

## PYTHON LAB ASSIGNMENT – 2

Sai Mohith Reddy Chagamreddy

Class ID - 9

**Objective:** The main objective of this lab assignment is to learn Object Oriented Programming in Python and get familiar with the numpy library.

**Features:**

The programs in the offers the following features:

1. Display Books that are in price range given by the User
2. A Contacts application, where user can see and edit the contact info
3. A School Management System
4. Display most frequently occurred numbers from a random list using Numpy

**Configuration:**

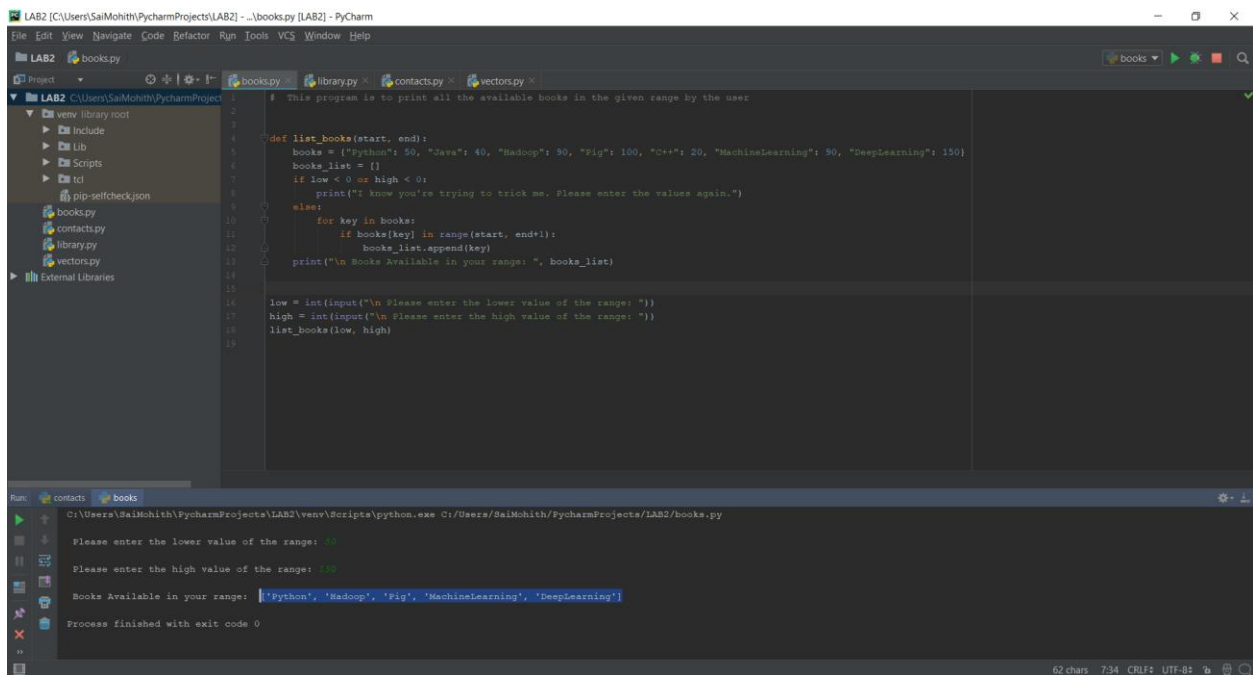
Python 3.6 interpreter

Numpy Library

JetBrains Pycharm Community Edition

Screenshots:

