S' for min). It uses `try` and `except` blocks to handle non-integer inputs. The bottom pane shows the Run console output for `Problem3_C_FlexibleFunction.py`, which matches the sample output in the caption.

```
12 except:
13     print("your input must be numbers")
14     continue
15 #find the Max or Min value in the list
16 if input_op=="L":
17     output=max(int_list)
18 elif input_op=="S":
19     output = min(int_list)
20 else:
21     output="please check the operator entered and try again."
22 #return result to the user:
23 print(output)
24 status = list(input("press Q to quite or C to continue"))[0].upper()
25 if status=="Q":
26     break
27
```

Run: Problem3\_C\_FlexibleFunction.py C:\Users\user\PycharmProjects\pythonProject1\Problem3\_C\_FlexibleFunction.py

input a list of integers: 1.5 2.5 3.5  
[1.5, 2.5, 3.5]  
type L to find the max or S to find the min: S  
1.5  
press Q to quite or C to continue: 2 3 4  
input a list of integers: 2 3 4  
[1.0, 2.0, 3.0, 4.0]  
type 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)
```

```
1 import math
2 #Point Class
3 class Point():
4     def __init__(self,x,y):
5         self.x=x
6         self.y=y
7 #Line Class
8 class Line(Point):
9     def __init__(self,line_start,line_end):
10         # super(Line,self).__init__([line_start[0],line_end[0]],[line_start[1],line_end[1]])
11         #create two points instances
12         self.point1=Point(line_start[0],line_start[1])
13         self.point2=Point(line_end[0],line_end[1])
14         print(self.point1.x,self.point1.y)
15         print(self.point2.x,self.point2.y)
16
17
18 #calculate the length of line
```

Run: 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