St. Francis Institute of Technology, Mumbai-400 103

A.Y. 2023-24

Class: SE-ITA/ITB, Semester: IV

Subject: **Python Lab**.

Experiment – 10: Python program to implement modules and packages.

1. **Aim:** To implement a python program to demonstrate the following:
   a. Implement queue using deque (deck) and show insertion of element from the rear side, deletion of element from the front side, rotate and extend queue..

   b. Create a user defined module to implement a data structure queue. The module should perform the following functions:
   1. Enqueue element from the rear side
   2. Dequeue element from the front side
   3. Rotate the queue
   4. Extend queue

2. Prerequisite:. Knowledge of Modules and Packages in Python.

3. Objective: Using In-built and modules and creating user defined modules.

4. Requirements: Personal Computer (PC), Windows /Linux Operating System, IDLE 3.6 for Python3.

5. Pre-Experiment Exercise:

Theory:

Modular programming refers to the process of breaking a large, unwieldy programming task into separate, smaller, more manageable subtasks or modules. Individual modules can then be cobbled together like building blocks to create a larger application.

The module is a simple Python file that contains collections of functions and global variables and with having a . py extension file. It is an executable file and to organize all the modules we have the concept called Package in Python.

There are actually three different ways to define a module in Python:

1. A module can be written in Python itself.
2. A module can be written in C and loaded dynamically at run-time, like the re (regular expression) module.
3. A built-in module is intrinsically contained in the interpreter, like the itertools module.

Module contents are made available to the caller with the import statement. The import statement takes many different forms, shown below.
import <module_name>
Alternate form
from <module name> import <name>
To import everything from a module=>
from <module_name> import *

Eg:Example: Consider a Python module math.py that contains a function to calculate the square of a number.

```
#math.py module

def square(i):

    return x**2
```

This module can be imported and used in the different files as follows:

```
#main.py file

import math

print(math.square(5)) #output 25
```

Python Packages are collections of modules that provide a set of related functionalities, and these modules are organized in a directory hierarchy. In simple terms, packages in Python are a way of organizing related modules in a single namespace.

Packages in Python are installed using a package manager like pip (a tool for installing and managing Python packages).

Each Python package must contain a file named _init_.py.

- _init_ file may be empty.
- This file contains the initialization code for the corresponding package.

Let there be any package (named my_package) that contains two sub-modules (mod_1, and mod_2)

my_package/

init.py

mod_1.py

mod_2.py

Note: init.py file is required to make Python treat the dictionary as a package.

Now import package in your program by writing import my_package.mod_1.py

Difference Between Module and Package in Python

Parameter	Module	Package
Definition	It can be a simple Python file (.py extensions) that contains collections of functions and global variables.	A Package is a collection of different modules with an _init_.py file.
Purpose	Code organization	Code distribution and reuse
Organization	Code within a single file	Related modules in a directory hierarchy
Sub-modules	None	Multiple sub-modules and sub-packages
Required Files	Only Python File(.py format)	'_init_.py' file and python files
How to import	import **module_name**	import **package_name.module_name**
Example	math, random, os, datetime, csv	Numpy, Pandas, Matplotlib, django

6. Laboratory Exercise

A. Procedure

.Open Idle for python

i. Open editor in Idle from menu file-new

ii. Type python code with proper syntax

iii. Save file with .py extension

iv. Execute the code inside the saved file using shortcut key F5 or using menu: Run-Run module

B. Program code with comments:

Write and execute your program code to achieve the given aim and attach it **with your own comments with neat indentation**.

7. Post-Experiments Exercise

A. Extended Theory:

B. Questions/Programs:

1. Write a Python program to create any user defined module and call its functionalities in another program.

C. **Conclusion:**

1. Write what was performed in the experiment/program.
2. What is the significance of experiment/program?

D. **References**

1. James Payne, ”Beginning Python: Using Python 2.6 and Python 3.1”,WroxPublication.
2. Dr.Nageswara Rao,”Core Python Programming”,Wiley Publication.
3. https://www.python.org/
4. www.pythonforbeginners.com
5. https://realpython.com/python-deque/
6. https://www.shiksha.com/online-courses/articles/difference-between-module-and-package-in-python/#:~:text=In%20simple%20terms%2C%20a%20module,organized%20in%20a%20directory%20hierarchy.
7. https://www.educative.io/answers/how-to-implement-a-queue-in-python
8. https://www.geeksforgeeks.org/deque-in-python/
9.