## 1. Books in the given price range:



The screenshot shows the PyCharm IDE with the 'books.py' file open. The code defines a dictionary of books with their prices and a function to filter them by a given price range. The execution output shows the user entering a range from 50 to 150, resulting in a list of books: Python, Madoop, Fig, MachineLearning, and DeepLearning.

```
LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - \books.py [LAB2] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help

LAB2 C:\Users\SaiMohith\PycharmProjects\LAB2
venv library root
  Include
  Lib
  Scripts
  tcl
  pip-selfcheck.json
books.py
contacts.py
library.py
vectors.py
External Libraries

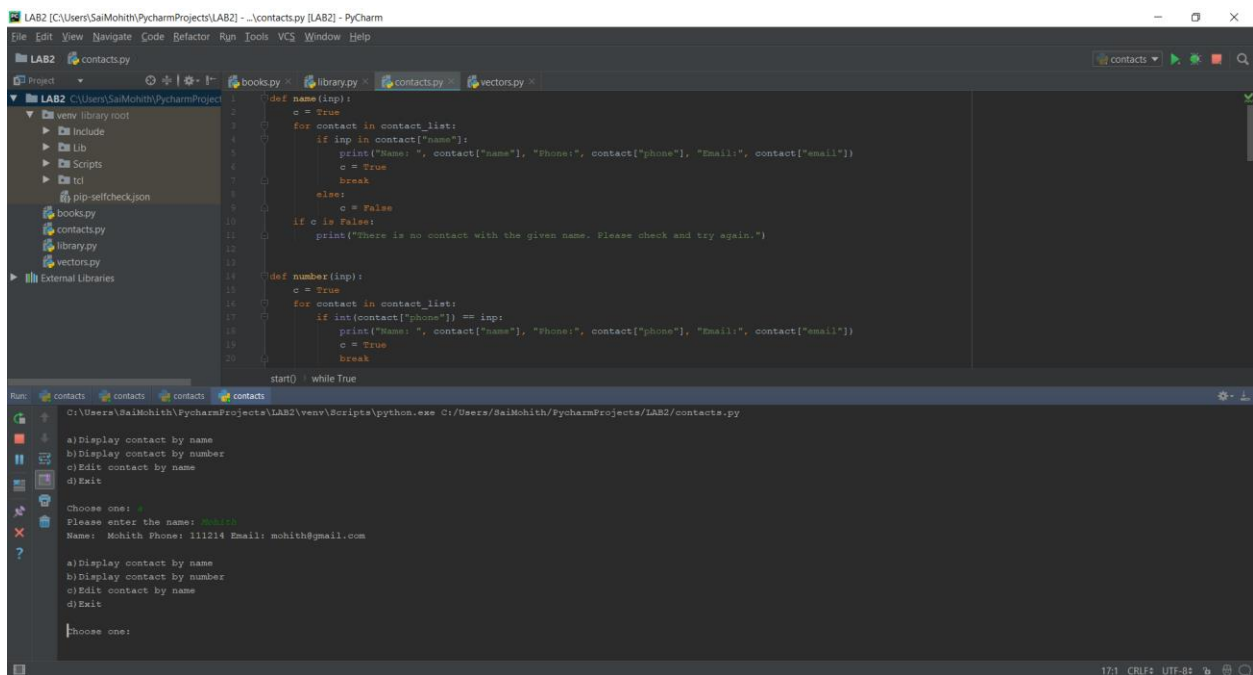
1 # This program is to print all the available books in the given range by the user
2
3
4 def list_books(start, end):
5     books = {"Python": 50, "Java": 40, "Madoop": 30, "Fig": 100, "C++": 20, "MachineLearning": 90, "DeepLearning": 150}
6     books_list = []
7     if low < 0 or high < 0:
8         print("I know you're trying to trick me. Please enter the values again.")
9     else:
10        for key in books:
11            if books[key] in range(start, end+1):
12                books_list.append(key)
13        print("\n Books Available in your range: ", books_list)
14
15 low = int(input("\n Please enter the lower value of the range: "))
16 high = int(input("\n Please enter the high value of the range: "))
17 list_books(low, high)
18
19

Run: contacts books
C:\Users\SaiMohith\PycharmProjects\LAB2\venv\Scripts\python.exe C:\Users\SaiMohith\PycharmProjects\LAB2\books.py

Please enter the lower value of the range: 50
Please enter the high value of the range: 150
Books Available in your range: ['Python', 'Madoop', 'Fig', 'MachineLearning', 'DeepLearning']
Process finished with exit code 0

62 chars 7:34 CRLF UTF-8
```

