Test V25.01.29

- Q1. Using Multiple Inheritance display Armstrong, Palindrome, Factorial, Fibonacci Series in each class indivisually.

In [1]:
```
# Armstrong
class Armstrong:
    def armstrong(self):
        num = int(input("Enter a Armstrong Number: "))
        temp = num
        sum = 0
        while(num>0):
            rem = num % 10
            sum = sum + (rem * rem *rem)
            num = num//10
        if sum == temp:
            print(sum," is Armstrong Number.")
        else:
            print(temp, "is not armstrong Nnumber.")

# Palindrome
class Palindrome():
    def palindrome(self):
        num = int(input("Enter a Palindrome Number: "))
        temp = num
        sum = 0
        while(num>0):
            rem = num % 10
            sum = (sum * 10) + rem
            num = num // 10
        if sum == temp:
            print(sum,"is Palindrome Number.")
        else:
            print(temp,"is not Palindrome Number.")

# Factorial
class Factorial():
    def factorial(self):
        num = int(input("Enter a Factorial Number: "))
        fact = 1
        for i in range(1,num+1):
            fact = fact*i
        print("Factorial of ",num,"is",fact)

# Fibonacci
class Fibonacci():
    def fibonacci(self):
        num = int(input("Enter a Fibonacci Series Range: "))
        temp = num
        p1 = 0
        p2 = 1
        print(p1," ",p2,end="  ")
        for i in range(2,num-1):
            p3 = p1 + p2
            print(p3," ",end="")
            p1 = p2
            p2 = p3
        if num == temp:
            print(num,"is Palindrom Number.")
        else:
            print(temp,"is not palindrome number.")

# Multilevel Inheritance
class Evaluate(Armstrong, Palindrome, Factorial, Fibonacci):
    def evaluate(self):
        super().fibonacci()
        super().factorial()
        super().palindrome()
        super().armstrong()

obj = Evaluate()
obj.evaluate()
```

```
0   1  1  2  5 is Palindrom Number.
Factorial of  5 is 120
121 is Palindrome Number.
153  is Armstrong Number.
```

- Q2. Using Multiple Inheritance Drow:
  - 1'st class: Pyramid
  - 2'nd class: Reverse Pyramid
  - 3'rd class: Diamond
  - 4'th class: Use Super Keyword

In [2]:

```
# Pyramid
class Pyramid:
    def pyramid(self):
        print("Pyramid: ")
        for i in range(1,6):
            for j in range(5,0,-1):
                if j>=i:
                    print(" ",end="")
            for k in range(1,6):
                if k<=i:
                    print("*",end="")
            for l in range(1,6):
                if l<=i-1:
                    print("*",end="")
            print()

# Reverse Pyramid
class ReversePyramid(Pyramid):
        def reversePyramid(self):
            super().pyramid()
            print()
            print("Reverse Pyramid: ")
            for i in range(1,6):
                for j in range(1,6):
                    if j<=i:
                        print(" ",end="")
                for k in range(5,0,-1):
                    if k>=i:
                        print("*",end="")
                for l in range(4,0,-1):
                    if l>=i:
                        print("*",end="")
                print()

# Diamond
class Diamond(ReversePyramid):
    def diamond(self):
        super().reversePyramid()
        print()
        print("Diamond: ")
        for i in range(1,6):
            for j in range(5,0,-1):
                if j>=i:
                    print(" ",end="")
            for k in range(1,6):
                if k<=i:
                    print("*",end="")
            for l in range(1,6):
                if l<=i-1:
                    print("*",end="")
            print()
        for i in range(1,5):
            for j in range(1,6):
                if j<=i+1:
                    print(" ",end="")
            for k in range(4,0,-1):
                if k>=i:
                    print("*",end="")
            for l in range(3,0,-1):
                if l>=i:
                    print("*",end="")
            print()

obj3 = Diamond()
obj3.diamond()
```

```
Pyramid:
     *
    ***
   *****
  *******
 *********

Reverse Pyramid:
 *********
  *******
   *****
    ***
     *

Diamond:
     *
    ***
   *****
  *******
 *********
  *******
   *****
    ***
     *
```

- `Using Multilevel Inheritance display Methods of String and List in each class. Use Super method to call the Methods.`

In [5]:
```python
# List:
# 1. `append:`
# 2. `extend:`
# 3. `insert:`
# 4. `remove:`
# 5. `pop:`
# 6. `index:`
# 7. `count:`
# 8. `reverse:`
# 9. `sort:` & `sort:` `reverse=True`
# 10. `clear:`
# 11.  `copy:`

# String:
# 1. isalnum
# 2. isalpha
# 3. isdigit
# 4. islower
# 5. isupper
# 6. Concatination
# 7. join
# 8. upper
# 9. lower
# 10. title
# 11. swapcase
# 12. replace
# 13. index
# 14. find

class ListStringMethods1:
    def method1(self):

        print("List Functions:")
        data1 = [1,2,3,4,5]
        data2 = [6,7,8,9,10]

        data1.append(data2)
        print(data1)

        data1.extend(data2)
        print(data1)

        data1.insert(10,11)
        print(data1)

        print()
        print("String Functions:")
        name = "Pratik"
        print(name.isalnum())
        print(name.isalpha())
        print(name.isdigit())

class ListStringMethods2(ListStringMethods1):
    def method2(self):
```

```
        super().method1()
        print()

        print("List Functions:")
        data1 = [1,2,3,4,5]
        data2 = [6,7,8,9,10]

        print(data1.pop())

        print(data1.index(3))

        print(data1.count(3))

        print()
        print("String Functions:")
        name = "Pratik"
        print(name.isupper())
        print(name.islower())
        print(name +" " + "Majage")

class ListStringMethods3(ListStringMethods2):
    def method3(self):

        super().method2()
        print()

        print("List Functions:")
        data1 = [1,2,3,4,5]

        data1.remove(1)
        print(data1)

        data1.reverse()
        print(data1)

        data1.clear()
        print(data1)

        print()
        print("String Functions:")
        name = "pratik"
        print(name.upper())
        print(name.lower())
        print(name.title())

class ListStringMethods4(ListStringMethods3):
    def method4(self):

        super().method3()
        print()

        print("List Functions:")
        data1 = [1,2,3,4,5]

        data1.sort()
        print(data1)

        data1.sort(reverse=True)
        print(data1)

        data2 = data1.copy()
        print(data2)

        print()
        print("String Functions:")
        q1 = "Pratik is bad boy"
        print(q1.replace("bad","good"))
        w1 = "aaBBccDD"
        print(w1.find("c"))
        print(w1.index("c"))

        # Additionals:
        name = "Pratik"
        string = "-"
        print(string.join(name))
        print(w1.swapcase())


obj = ListStringMethods4()
obj.method4()
```

```
List Functions:
[1, 2, 3, 4, 5, [6, 7, 8, 9, 10]]
[1, 2, 3, 4, 5, [6, 7, 8, 9, 10], 6, 7, 8, 9, 10]
[1, 2, 3, 4, 5, [6, 7, 8, 9, 10], 6, 7, 8, 9, 11, 10]

String Functions:
True
True
False

List Functions:
5
2
1

String Functions:
False
False
Pratik Majage

List Functions:
[2, 3, 4, 5]
[5, 4, 3, 2]
[]

String Functions:
PRATIK
pratik
Pratik

List Functions:
[1, 2, 3, 4, 5]
[5, 4, 3, 2, 1]
[5, 4, 3, 2, 1]

String Functions:
Pratik is good boy
4
4
P-r-a-t-i-k
AAbbCCdd
```