1. **Write a program that accepts a sentence as input and remove duplicate words .Sort them alphanumerically and print it.**

```
#Hardcoded string
inputstring = "This is an example to sort words using a list"

# uncomment to take input from the user
#inputstring = input("Enter a string: ")

# breakdown the string into a list of words
words = inputstring.lower().split()

# Using sort() method to sort the list alphabetically
words.sort()

# display the sorted words

print("The sorted words are:")
for word in words:
    print(word)
```

2. With any given number n, Write a program to generate a dictionary that contains (k, k*k) .Print the dictionary generated and the series should include both 1 and k.

```
#Taking the dictionary range from the user
print("Enter the desired range for square dictionary:")
seriesrange = int(input())
#Initializing the dictionary
dictionary = dict()
#Iterating through the dictionary and appending square for the value
for number in range(1,seriesrange+1):
    dictionary[number]=number*number
#Printing the dictionary output
print(dictionary)
```

3. Write a python program to create any one of the following management systems. You can also pick one of your own. Airline Booking Reservation System (classes for Flight,Person,Employee,Passenger etc.

```
#Airline reservation system
#5 classes
#Flight, Person, Flights, Passenger, Pilot

#Person class for including Passenger, Flight booking agent, Piolot
class Person():      # class

    def __init__(self, name, flight): # method to map person and flight
        self.name = name #Usage of self
        self.flight = flight

    def __perdetails(self):         # private method to display person details
        print("Agent name: %s" % self.name)
        print("Flight name: %s" % self.flight)
```

```
class Flights():     # class to display flights in the airport

    def __init__(self):      # methods
        self.upcomingflights = 60
        self.departedflights = 30
        self.arrivingflights = 40

    def airlinedetails(self):
        print("Total number of flights: %s" % self.upcomingflights)
        print("Departed flights: %s" % self.departedflights)
        print("Arriving flights: %s" % self.arrivingflights)


class Passenger():   # class

    def __init__(self, name, passport):   #methods
        self.name = name
        self.passport = passport

    def passengerdetails(self):
        print("Passenger name: %s" % self.name)
        print("Passport : %d" % self.passport)

#Multiple Inheritance - Creation of Pilot to inherit both Person and Passenger
class Pilot(Person, Passenger):   #class

    def __init__(self, name, salary):   #methods
        self.name = name
        self.salary = salary

    def flightdetails(self):
        print("Pilot :", self.name, "Salary :", self.salary)

class Flight(Pilot):           # class which relates Pilot class to Flight
    def __init__(self, airportname, flightnumber):
        self.airportname = airportname
        self.flightnumber = flightnumber

    def pr(self):                   # method
        print(self.name)
        #Super class constructor calling
        super.__init__("Christopher","$7000")

# object creations and calling

person =Person("Christopher Bale","United Airlines")
#Private method call to display person details
person._Person__perdetails()

#Methods to display passenger details
passenger=Passenger("Tessa",8178603)
passenger.passengerdetails()

#Pilot details
pilot = Pilot("Ashley,",3000)
pilot.flightdetails()

#Airport details
airport = Flight("Denver International Airport", 130)
print("Airport: %s" % airport.airportname)
print("Total number of flights : %d" % airport.flightnumber)
```

4. Using Numpy create random vector of size 15 and replace the maximum value by 100

```
#Importing numpy package
import numpy as np
#Generating random vector
vector = np.random.random(15)
#appending maximum value to the vector
vector[vector.argmax()] = 10
print(vector)
```