## 2. Contacts Application:



The screenshot shows the PyCharm IDE with the 'contacts.py' file open. The code defines a dictionary of contacts and functions to display, edit, or delete a contact by name or number. The execution output shows the user choosing to display a contact by name, entering 'Mohith', and seeing the contact details: Mohith Phone: 111214 Email: mohith@gmail.com.

```
LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - \contacts.py [LAB2] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help

LAB2 C:\Users\SaiMohith\PycharmProjects\LAB2
venv library root
  Include
  Lib
  Scripts
  tcl
  pip-selfcheck.json
books.py
contacts.py
library.py
vectors.py
External Libraries

1 def name(inp):
2     c = True
3     for contact in contact_list:
4         if inp in contact["name"]:
5             print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
6             c = True
7             break
8     else:
9         c = False
10    if c is False:
11        print("There is no contact with the given name. Please check and try again.")
12
13 def number(inp):
14     c = True
15     for contact in contact_list:
16         if int(contact["phone"]) == inp:
17             print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
18             c = True
19             break
20
21 start() while True

Run: contacts contacts contacts
C:\Users\SaiMohith\PycharmProjects\LAB2\venv\Scripts\python.exe C:\Users\SaiMohith\PycharmProjects\LAB2\contacts.py

a)Display contact by name
b)Display contact by number
c)Edit contact by name
d)Exit
Choose one: a
Please enter the name: Mohith
Name: Mohith Phone: 111214 Email: mohith@gmail.com

a)Display contact by name
b)Display contact by number
c)Edit contact by name
d)Exit
Choose one:

17:1 CRLF UTF-8
```

LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...contacts.py [LAB2] - PyCharm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

LAB2 contacts.py

Project

view library root

Include

Lib

Scripts

tdl

pip-selfcheck.json

books.py

contacts.py

library.py

vectors.py

External Libraries

```
11 if c is False:
12     print("There is no contact with the given name. Please check and try again.")
13
14 def number(inp):
15     c = True
16     for contact in contact_list:
17         if int(contact["phone"]) == inp:
18             print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
19             c = True
20             break
21         else:
22             c = False
23     if c is False:
24         print("There is no contact with given number. Please check and try again.")
25
26
27 def edit(inp):
28     c = True
29     for contact in contact_list:
30
31 start() while True
```

Run: contacts contacts contacts contacts

Choose one: 1

Please enter the name: Mohith

Name: Mohith Phone: 111214 Email: mohith@gmail.com

a) Display contact by name  
b) Display contact by number  
c) Edit contact by name  
d) Exit

Choose one: 2

Please enter the number: 111214

Name: Mohith Phone: 111214 Email: mohith@gmail.com

a) Display contact by name  
b) Display contact by number  
c) Edit contact by name  
d) Exit

Choose one:

96 chars, 2 line breaks 1952 CRLF: UTF-8

LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...contacts.py [LAB2] - PyCharm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

LAB2 contacts.py

Project

view library root

Include

Lib

Scripts

tdl

pip-selfcheck.json

books.py

contacts.py

library.py

vectors.py

External Libraries

```
28 c = True
29 for contact in contact_list:
30     if inp in contact["name"]:
31         print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
32         print("1) Edit Name\n2) Edit Number\n3) Edit Email\n4) Exit\n")
33         while True:
34             new = int(input("Choose what to edit: "))
35             if new == 1:
36                 new_name = str(input("New Name: "))
37                 contact["name"] = new_name
38                 print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
39             elif new == 2:
40                 new_number = int(input("New Number: "))
41                 contact["phone"] = new_number
42                 print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
43             elif new == 3:
44                 new_email = str(input("New Email: "))
45                 contact["email"] = new_email
46                 print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
47             elif new == 4:
48                 break
49             else:
50                 print("Invalid Response")
51         c = True
52         break
53     else:
54         c = False
55     if c is False:
56         print("There is no contact with the given name. Please check and try again.")
57
58 start() while True
```

Run: contacts contacts contacts contacts

Choose one: 1

Please enter the name to edit the contact: Mohith

Name: Mohith Phone: 111214 Email: mohith@gmail.com

1) Edit Name  
2) Edit Number  
3) Edit Email  
4) Exit

Choose what to edit:

