-: 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 :

```
this is first line
second line, it's boring
$still boring$
```

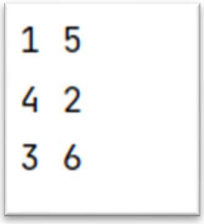
OUTPUT :

```
Printing using read() ...
this is first line
second line, it's boring
$still boring$

Printing using readline() ...
this is first line
second line, it's boring
$still boring$

Printing using readlines() ...
['this is first line\n', "second line, it's boring\n", '$still boring$']
```

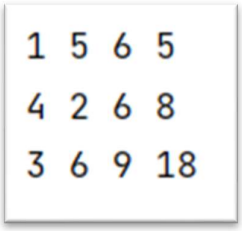
2. Write a python program to read a file containing pairs of numbers in a file. Create a file that contains the pairs of numbers as well as addition and multiplication of the two numbers in the same line.

```
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:



3. 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 :

```
#Header
#Following text explains about file handling in python

f.read('filename', 'operating mode')
operating modes can be as following
- w
- r
- a
- r+
- a+
- w+
```

OUTPUT:

```
Actual Content after removing Heading

f.read('filename', 'operating mode')
operating modes can be as following
- w
- r
- a
- r+
- a+
- w+
```

4. 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 :

```
4.0
-8.0
1.0
-9.0
12.0
```

OUTPUT:

```
Min value : -9.0
```

5. 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 :

```
-
5
68
-
1
0
-5
```

OUTPUT:

```
Min value : -5.0
```