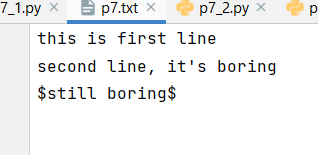
**-: Practical Set – 7 :-**

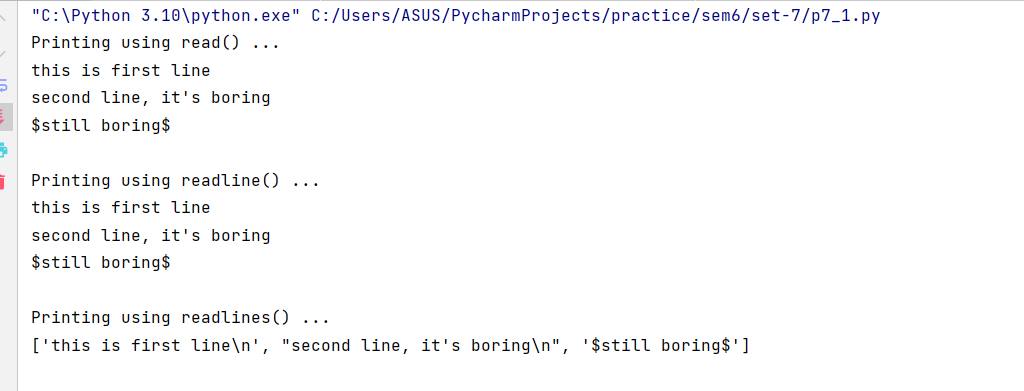
1. Write a python program to read the text file using read (), readlines() and readline() methods.

with open('p7.txt', 'r') as f:  
 print("Printing using read() ... ")  
 print(f.read())  
with open('p7.txt', 'r') as f:  
 print("\nPrinting using readline() ...")  
 while True:  
 line = f.readline()  
 print(line, end='')  
 if not line:  
 break  
with open('p7.txt', 'r') as f:  
 print("\n\nPrinting using readlines() ...")  
 line = f.readlines()  
 print(line)

**File Data :**



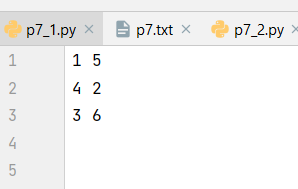
**OUTPUT :**



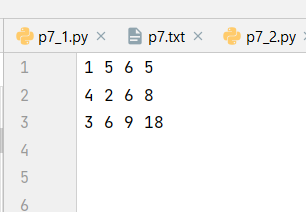
1. Write a python program to read a file containing pairs of numbers in a file. Create a file that contains the pairs of numbers as well as addition and multiplication of the two numbers in the same line.

with open("p7\_2", "r") as input\_file:  
 with open("output", "w") as output\_file:  
 for line in input\_file:  
 a, b = map(int, line.strip().split())  
 add = a + b  
 mul = a \* b  
 output\_file.write(f"{a} {b} {add} {mul}\n")