34:1 CRLF: UTF-8

```
LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...contacts.py [LAB2] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help

Project: LAB2
  view library root
  Include
  Lib
  Scripts
  tcl
  pip-selfcheck.json
  books.py
  contacts.py
  library.py
  vectors.py

External Libraries:

Run: contacts contacts contacts contacts
Choose one: 1
Please enter the name to edit the contact: Mohith
Name: Mohith Phone: 111214 Email: mohith@gmail.com
1) Edit Name
2) Edit Number
3) Edit Email
4) Exit
Choose what to edit: 1
New Name: Sai
Name: Sai Phone: 111214 Email: mohith@gmail.com
Choose what to edit: 2
New Number: 16233203
Name: Sai Phone: 16233203 Email: mohith@gmail.com
Choose what to edit: 3
New Email: saimohith@gmail.com
Name: Sai Phone: 16233203 Email: saimohith@gmail.com
Choose what to edit: 4
a) Display contact by name
b) Display contact by number
c) Edit contact by name
d) Exit
Choose one: |
```

### 3. School Management Systems:

```
LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...library.py [LAB2] - PyCharm
File Edit View Navigate Code Refactor Run Tools VCS Window Help

Project: LAB2
  view library root
  Include
  Lib
  Scripts
  tcl
  pip-selfcheck.json
  books.py
  contacts.py
  library.py
  vectors.py

External Libraries:

Run: contacts contacts contacts library
Name: Mohith
ID: 16233203
Dept: CS
Mohith is a school member of UMKC
Mohith is Student
Grade is A. Outstanding!
Name: Minh
ID: 18646537
Dept: Communication
Minh is a school member of UMKC
Minh is Student
Grade is F. Study Hard!
Name: YuyTung Lee (Ph.D)
Dept: CS
YuyTung Lee is a school member of UMKC
YuyTung Lee is Instructor for Python & BigData
```

LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - \library.py [LAB2] - PyCharm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

Project LAB2

- view library root
- Include
- Lib
- Scripts
- test
- pip-selfcheck.json
- books.py
- contacts.py
- library.py
- vectors.py
- External Libraries

```
24 def add_teacher(self, i):
25     SchoolMember.num_of_teachers += i
26
27
28 class Grade: # Another parent class for getting the Grade (2nd Class)
29
30 def __init__(self, g): # init constructor
31     self.g = g
32     if self.g in range(90, 101):
33         print("Grade is A. Outstanding!")
34     elif self.g in range(80, 90):
35         print("Grade is A-. Excellent!")
36     elif self.g in range(70, 80):
37         print("Grade is B. Good!")
38     elif self.g in range(60, 70):
39         print("Grade is B-. Average!")
40     elif self.g in range(50, 60):
41         print("Grade is C. Below Average!")
42     else:
43         print("Grade is F. Study Hard!")
```

Run: contacts contacts contacts library

Name: Yjie Han (Ph.D)  
Dept: CS  
Yjie Han is a school member of UMKC  
Yjie Han is instructor for Parallel Algorithms

Name: Rashmi  
ID: 9274527  
Dept: CSE  
Rashmi is a school member of UMKC  
Rashmi is Student  
Grade is A. Outstanding!  
Rashmi works under Lee as RA

Name: Dig Vijay  
ID: 91746292  
Dept: CS  
Dig Vijay is a school member of UMKC  
Dig Vijay is Student

10758 CRUF: UTF-8

LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - \library.py [LAB2] - PyCharm

File Edit View Navigate Code Refactor Run Tools VCS Window Help

Project LAB2

- view library root
- Include
- Lib
- Scripts
- test
- pip-selfcheck.json
- books.py
- contacts.py
- library.py
- vectors.py
- External Libraries

```
34 class Teacher(SchoolMember): # Single Inheritance (3rd Class)
35
36 def __init__(self, name, dep, c, suffix): # init constructor
37     self.name = name
38     self.dep = dep
39     self.c = c
40     self.suffix = suffix
41     print("\nName:", name, "{}{}".format(suffix), "\nDept:", dep)
42     super().__init__(name) # Super call for SchoolMember
43     super().display_teacher_info(name, c)
44     super().add_member(i)
45     super().add_teacher(i)
46
47
48 class Student(SchoolMember, Grade): # Multiple Inheritance (4th Class). Inherits SchoolMember and Grade Classes
49
50 def __init__(self, name, dep, id, grade): # init constructor
51     self.name = name
52     self.dep = dep
53     self.id = id
54     self.grade = grade
55     print("\nName:", name, "\nID:", id, "\nDept:", dep)
56     super().__init__(name) # Super call for SchoolMember
57     super().display_at_info(name)
58     Grade(grade) # calls the Grade Class
59     super().add_member(i)
60     super().add_student(i)
```

