CODE :

```python
class Person:
    tabnine: test | explain | document | ask
    def _init_(self, fname, lname):
        self.firstname = fname
        self.lastname = lname

    tabnine: test | explain | document | ask
    def printname(self):
        print(self.firstname, self.lastname)

class Student(Person):
    tabnine: test | explain | document | ask
    def __init__(self, fname, lname, year):  # Ensure it accepts three arguments
        super().__init__(fname, lname)
        self.graduationyear = year


    tabnine: test | explain | document | ask
    def welcome(self):
        print("Welcome", self.firstname, self.lastname, "to the class of", self.graduationyear)

x = Student("Aditya", "Sharma", 2005)
 # Changed names
x.printname()
x.welcome()

class Polygon:
    tabnine: test | explain | document | ask
    def _init_(self, no_of_sides):
        self.n = no_of_sides
        self.sides = [0 for i in range(no_of_sides)]

    tabnine: test | explain | document | ask
    def inputSides(self):
        self.sides = [float(input("Enter side "+str(i+1)+" : ")) for i in range(self.n)]

    tabnine: test | explain | document | ask
    def dispSides(self):
        for i in range(self.n):
            print("Side",i+1,"is",self.sides[i])

class Triangle(Polygon):
    tabnine: test | explain | document | ask
    def _init_(self):
        super()._init_(3)

    tabnine: test | explain | document | ask
    def findArea(self):
        a, b, c = self.sides
        s = (a + b + c) / 2
        area = (s*(s-a)(s-b)(s-c)) ** 0.5
        print('The area of the triangle is %0.2f' %area)

t = Triangle()
t.inputSides()
t.dispSides()
t.findArea()
```

```python

class SuperClass:
    tabnine: test | explain | document | ask
    def super_method(self):
        print("Super Class method called")

class DerivedClass1(SuperClass):
    tabnine: test | explain | document | ask
    def derived1_method(self):
        print("Derived class 1 method called")

class DerivedClass2(DerivedClass1):
    tabnine: test | explain | document | ask
    def derived2_method(self):
        print("Derived class 2 method called")

d2 = DerivedClass2()
d2.super_method()
d2.derived1_method()
d2.derived2_method()

class shapes:
    tabnine: test | explain | document | ask
    def _init_(self, no_sides):
        self.n = no_sides
        self.sides = [0 for i in range(no_sides)]

    tabnine: test | explain | document | ask
    def takeSides(self):
        self.sides = [float(input("Enter side "+str(i+1)+" : ")) for i in range(self.n)]

    tabnine: test | explain | document | ask
    def disSides(self):
        for i in range(self.n):
            print("Side",i+1,"is",self.sides[i])

class rec(shapes):
    tabnine: test | explain | document | ask
    def _init_(self):
        shapes._init_(self,2)  # Changed number of sides

    tabnine: test | explain | document | ask
    def findArea(self):
        a, b = self.sides
        area = a * b  # Changed area calculation
        print('The area of the rectangle is', area)

t = rec()
t.takeSides()

mytuple = ("apple", "orange", "grapes")  # Changed fruits
myit = iter(mytuple)
print(next(myit))
print(next(myit))
print(next(myit))

mystr = "orange"  # Changed fruit
myit = iter(mystr)
print(next(myit))
print(next(myit))
print(next(myit))
print(next(myit))
print(next(myit))
```

```python
mytuple = ("apple", "orange", "grapes")  # Changed fruits
for x in mytuple:
    print(x)

mystr = "orange"  # Changed fruit
for x in mystr:
    print(x)

class MyNumbers:
    tabnine: test | explain | document | ask
    def __iter__(self):
        self.a = 2  # Changed starting number
        return self

    tabnine: test | explain | document | ask
    def __next__(self):
        if self.a <= 22:  # Changed upper limit
            x = self.a
            self.a += 2  # Changed increment value
            return x
        else:
            raise StopIteration

myclass = MyNumbers()
myiter = iter(myclass)
for x in myiter:
    print(x)

def myfunc():
    x = 500  # Changed value
    print(x)

myfunc()


def myfunc():
    x = 500  # Changed value
    def myinnerfunc():
        print(x)
    myinnerfunc()

myfunc()

x = 500  # Changed value
def myfunc():
    print(x)

myfunc()
print(x)

x = 500  # Changed value
def myfunc():
    x = 400  # Changed local value
    print(x)

myfunc()
print(x)

tabnine: test | explain | document | ask
def myfunc():
    global x
    x = 500  # Changed value
myfunc()
print(x)
```

OUTPUT :

```
Aditya Sharma
Welcome Aditya Sharma to the class of 2005
Enter side 1 : [user input]
Enter side 2 : [user input]
Enter side 3 : [user input]
Side 1 is [side 1 value]
Side 2 is [side 2 value]
Side 3 is [side 3 value]
The area of the triangle is [calculated area value]
Super Class method called
Derived class 1 method called
Derived class 2 method called
Enter side 1 : [user input]
Enter side 2 : [user input]
Side 1 is [side 1 value]
Side 2 is [side 2 value]
```