**File Data :**



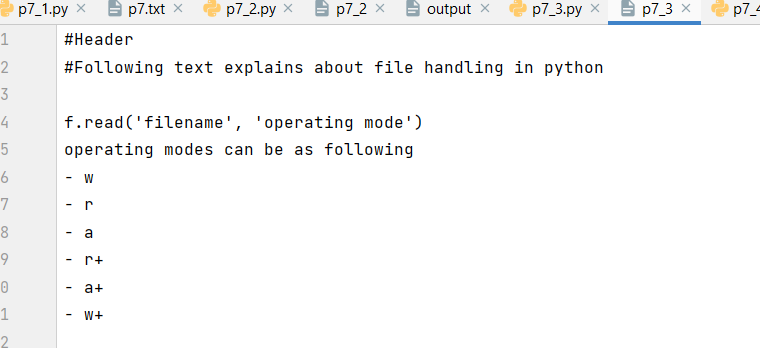
**OUTPUT:**



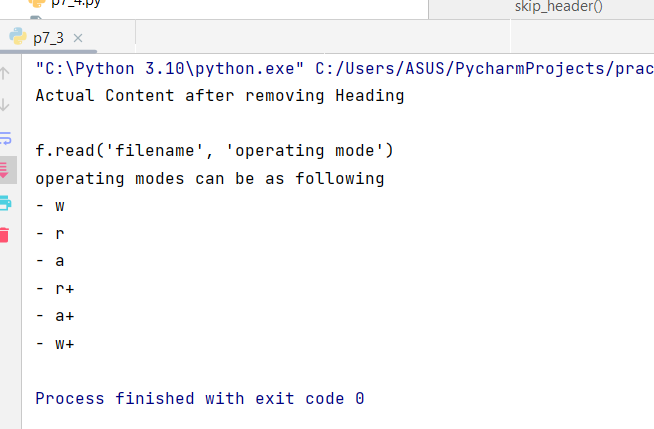
1. A text file contains a header line, few comments lines followed by actual lines of data. Write a python program to create a function skip\_header() that skips the header and all the comment lines and prints only actual lines of data.

def skip\_header(filename):  
 print("Actual Content after")  
 with open(filename, "r") as f:  
 f.readline()  
 while True:  
 line = f.readline()  
 if not line.startswith("#"):  
 break  
 while line:  
 print(line.strip())  
 line = f.readline()  
skip\_header('p7\_3')

**File Data :**



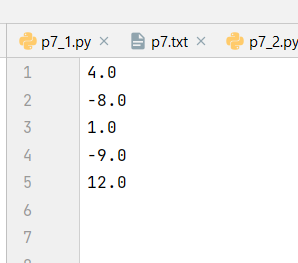
**OUTPUT:**



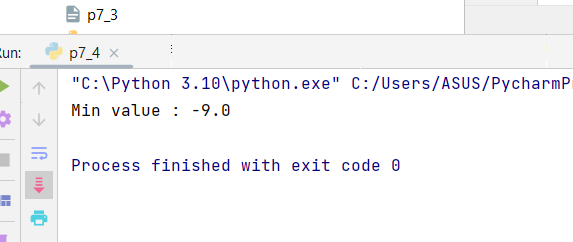
1. Write a python program to create a function that returns smallest value from the given text file.

def find\_smallest\_value(filename):  
 with open(filename, 'r') as file:  
 lines = file.readlines()  
 min\_value = None  
 for line in lines:  
 values = line.strip().split()  
 for value in values:  
 try:  
 value = float(value)  
 except ValueError:  
 continue  
 if min\_value is None or value < min\_value:  
 min\_value = value  
 return min\_value  
print(f"Min value : {find\_smallest\_value('p7\_4')}")

**File Data :**



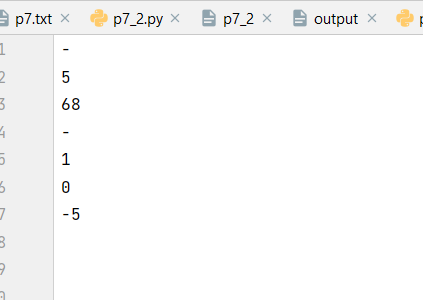
**OUTPUT:**



1. Write the program-4 for a text file with missing values (missing values are represented as hyphen (-)).

def find\_smallest\_value(filename):  
 with open(filename, 'r') as file:  
 lines = file.readlines()  
 max\_value = None  
 for line in lines:  
 values = line.strip().split()  
 for value in values:  
 if value == '-':  
 continue  
 try:  
 value = float(value)  
 except ValueError:  
 continue  
 if max\_value is None or value > max\_value:  
 max\_value = value  
 return max\_value  
print(find\_smallest\_value('p7\_5'))

**File Data :**



**OUTPUT:**