Run: contacts contacts contacts library

Name: Dig Vijay  
ID: 91746292  
Dept: CS  
Dig Vijay is a school member of UMKC  
Dig Vijay is Student  
Grade is A. Outstanding!  
Dig Vijay works under Praveen Rao as RA

10758 CRUF: UTF-8

The image shows a PyCharm IDE window titled "LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...\library.py [LAB2] - PyCharm". The editor displays the following Python code:

```
83 class RA(Student): # Sch Class. It inherits Student Class
84
85     def __init__(self, name, dep, id, grade, prof): # init constructor
86         self.name = name
87         self.id = id
88         self.dep = dep
89         self.grade = grade
90         self.prof = prof
91
92     def __init__(name, dep, id, grade) # Super call for Student
93         super().__init__(name, dep, id, grade, "Prof") # also inherits the SchoolMember class
94
95 # Student Instance
96 st1 = Student("Mohith", "CS", "1623203", 95)
97 st2 = Student("Mish", "Communication", "1646537", 49)
98
99 # Teacher Instance
100 t1 = Teacher("Yugyung Lee", "CS", "Python & BigData", "Ph.D")
101 t2 = Teacher("Jillie Han", "CS", "Parallel Algorithms", "Ph.D")
102
103 # RA Instance
104 r1 = RA("Rashmi", "CSE", "9274827", 91, "Lee")
105 r2 = RA("Dip Vijay", "CS", "91748292", 99, "Rashmi")
106
107 # School Member Instance
108 print("\n")
109 sm = SchoolMember("Jack")
110
```

The Run console at the bottom shows the following output:

```
Jack is a school member of UMEC
Grade is A-. Excellent!
Number of School Members: 6
Number of Students: 4
Number of Teachers: 2
```

#### 4. Vectors output:

The image shows a PyCharm IDE window titled "LAB2 [C:\Users\SaiMohith\PycharmProjects\LAB2] - ...\vectors.py [LAB2] - PyCharm". The editor displays the following Python code:

```
1 import numpy as np
2
3 array = np.random.randint(0, 20, size=15)
4 print(array)
5 counts = np.bincount(array)
6 print("Most Repeated Value:", np.argmax(counts))
7
```

The Run console at the bottom shows the following output:

```
C:\Users\SaiMohith\PycharmProjects\LAB2\venv\Scripts\python.exe C:\Users\SaiMohith\PycharmProjects\LAB2\vectors.py
[ 7  4  1  2 17 12  0  4  1  5  1 15 17 15 13]
Most Repeated Value: 1
Process finished with exit code 0
```

Code Implementation:

### 1. Books program:

This program is to display all the books in the library available according to the price range given by the user.

We define a function `list_books` here to check the books available in the price range. Also we create a dictionary of books available in the library in this function. Keys are the titles and the values are the prices.

```
def list_books(start, end):  
  
    books = {"Python": 50, "Java": 40, "Hadoop": 90, "Pig": 100, "C++": 20, "MachineLearning":  
90, "DeepLearning": 150}  
  
    books_list = []  
  
    if low < 0 or high < 0:  
        print("I know you're trying to trick me. Please enter the values again.")  
    else:  
        for key in books:  
            if books[key] in range(start, end+1):  
                books_list.append(key)  
  
    print("\n Books Available in your range: ", books_list)
```

Price ranges will be entered by the user and the function will be called.

```
low = int(input("\n Please enter the lower value of the range: "))  
high = int(input("\n Please enter the high value of the range: "))  
list_books(low, high)
```

## 2. Contacts List Application:

This program is to display contacts in the contact list based on the user specification. We can display the contact by name or by number. User can also edit the contact in this program.

