Inheritance

Inheritance in Python is a fundamental concept in Object-Oriented Programming (OOP) that allows one class (called the child class or derived class) to acquire properties and behaviors (methods) from another class (called the parent class or base class).

Why is inheritance useful?

- **Code Reusability:** Avoid rewriting common code.
- **Hierarchy:** Establish a relationship between classes.
- **Extensibility:** Add or override features in the child class.

In [23]:
```
# init method --> constructors
class Employee:
    amt=1.20
    def __init__(self,fname,lname,desig='',sal=0):
        self.fname=fname
        self.lname=lname
        self.desig=desig
        self.sal=sal
        self.email=self.fname.lower()+self.lname.lower()+'@zh.du.ac.in'
    def info(self):
        print('Name of Emp\t',self.fname+' '+self.lname)
        print('Designation \t',self.desig)
        print('Email \t\t',self.email)
        print('Salary \t\t',self.sal)
    def apply_raise(self):
        self.sal=self.sal*self.amt
        print('Salary after appraisal\t',self.sal)
```

In [24]:
```
# plan of action
#employee--> child(Developer,Manager)
```

In [25]:
```
#Method overiding
class Developer(Employee):
    #pass
    def __init__(self,fname,lname,desig,sal,lang):
        super().__init__(fname,lname,desig,sal)
        self.lang=lang
    def info(self):
        print('-'*60)
        super().info()
        print('Language\t',self.lang)
        print('-'*60)
```

In [26]:
```
dev1=Developer(fname='Vipul',lname='Pandey',desig='Developer',sal=100000,lang=
```

```
dev1.info()
```

```
-------------------------------------------------------------
Name of Emp         Vipul Pandey
Designation          Developer
Email                  vipulpandey@zh.du.ac.in
Salary                  100000
Language         Python
-------------------------------------------------------------
```

In [27]:
```
#manager

class Manager(Employee):
    def __init__(self,fname,lname,desig,sal,emp=None):
        super().__init__(fname,lname,desig,sal)
        self.emp = emp
        if emp==None:
            self.emp= []
        else:
            temp = None
            temp = self.emp
            self.emp = []
            self.emp.append(temp)
    def add_emp(self,cand):
        self.emp.append(cand)
        print(f'{cand} is successfully added to your list.')
    def remove_emp(self,cand):
        if cand not in self.emp:
            print(f'{cand} : Candidate not found.')
        else:
            self.emp.remove(cand)
            print(f'{cand} successfully removed.')
    def replace_emp(self,cand,replacement):
        if cand not in self.emp:
            print('Candidate not found')
        else:
            temp = self.emp.index(cand)
            self.emp[temp] = replacement
            print(f'successfully replaced {cand} with {replacement}')
    def show_all(self):
        count =1
        for i in self.emp:
            print(count,'.',i)
            count+=1
```

In [28]:
```
man1=Manager('Vipul','Pandey','Manager',100000,'Ram')
```

In [29]:
```
man1.emp
```

Out[29]:
```
['Ram']
```

In [30]:
```
man1.add_emp('Gaurav')
man1.add_emp('Raju')
```

```
man1.add_emp('Harsh')
```

```
Gaurav is successfully added to your list.
Raju is successfully added to your list.
Harsh is successfully added to your list.
```

In [31]:
```
man1.emp
```

Out[31]:
```
['Ram', 'Gaurav', 'Raju', 'Harsh']
```

In [32]:
```
man1.remove_emp('soorya')
man1.remove_emp('Harsh')
```

```
soorya : Candidate not found.
Harsh successfully removed.
```

In [33]:
```
man1.emp
```

Out[33]:
```
['Ram', 'Gaurav', 'Raju']
```

In [34]:
```
man1.replace_emp('Raju','Anshul')
```

```
successfully replaced Raju with Anshul
```

In [35]:
```
man1.emp
```

Out[35]:
```
['Ram', 'Gaurav', 'Anshul']
```

In [36]:
```
man1.show_all()
```

```
1 . Ram
2 . Gaurav
3 . Anshul
```

Types of Inheritance in Python:

- **Single Inheritance** – One child inherits from one parent.
- **Multiple Inheritance** – One child inherits from multiple parents.
- **Multilevel Inheritance** – Child -> Parent -> Grandparent.
- **Hierarchical Inheritance** – Multiple children inherit from one parent.
- **Hybrid Inheritance** – Combination of the above.

In [38]:
```
# single inheritance
class parent:
    print('parent')

class child(parent):
    pass
```

parent

In [39]:
```
# multiple inheritance

class papa:
    pass
class mummy:
    pass

class child(papa,mummy):
    pass

# help(child)
```

In [40]:
```
# mutlilevel inheritance

class grandparents:
    pass
class parents(grandparents):
    pass
class child(parents):
    pass
```

In [41]:
```
# Hierarchical Inheritance

class parent:
    pass

class child1(parent):
    pass
class child2(parent):
    pass
```

In [ ]: