TYPES OF INHERITANCE PYTHON:

Inheritance is defined as the mechanism of inheriting the properties of the base class to the child class. Here we a going to see the types of inheritance in Python.

Types of inheritance Python

Types of Inheritance in Python

Types of Inheritance depend upon the number of child and parent classes involved. There are four types of inheritance in Python:

SINGLE INHERITANCE:

Single inheritance enables a derived class to inherit properties from a single parent class, thus enabling code reusability and the addition of new features to existing code.

Example:

```
# Python program to demonstrate
# single inheritance

# Base class
class Parent:
    def func1(self):
        print("This function is in parent class.")
```

```
# Derived class


class Child(Parent):
    def func2(self):
        print("This function is in child class.")



# Driver's code
object = Child()
object.func1()
object.func2()
```

Output:

```
This function is in parent class.
This function is in child class.
```

MULTIPLE INHERITANCE:

When a class can be derived from more than one base class this type of inheritance is called multiple inheritances. In multiple inheritances, all the features of the base classes are inherited into the derived class.

Example:

```
# Python program to demonstrate
# multiple inheritance
# Base class1
class Mother:
    mothername = ""

    def mother(self):
        print(self.mothername)

# Base class2
class Father:
    fathername = ""

    def father(self):
        print(self.fathername)

# Derived class


class Son(Mother, Father):
    def parents(self):
        print("Father :", self.fathername)
        print("Mother :", self.mothername)



# Driver's code
s1 = Son()
s1.fathername = "RAM"
```

```
s1.mothername = "SITA"

s1.parents()
```

Output:

```
Father : RAM

Mother : SITA
```

MULTILEVEL INHERITANCE :

In multilevel inheritance, features of the base class and the derived class are further inherited into the new derived class. This is similar to a relationship representing a child and a grandfather.

Example:

```
# Python program to demonstrate

# multilevel inheritance


# Base class



class Grandfather:


    def __init__(self, grandfathername):

        self.grandfathername = grandfathername


# Intermediate class
```

```
class Father(Grandfather):
    def __init__(self, fathername, grandfathername):
        self.fathername = fathername

        # invoking constructor of Grandfather class
        Grandfather.__init__(self, grandfathername)

# Derived class


class Son(Father):
    def __init__(self, sonname, fathername, grandfathername):
        self.sonname = sonname

        # invoking constructor of Father class
        Father.__init__(self, fathername, grandfathername)

    def print_name(self):
        print('Grandfather name :', self.grandfathername)
        print("Father name :", self.fathername)
        print("Son name :", self.sonname)



#  Driver code
```

```
s1 = Son('Prince', 'Rampal', 'Lal mani')

print(s1.grandfathername)

s1.print_name()
```

Output:

```
Lal mani

Grandfather name : Lal mani

Father name : Rampal

Son name : Prince
```

HIERARCHICAL INHERITANCE:

When more than one derived class are created from a single base this type of inheritance is called hierarchical inheritance. In this program, we have a parent (base) class and two child (derived) classes.

Example:

```
# Python program to demonstrate

# Hierarchical inheritance



# Base class

class Parent:

    def func1(self):

        print("This function is in parent class.")
```

```
# Derived class1



class Child1(Parent):

    def func2(self):

        print("This function is in child 1.")


# Derivied class2



class Child2(Parent):

    def func3(self):

        print("This function is in child 2.")



# Driver's code

object1 = Child1()

object2 = Child2()

object1.func1()

object1.func2()

object2.func1()

object2.func3()
```

Output:

```
This function is in parent class.
This function is in child 1.

This function is in parent class.

This function is in child 2.
```

Hybrid Inheritance:

Inheritance consisting of multiple types of inheritance is called hybrid inheritance.

Example:

```python
# Python program to demonstrate

# hybrid inheritance



class School:

    def func1(self):

        print("This function is in school.")



class Student1(School):

    def func2(self):

        print("This function is in student 1. ")
```

```
class Student2(School):

    def func3(self):

        print("This function is in student 2.")




class Student3(Student1, School):

    def func4(self):

        print("This function is in student 3.")




# Driver's code

object = Student3()

object.func1()

object.func2()
```

Output:

```
This function is in school.

This function is in student 1.
```