SIMPLE PROGRAMS BY USING 00P:

```
p=int(input("enter principle amount"))
r=float(input("enter interst rate"))
t=float(input("enter the time period (in year)"))
simpleinterest=((p*r*t)/100)
print("simple interest is",simpleinterest)
```

Oop by using constructor:

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

    def greet(self):
        print("Hello, my name is " + self.name)

person1 = person("John", 36)
person1.greet()
```

Oop without using constructor

```
class person:
  def dhanu(self):
    print(f"my name is {self.name}")
person1=person()
person1.name="naadhi"
person1.dhanu()
```

Polymorphism program

```
class Shapes():
  def area(self):
    pass
class rectangle(Shapes):
  def __init__(self,length,breadth):
    self.length=length
    self.breadth=breadth
  def area(self):
    return self.length*self.breadth
class circle(Shapes):
  def __init__(self,radius):
   self.radius=radius
```

```
  def area(self):
    return 3.14*self.radius*self.radius
shapes=[rectangle(5,3),circle(4)]
for i in shapes:
  print(f"area of the shapes is {i.area()}")
```

Tupple prgram

```
tupple = [2,5,3,7,5]
print(tupple[2])
```

Class inheritance program

```
class animal:
  def __init__(self,name,species):
    self.name=name
    self.species=species

  def speak(self):
    print(f"the {self.name} make a sound")
class dog(animal):
  def __init__(self,name,breed,school_level):
    super().__init__(name,species="dog")
    self.breed=breed
    self.school_level=school_level

  def speak(self):
    print(f"{self.name} the {self.breed} says woof!")

  def school_level(self):
    print(f"{self.name} is at {self.school_level} level in training")

my_dog=dog("rex","german shepherd","intermediate")
my_dog.speak()
my_dog._school_level()
```

Creating bank account using oop

```
class bank:
  def __init__(self,balance):
    self.balance=balance
  def invest(self,amount):
    self.balance+=amount
    print(f"current balance{self.balance}")
  def withdraw(self,amount):
    if amount<=self.balace:
      self.balance-=amount
      print(f"current balance{self.balance}")
```

```
    else:
        print("no banalance")

obj=bank(1000)
obj.invest(500)
```

Multiple inheritance peogram

```
# multipple inheritance
class cars:
  def thar(self):
    print("its cost is 100000 and its milage is 200km/h")

class bike:
  def ktm(self):
    print("its cost is 200000 and its milage is 60km/h")

class Vehicles(cars,bike):
    pass

obj=Vehicles()
obj.thar()
```

Grading system program

```
marks=int(input("enter the marks"))
if marks>=90:
  print("grade A")
elif marks>=75 and marks<90:
    print("grade B")
elif marks>=50 and marks<75:
      print("grade C")
elif marks<50:
        print("grade F")
```