To get the contact info by name, we define a function **\*\*name\*\*** which takes the user given input and displays the contact info if available. If the name given by user is not available, it displays **\*\*There is no contact with the given name. Please check and try again.\*\***

```
def name(inp):  
    c = True  
    for contact in contact_list:  
        if inp in contact["name"]:  
            print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])  
            c = True  
            break  
    else:  
        c = False  
    if c is False:  
        print("There is no contact with the given name. Please check and try again.")
```

In the same way, we define a function **\*\*number\*\*** to display the contact info based on the given number.

```
def number(inp):  
    c = True  
    for contact in contact_list:  
        if int(contact["phone"]) == inp:  
            print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])  
            c = True  
            break  
    else:  
        c = False
```



if c is False:

```
print("There is no contact with given number. Please check and try again.")
```

We, define a function **\*\*edit\*\*** to edit the contact info. We ask user to choose to edit by name, number or email. After editing the info the edited contact is displayed.

```
def edit(inp):
```

```
c = True
```

```
for contact in contact_list:
```

```
    if inp in contact["name"]:
```

```
        print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:", contact["email"])
```

```
        print("1) Edit Name\n""2)Edit Number\n""3)Edit Email\n""4)Exit\n")
```

```
    while True:
```

```
        new = int(input("Choose what to edit: "))
```

```
        if new == 1:
```

```
            new_name = str(input("New Name: "))
```

```
            contact["name"] = new_name
```

```
            print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:",  
contact["email"])
```

```
        elif new == 2:
```

```
            new_number = int(input("New Number: "))
```

```
            contact["phone"] = new_number
```

```
            print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:",  
contact["email"])
```

```
        elif new == 3:
```

```
            new_email = str(input("New Email: "))
```

```
            contact["email"] = new_email
```

```
            print("Name: ", contact["name"], "Phone:", contact["phone"], "Email:",  
contact["email"])
```

```
        elif new == 4:
```

```

        break

    else:

        print("Invalid Response")

    c = True

    break

else:

    c = False

if c is False:

    print("There is no contact with the given name. Please check and try again.")

```

Then we define a function **\*\*start\*\*** and call it to start the application. In the start function, we ask the user to choose the action by giving different options a, b, c, d as shown below. If User choose a, function name will be called. number will be called when b, edit will be called when c, application will stop when user choose d. If user enters any other, then it prompts user to select a valid option. Then we create a contact list and start the application.

```

def start():

while True:

    print("\na)Display contact by name \n"

        "b)Display contact by number \n"

        "c)Edit contact by name \n"

        "d)Exit \n")

    ch = str(input("Choose one: "))

    if ch == 'a':

        n = str(input("Please enter the name: "))

        name(n)

    elif ch == 'b':

        n = int(input("please enter the number:"))

        number(n)

    elif ch == 'c':

```

```

n = str(input("Please enter the name to edit the contact:"))

edit(n)

elif ch == 'd':

    exit()

else:

    print("please choose from the given options")

contact_list = [{"name": "Uma", "phone": "910529", "email": "dad@gmail.com"},
                {"name": "Srinivasa", "phone": "910210", "email": "mom@gmail.com"},
                {"name": "Sumanth", "phone": "910928", "email": "bro@gmail.com"},
                {"name": "Mohith", "phone": "111214", "email": "mohith@gmail.com"}]

start()

```

### 3. School Management Systems:

This program is to create a school management system. We have five classes:

1. **\*\*SchoolMember\*\*** which is responsible for displaying the info and adding the members etc.
2. **\*\*Grade\*\*** which provides the grade to the student depending on their score.
3. **\*\*Student\*\*** which takes the info of the students. It inherits the class SchoolMember and Grade (Multiple Inheritance).
4. **\*\*Teacher\*\*** which takes the info of the students. It also inherits the class SchoolMember. (Single Inheritance)
5. **\*\*RA\*\*** which takes the info of the RA. It inherits the Student Class and SchoolMember.

