PRACTICAL 8

NAME:SAKSHEE SAWANT ROLL NO.-51 CLASS -D7B

AIM:To understand building modules ,Packages ,Data Structures in python

THEORY: · Package – The package statement switches the current naming context to a specified namespace (symbol table) · Thus,

1) A package is a collection of code which lives in its own namespace.

2) Package enable the construction of modules which ,when used ,wont clobber

variables and functions outside of the modules own namespace.

3) The package stays in effect until either another package statement is invoked or

until the end of the current block of the file. · Module- A module is a reusable package defined in a library file whose name is the same as the name of the package.

1) The function require and use will load a module.

2) Both function require and will call the eval function to process the code. · Data structures in Python-

1) STACK: Stack is a collection of objects inserted and removed in a last in first

out (LIFO). Objects can be inserted onto stack at any time but only the object

inserted last can be accessed or removed which coins the object to the top of the

stack.

BASIC OPERATIONS PERFORMED IN STACK:

a) Push- Adds an item in the stack. If the stack is full then it shows stack

overflow.

b) Pop- Removes an item from the stack. The items are popped in the

reverse order in which they are pushed. It the stack is empty , then it

shows stack underflow.

c) Peek: Returns top element of the stack.

2) OUEUE: Queue works on the principle of ‘FIRST IN- FIRST OUT’. In queue, insertion and deletions are made from the beginning of list.

BASIC OPERATIONS ON QUEUE:

a) Insertion – It inserts the elements at the beginning of the queue.

b) Deletion – It deletes the elements from the start of the queue.

3) LINKED LIST: Linked list is often compared to arrays. Whereas an array is fixed size of sequence, a linked list can have its elements to be dynamically allotted.

CONCLUSION:

1)ModuleNotFoundError: No module named 'stack'

2) Indentation Error:

The statements in the loops and control structure were not indented.

Solution:- we use proper indentation in the statements.

3) Type Error:

We were trying to call a function but it was giving type error-less arguments given.

4)NameError: name 'returnstk' is not defined

Solution:- We provided all arguments required.

AIM: Building Modules, Packages, Data Structures in Python

Create a package which combines modules for data structure for stack and queue. Import the package for the following operations:*

i) Write a program using stack to convert decimal to binary number

```
stk=[]
def push(x):
    stk.append(x)
def pop():
    if len(stk)!=0:
        returnstk.pop()
    else:
        print("EMPTY")
def peek():
    if len(stk)!=0:
        returnstk[-1]
    else:
        print("EMPTY")
def display():
     if len(stk)!=0:
         for i in stk:
             print(i)
     else:
          print("EMPTY")


q=[]
def insert(x):
    q.append(x)
def delete():
```

```
    if len(q)!=0:
        returnq.pop(0)
    else:
        print("EMPTY")
def display():
    for i in q:
        print(i)
from stack import*
c=a=int(input("ENTERADECIMALNO"))
while a!=0:
    push(a%2)
    a=a//2
    display()
```

OUTPUT:

ii) Evaluate arithmetic expression by importing the packages of Stack and

queue

CODE:

```
stk=[]
def push(x):
    stk.append(x)
def pop():
    if len(stk)!=0:
        returnstk.pop()
    else:
        print("EMPTY")
def peek():
    if len(stk)!=0:
        returnstk[-1]
    else:
        print("EMPTY")
def display():
     if len(stk)!=0:
         for i in stk:
             print(i)
     else:
          print("EMPTY")


q=[]
def insert(x):
    q.append(x)
```

```
def delete():
    if len(q)!=0:
        returnq.pop(0)
    else:
        print("EMPTY")
def display():
    for i in q:
        print(i)
from stack import*
pf=input("ENTER A POSTFIX EXPRESSION")
i=0
def eval(pf):
    for i in pf:

        if(i.isdigit()):

            push((int)(i))
        else:

            x=(int)(pop())
            y=(int)(pop())
            if i=='+':
                push(x+y)
            if i=='-':
                push(y-x)
```

```
        if i=='*':
            push(x*y)
        if i=='/':
            push(y/x)
        if i=='^':
            push(y**x)
    if(len(stk)==1):
        print("EVALUATION OF POSTFIX EXPRESSION IS:",peek())
eval(pf)
```

OUTPUT: