1a. To check if a number is even or odd

```
a = int(input("Enter a number :"))

if a%2 == 0 :
    print(a,'is an even number')
else :
    print(a,"is an odd number")
```

```
Enter a number :12
12 is an even number
```

1b.Calculating factorial using recursion

```
def fact(n):
    return 1 if n==1 else n*fact(n-1)
a = int(input("Enter a number: "))
print(a,"!","=",fact(a))
```

```
Enter a number: 10
10 ! = 3628800
```

2.Removing duplicates from a List maintaining order

```
my_list = [1, 2, 2, 3, 4, 1, 5, 3, 6]
result = []
[result.append(item) for item in my_list if item not in result]
print(result)
```

```
[1, 2, 3, 4, 5, 6]
```

3.Removing duplicates from a List using sets

```
my_list = [1, 2, 2, 3, 4, 1, 5, 3, 6]
result = []
seen = set()
for item in my_list:
    if item not in seen:
        result.append(item)
        seen.add(item)

print(result)
```

```
[1, 2, 3, 4, 5, 6]
```

4.Calculate occurance of each character in a String

```
str = input("Enter a String :")
char_count_dic = {}

for char in str:
    if char in char_count_dic:
        char_count_dic[char] += 1
    else:
        char_count_dic[char] = 1

print(char_count_dic)
```

```
Enter a String :yappadoobee
{'y': 1, 'a': 2, 'p': 2, 'd': 1, 'o': 2, 'b': 1, 'e': 2}
```

5. Finding Index of an element in a tuple

```
tup = (1,7,67,45,47,5)
x = int(input("Enter element whose index is needed: "))

if x in tup:
    print(tup.index(x))
else:
    print("element is not in tuple")
```

```
Enter element whose index is needed: 7
1
```

6.Generate first N fibonacci number using both recursion and loop

```
def recursive_fib(n, a=0, b=1):
    if n == 0:
        return
    print(a, end=' ')
    recursive_fib(n - 1, b, a + b)


def loop_fib(n):
    a, b = 0, 1
    for _ in range(n):
        print(a, end=' ')
        a, b = b, a + b


n = 10
print("First Fibonacci numbers (using recursion): ")
recursive_fib(n)
print("\n-----------------------------------------------\nFirst
Fibonacci numbers (using loop): ")
loop_fib(n)
```

```
First Fibonacci numbers (using recursion):
0
1
1
2
3
5
8
13
21
34

--------------------------------------------------
First Fibonacci numbers (using loop):
0 1 1 2 3 5 8 13 21 34
```

7a. Largest of 3 numbers using if-else

```
a = int(input("Enter a number A :"))
b = int(input("Enter a number B :"))
c = int(input("Enter a number C :"))

if a>b:
    if a>c:
        max = c
    else:
        max = a
else:
    if b>c:
        max = b
    else:
        max = c
print(max,"is greatest among A,B,C")


Enter a number A :12
Enter a number B :12
Enter a number C :12
12 is greatest among A,B,C
```

7b. Print multiplication table of a given number

```
a = int(input("Enter a number :"))
q = int(input("Enter till what quotient you want the table :"))

for i in range(1,q+1):
    print(a,'x',q,'=',a*i)

Enter a number :2
Enter till what quotient you want the table :10
```

```
2 x 10 = 2
2 x 10 = 4
2 x 10 = 6
2 x 10 = 8
2 x 10 = 10
2 x 10 = 12
2 x 10 = 14
2 x 10 = 16
2 x 10 = 18
2 x 10 = 20
```

8. Write python code that prints number 1-50 but skips multiples of 5 using `continue`

```
for i in range(1,51):
    if i%5 == 0 :
        continue
    print(i)
```

```
1
2
3
4
6
7
8
9
11
12
13
14
16
17
18
19
21
22
23
24
26
27
28
29
31
32
33
34
36
37
```

```
38
39
41
42
43
44
46
47
48
49
```

9. Python program that stops execution using `break` when it finds a negative number in a list

```
list = []
n = int(input("Enter how many numbers you want in list "))

for i in range(n):
    num = int(input(f"Enter number {i + 1}: "))
    list.append(num)

for num in list:
    if num < 0:
        print("Negative number found:", num)
        break
    print("Current Number:", num)
```

```
Enter how many numbers you want in list 4
Enter number 1: 1
Enter number 2: 2
Enter number 3: 3
Enter number 4: -7
Current Number: 1
Current Number: 2
Current Number: 3
Negative number found: -7
```

10. Use `pass` statement inside a loop

```
for i in range(10):
    pass                    #used as a placeholder for future code
print("Code ended")
```

```
Code ended
```

Python Basics Over ~