In- Lab Exercise

Q1.Implement a queue using deque (deck) and show insertion of elements from the rear side, deletion of elements from the front side, rotate and extend the queue.

Code:

```
import collections
List = list()
de = collections.deque(List)
def Enqueue(de):
    a = int(input("Enter the element to add into queue:"))
    de.append(a)

def Dequeue(de):
    de.popleft()

def rotate(de):
    de.rotate(-1)

def Extend(de):
    n = int(input("Enter total number to extend"))
    List1 = []
    print("Enter the elements:")
    for a in range(n):
        a = int(input())
        List1.append(a)
    de.extend(List1)

def Print(de):
    print("\n",de)

while(True):
```

```
                                        i=int(input("\nMenu          of
Deque\n1.Enqueue\n2.Dequeue\n3.Rotate\n4.Extend\n5.Print\n6.Exit\nEnter
your choice:"))
    if i == 1:
        Enqueue(de)
    elif i == 2:
        Dequeue(de)
    elif i == 3:
        rotate(de)
    elif i == 4:
        Extend(de)
    elif i == 5:
        Print(de)
    elif i == 6:
        break
```

Output:

1. Enqueue

```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>python deque.py

Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 1
Enter the element to add into queue: 63

Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 deque([63])
```

2. Dequeue

```
Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 2

Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 deque([])
```

3. Extend

```
Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 4
Enter total number to extend: 3
Enter the elements:
63
64
65

Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 deque([63, 64, 65])
```

4. Rotate

```
Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 3

Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 deque([64, 65, 63])
```

5. Print

```
Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 deque([64, 65, 63])
```

6. Exit

```
Menu of Deque
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 6

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>
```

Q2.Create a user defined module to implement a data structure queue. The module should perform the following functions:

1. Enqueue element from the rear side
2. Dequeue element from the front side
3. Rotate the queue
4. Extend queue

Code:

```
class Queue:
    def __init__(self):
        self.items = []

    def enqueue(self, item):
        self.items.append(item)

    def dequeue(self):
        if not self.is_empty():
            return self.items.pop(0)
        return None

    def rotate(self):
        if not self.is_empty():
            temp = self.dequeue()
            self.enqueue(temp)

    def extend(self, items):
        self.items.extend(items)

    def is_empty(self):
        return len(self.items) == 0

queue = Queue()

while(True):
```

```
                i    =    int(input("\nMenu    of    User
Defined\n1.Enqueue\n2.Dequeue\n3.Rotate\n4.Extend\n5.Prin
t\n6.Exit\nEnter your choice:"))
    if i == 1:
            item = input("Enter the element to add into
queue:")
        queue.enqueue(item)
    elif i == 2:
        queue.dequeue()
    elif i == 3:
        queue.rotate()
    elif i == 4:
        n = int(input("Enter total number to extend"))
        items = []
        print("Enter the elements:")
        for a in range(n):
            a = input()
            items.append(a)
        queue.extend(items)
    elif i == 5:
        print("\n", queue.items)
    elif i == 6:
        break
    else:
        print("Invalid choice")
```

Output:

1. Enqueue

```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>python queue.py

Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 1
Enter the element to add into queue: 63

Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 ['63']
```

2.Dequeue

```
Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 2

Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 []
```

