## Simple

```
class Grandfather:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def show_details(self):
        return f'Name: {self.name}, Age: {self.age}'

    def speak(self):
        return 'Grandfather speaks wisely.'

class Father(Grandfather):
    def __init__(self, name, age, occupation):
        super().__init__(name, age)
        self.occupation = occupation

    def show_occupation(self):
        return f'Occupation: {self.occupation}'

    def speak(self):
        return 'Father speaks carefully.'



# Creating an object of the Father class
father_obj = Father('John', 50, 'Engineer')
print(father_obj.show_details())
print(father_obj.show_occupation())
print(father_obj.speak())
```

```
Name: John, Age: 50
Occupation: Engineer
Father speaks carefully.
```

Start coding or generate with AI.

```
# Heirarchial


class Grandfather:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def show_details(self):
        return f'Name: {self.name}, Age: {self.age}'

    def speak(self):
        return 'Grandfather speaks wisely.'

class Child1(Grandfather):
    def __init__(self, name, age, hobby):
        super().__init__(name, age)
        self.hobby = hobby

    def show_hobby(self):
        return f'Hobby: {self.hobby}'

    def speak(self):
        return 'Child1 speaks enthusiastically.'

class Child2(Grandfather):
    def __init__(self, name, age, favorite_subject):
        super().__init__(name, age)
        self.favorite_subject = favorite_subject

    def show_favorite_subject(self):
        return f'Favorite Subject: {self.favorite_subject}'

    def speak(self):
```

```
        return 'Child2 speaks thoughtfully.'

# Creating objects of Child1 and Child2 classes
child1_obj = Child1('Alice', 20, 'Painting')
child2_obj = Child2('Bob', 22, 'Mathematics')
print(child1_obj.show_details())
print(child1_obj.show_hobby())
print(child1_obj.speak())

print(child2_obj.show_details())
print(child2_obj.show_favorite_subject())
print(child2_obj.speak())
```

```
Name: Alice, Age: 20
Hobby: Painting
Child1 speaks enthusiastically.
Name: Bob, Age: 22
Favorite Subject: Mathematics
Child2 speaks thoughtfully.
```

Start coding or generate with AI.

```
# Multilevel Inheritance Example

class Grandfather:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def show_details(self):
        return f'Name: {self.name}, Age: {self.age}'

    def speak(self):
        return 'Grandfather speaks wisely.'

class Father(Grandfather):
    def __init__(self, name, age, occupation):
        super().__init__(name, age)
        self.occupation = occupation

    def show_occupation(self):
        return f'Occupation: {self.occupation}'

    def speak(self):
        return f'Father speaks wisely'

class Child(Father):
    def __init__(self, name, age, hobby):
        super().__init__(name, age, 'Engineer')
        self.hobby = hobby

    def show_hobby(self):
        return f'Hobby: {self.hobby}'

    def speak(self):
        return super().speak()

# Creating an object of the Child class
child_obj = Child('Charlie', 18, 'Cycling')
print(child_obj.show_details())
print(child_obj.show_occupation())
print(child_obj.show_hobby())
print(child_obj.speak())
```

```
Name: Charlie, Age: 18
Occupation: Engineer
Hobby: Cycling
Father speaks wisely
```

Start coding or generate with AI.

## Multiple

```
# Multiple Inheritance Example
```

```
# Multiple Inheritance Example

class Father:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def show_details(self):
        return f'Name: {self.name}, Age: {self.age}'


    # def show_favorite_food(self):
    #     return "Here"

    def speak(self):
        return 'Father speaks carefully.'

class Mother:
    def __init__(self, name, favorite_food):
        self.name = name
        self.favorite_food = favorite_food

    def show_favorite_food(self):
        return f'Favorite Food: {self.favorite_food}'

    def speak(self):
        return 'Mother speaks lovingly.'

class Child(Father,Mother):
    def __init__(self, name, age, favorite_food, hobby):
        Father.__init__(self, name, age)
        Mother.__init__(self, name, favorite_food)
        self.hobby = hobby

    def show_hobby(self):
        return f'Hobby: {self.hobby}'

    # def speak(self):
    #     return 'Child speaks excitedly.'

# Creating an object of the Child class
child_obj = Child('Daisy', 16, 'Pizza', 'Dancing')
print(child_obj.show_details())
print(child_obj.show_favorite_food())
print(child_obj.show_hobby())
print(child_obj.speak())
```

```
Name: Daisy, Age: 16
Favorite Food: Pizza
Hobby: Dancing
Father speaks carefully.
```

```
Child.mro()
```

```
[__main__.Child, __main__.Father, __main__.Mother, object]
```

Start coding or generate with AI.