Student Management System:

```
class Student:
def __init__(self, name, rollno, m1, m2):
                self.name = name
                self.rollno = rollno
                self.m1 = m1
                self.m2 = m2

def accept(self, Name, Rollno, marks1, marks2):
                ob = Student(Name, Rollno, marks1, marks2)
                ls.append(ob)

def display(self, ob):
                print("Name : ", ob.name)
                print("RollNo : ", ob.rollno)
                print("Marks1 : ", ob.m1)
                print("Marks2 : ", ob.m2)
                print("\n")


def search(self, rn):
                for i in range(ls.__len__()):
                        if(ls[i].rollno == rn):
                                return i

def delete(self, rn):
                i = obj.search(rn)
                del ls[i]

def update(self, rn, No):
                i = obj.search(rn)
                roll = No
                ls[i].rollno = roll


ls = []
obj = Student(", 0, 0, 0)

print("\nOperations used, ")
print("\n1.Accept Student details\n2.Display Student Details\n3.Search Details of a Student\
n4.Delete Details of Student\n5.Update Student Details\n6.Exit")

ch = int(input("Enter choice:"))
 if(ch == 1):
obj.accept("ABC", 1, 100, 100)
obj.accept("BBB", 2, 90, 90)
obj.accept("CAD", 3, 80, 80)

elif(ch == 2):
print("\n")
print("\nList of Students\n")
for i in range(ls.__len__()):
```

```
            obj.display(ls[i])

 elif(ch == 3):
print("\n Student Found, ")
s = obj.search(2)
obj.display(ls[s])

elif(ch == 4):
obj.delete(2)
print(ls.__len__())
print("List after deletion")
for i in range(ls.__len__()):
            obj.display(ls[i])

elif(ch == 5):
obj.update(3, 2)
print(ls.__len__())
print("List after updation")
for i in range(ls.__len__()):
            obj.display(ls[i])

else:
print("Thank You !")
```

Inheritance Example

```
class Polygon:
    # Initializing the number of sides
    def __init__(self, no_of_sides):
        self.n = no_of_sides
        self.sides = [0 for i in range(no_of_sides)]

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

    # method to display the length of each side of the polygon
    def dispSides(self):
        for i in range(self.n):
            print("Side",i+1,"is",self.sides[i])

class Triangle(Polygon):
    # Initializing the number of sides of the triangle to 3 by
    # calling the __init__ method of the Polygon class
    def __init__(self):
        Polygon.__init__(self,3)

    def findArea(self):
        a, b, c = self.sides

        # calculate the semi-perimeter
        s = (a + b + c) / 2
```

```
        # Using Heron's formula to calculate the area of the triangle
        area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
        print('The area of the triangle is %0.2f' %area)

# Creating an instance of the Triangle class
t = Triangle()

# Prompting the user to enter the sides of the triangle
t.inputSides()

# Displaying the sides of the triangle
t.dispSides()

# Calculating and printing the area of the triangle
t.findArea()
```

Inheritance with super keyword:

```
# Python program to demonstrate
# super function

class Animals:
        # Initializing constructor
        def __init__(self):
                self.legs = 4
                self.domestic = True
                self.tail = True
                self.mammals = True

        def isMammal(self):
                if self.mammals:
                        print("It is a mammal.")

        def isDomestic(self):
                if self.domestic:
                        print("It is a domestic animal.")

class Dogs(Animals):
        def __init__(self):
                super().__init__()

        def isMammal(self):
                super().isMammal()

class Horses(Animals):
        def __init__(self):
                super().__init__()

        def hasTailandLegs(self):
                if self.tail and self.legs == 4:
                        print("Has legs and tail")
```

```
# Driver code
Tom = Dogs()
Tom.isMammal()
Bruno = Horses()
Bruno.hasTailandLegs()
```