We used `__init__` constructor to build all classes. Different methods are used in each class. We invoke classes by creating instances at the end of the code. Super function is used to call upon single inheritance and multiple. We use self argument which automatically points to the current object

Each class have different methods to display info as shown below:

```

class SchoolMember: # Parent Class School Member (1st Class)
    num_of_school_members = 0
    num_of_students = 0
    num_of_teachers = 0
    __name_of_school = "UMKC" # Private Data Member

    def __init__(self, name): # init constructor
        self.name = name # self

        print(name, "is a school member of", SchoolMember.__name_of_school) # displays
school members

    def display_teacher_info(self, name, course): # displays teacher informarion
        self.name = name

        self.course = course

        print(name, "is instructor for", course)

    def display_ra_info(self, name, professor): # displays RA information
        self.name = name

        self.professor = professor

        print(name, "works under", professor, "as RA")

    def display_st_info(self, name): # displays Student information
        self.name = name

        print(name, "is Student")

    def add_member(self, i):

        SchoolMember.num_of_school_members += i

```

```
def add_student(self, i):
```

```
    SchoolMember.num_of_students += i
```

```
def add_teacher(self, i):
```

```
    SchoolMember.num_of_teachers += i
```

```
class Grade: # Another parent class for getting the Grade (2nd Class)
```

```
    def __init__(self, g): # init constructor
```

```
        self.g = g
```

```
        if self.g in range(90, 101):
```

```
            print("Grade is A. Outstanding!")
```

```
        elif self.g in range(80, 90):
```

```
            print("Grade is A-. Excellent!")
```

```
        elif self.g in range(70, 80):
```

```
            print("Grade is B. Good!")
```

```
        elif self.g in range(60, 70):
```

```
            print("Grade is B-. Average!")
```

```
        elif self.g in range(50, 60):
```

```
            print("Grade is C. Below Average!")
```

```
        else:
```

```
            print("Grade is F. Study Hard!")
```

```
class Teacher(SchoolMember): # Single Inheritance (3rd Class)
```

```

def __init__(self, name, dep, c, suffix): # init constructor
    self.name = name
    self.dep = dep
    self.c = c
    self.suffix = suffix
    print("\nName:", name, ('{}`).format(suffix)), "\nDept:", dep)
    super().__init__(name) # Super call for SchoolMember
    super().display_teacher_info(name, c)
    super().add_member(1)
    super().add_teacher(1)

```

class Student(SchoolMember, Grade): # Multiple Inheritance (4th Class). Inherits SchoolMember and Grade Classes

```

def __init__(self, name, dep, id, grade): # init constructor
    self.name = name
    self.dep = dep
    self.id = id
    self.grade = grade
    print("\nName:", name, "\nID:", id, "\nDept:", dep)
    super().__init__(name) # Super call for SchoolMember
    super().display_st_info(name)
    Grade(grade) # calls the Grade Class
    super().add_member(1)
    super().add_student(1)

```

```
class RA(Student): # 5th Class. It inherits Student Class
```

```
    def __init__(self, name, dep, id, grade, prof): # init constructor
        self.name = name
        self.id = id
        self.dep = dep
        self.grade = grade
        self.prof = prof
        super().__init__(name, dep, id, grade) # Super call for Student
        super().display_ra_info(name, prof) # also inherits the Schoolmember class
```

We create instances for all the above classes with the info of students, teachers, RA.

```
# Studeent Instance
```

```
st1 = Student("Mohith", "CS", "16233203", 95)
st2 = Student("Minh", "Communication", "18646537", 49)
```

```
# Teacher Instance
```

```
t1 = Teacher("YugYung Lee", "CS", "Python & BigData", "Ph.D")
t2 = Teacher("Yjie Han", "CS", "Parallel Algorithms", "Ph.D")
```

```
# RA Instance
```

```
r1 = RA("Rashmi", "CSE", "9274827", 91, "Lee")
r2 = RA("Dig Vijay", "CS", "91746292", 98, "Praveen Rao")
```

```
# School Member Instance
```

```
print("\n")
sm = SchoolMember("Jack")
```

```
# Grade Instance
```

```
print("\n")
```

```
gd = Grade(89)
```

```
print("\n")
```

```
print("Number of School Members:", SchoolMember.num_of_school_members)
```

```
print("Number of Students:", SchoolMember.num_of_students)
```

```
print("Number of Teachers:", SchoolMember.num_of_teachers)
```

#### 4. Vectors:

This program to get the most frequently repeated element in the array created by random numbers between 0-20 using Numpy.