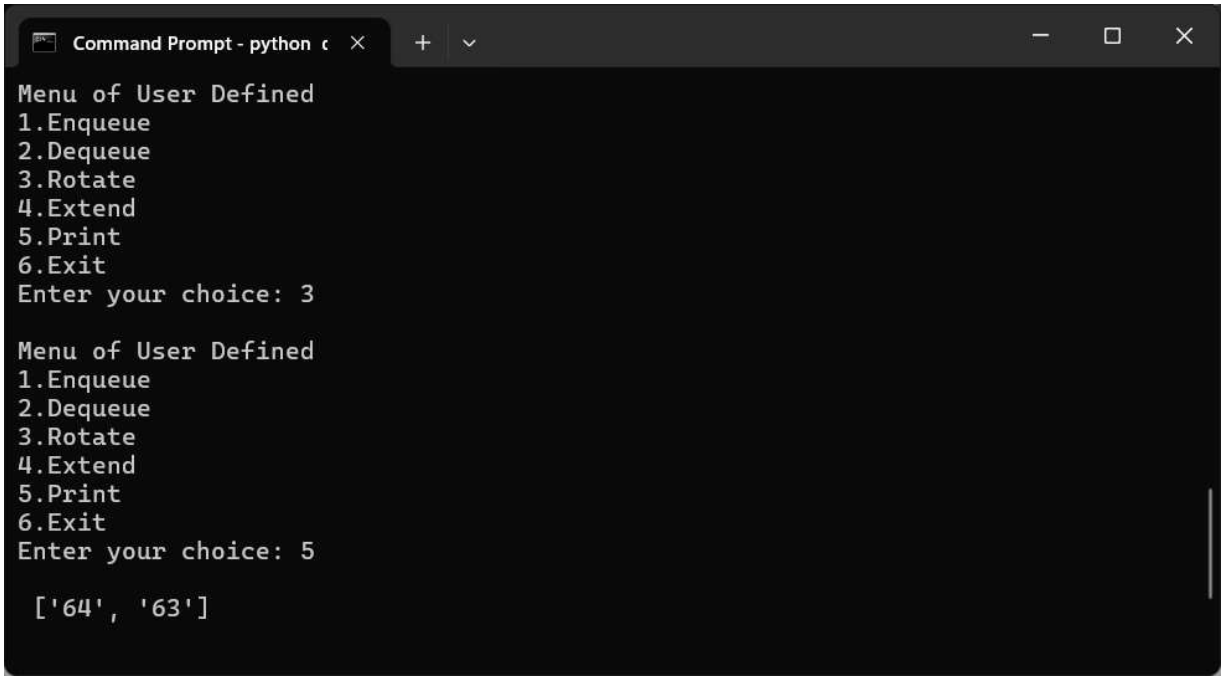
3.Extend

```
Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 4
Enter total number to extend: 2
Enter the elements:
63
64

Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 ['63', '64']
```

4. Rotate



5. Print

```
Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 5

 ['64', '63']
```

6. Invalid Choice

```
Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 63
Invalid choice
```

7. Exit

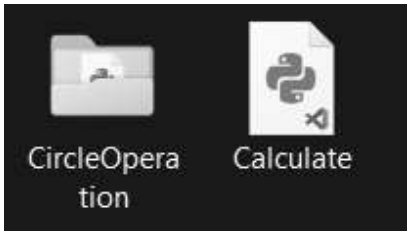
```
Menu of User Defined
1.Enqueue
2.Dequeue
3.Rotate
4.Extend
5.Print
6.Exit
Enter your choice: 6

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>
```

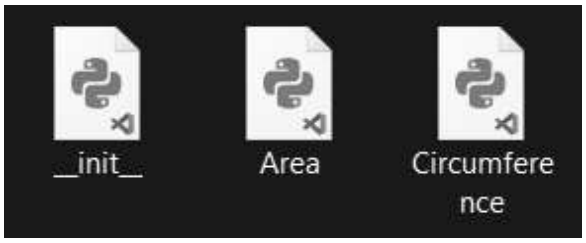
Post-Lab Programs:

Q1.Write a Python program to create any user defined module and call its functionalities in another program.

File Structure:



Root Directory



Inside CircleOperation

Code:

1) Area.py

```
import math

def area(radius):
    return math.pi*(radius**2)
```

2) Circumference.py

```
import math

def circumference(radius):
    return 2*math.pi*radius
```

3) Calcuate.py

```
from CircleOperation import Area,Circumference

radius=int(input("Enter the Radius: "))
print("Area of Circle with radius",radius,"is",Area.area(radius))
print("Circumference of Circle with radius",radius,"is",Circumference.circumference(radius))
```

Output:

```
F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>python Calculate.py
Enter the Radius: 5
Area of Circle with radius 5 is 78.53981633974483
Circumference of Circle with radius 5 is 31.41592653589793

F:\College Stuff\Vishal Mahajan SE IT SEM 4\Python Lab\EXP10>
```