We first import the numpy library as shown below:

```
import numpy as np
```

We create a array of size 15 with the random elements between 0-20. We use bincount function in the numpy to get the counts of each element. Then we use argmax from the numpy to get the element with has max value in the count.

```
array = np.random.randint(0, 20, size=15)
```

```
print(array)
```

```
counts = np.bincount(array)
```

```
print("Most Repeated Value:", np.argmax(counts))
```



Code Deployment:

1. Books Program:

Program output for the input ranges 50 and 150 is shown below:

Please enter the lower value of the range: 50

Please enter the high value of the range: 150

Books Available in your range: ['Python', 'Hadoop', 'Pig', 'MachineLearning', 'DeepLearning']

2. Contact List Output:

The output for the program for the user actions:

a)Display contact by name

b)Display contact by number

c)Edit contact by name

d)Exit

Choose one: a

Please enter the name: Mohith

Name: Mohith Phone: 111214 Email: mohith@gmail.com

a)Display contact by name

b)Display contact by number

c)Edit contact by name

d)Exit

Choose one: b

please enter the number:111214

Name: Mohith Phone: 111214 Email: mohith@gmail.com

a)Display contact by name

b)Display contact by number

c)Edit contact by name

d)Exit

Choose one: c

Please enter the name to edit the contact:Mohith

Name: Mohith Phone: 111214 Email: mohith@gmail.com

1) Edit Name

2)Edit Number

3)Edit Email

4)Exit

Choose what to edit: 1

New Name: Sai

Name: Sai Phone: 111214 Email: mohith@gmail.com

Choose what to edit: 2

New Number: 16233203

Name: Sai Phone: 16233203 Email: mohith@gmail.com

Choose what to edit: 3

New Email: saimohith@gmail.com

Name: Sai Phone: 16233203 Email: saimohith@gmail.com

Choose what to edit: 4

a)Display contact by name

b)Display contact by number

c)Edit contact by name

d)Exit

Choose one: c

Please enter the name to edit the contact: Mohith

There is no contact with the given name. Please check and try again.

- a) Display contact by name
- b) Display contact by number
- c) Edit contact by name
- d) Exit

Choose one: e

please choose from the given options

- a) Display contact by name
- b) Display contact by number
- c) Edit contact by name
- d) Exit

Choose one: d

Process finished with exit code 0

### 3. School Management System:

Output for the given instances:

Name: Mohith

ID: 16233203

Dept: CS

Mohith is a school member of UMKC

Mohith is Student

Grade is A. Outstanding!

Name: Minh

ID: 18646537

Dept: Communication

Minh is a school member of UMKC

Minh is Student

Grade is F. Study Hard!

Name: YugYung Lee (Ph.D)

Dept: CS

YugYung Lee is a school member of UMKC

YugYung Lee is instructor for Python & BigData

Name: Yjie Han (Ph.D)

Dept: CS

Yjie Han is a school member of UMKC

Yjie Han is instructor for Parallel Algorithms

Name: Rashmi

ID: 9274827

Dept: CSE

Rashmi is a school member of UMKC

Rashmi is Student

Grade is A. Outstanding!

Rashmi works under Lee as RA

Name: Dig Vijay

ID: 91746292

Dept: CS

Dig Vijay is a school member of UMKC

Dig Vijay is Student

Grade is A. Outstanding!

Dig Vijay works under Praveen Rao as RA

Jack is a school member of UMKC

Grade is A-. Excellent!

Number of School Members: 6

Number of Students: 4

Number of Teachers: 2

Process finished with exit code 0

#### 4. Vectors:

The output for randomly created array:

[ 0 5 2 13 13 0 8 19 12 14 2 13 18 0 13]

Most Repeated Value: 13

Process finished with exit code 0

#### Limitations:

Python 3.6 is used for writing this code. Some functions may not work properly when run on python 2(2.x) version.

#### References:

<https://docs.python.org/2/tutorial/classes.html>

<https://www.geeksforgeeks.org/object-oriented-programming-in-python-set-2-data-hiding-and-object-printing/>

<https://www.pythonlearn.com/html-008/cfbook022.html>

[www.stackoverflow.com](http://www.stackoverflow.com)