```
#Python Practicals
#Practical 1 Basics 1
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
c = a+b
d = a-b
e = a*b
f = a/b
g = b//a
h = a**b
i = b%a
print("Addition of two numbers is: ",c)
print("Subtraction of two numbers is: ",d)
print("Multiplication of two numbers is: ",e)
print("Division of two numbers is: ",f)
print("Floor division of two numbers is: ",g)
print("Exponential of two numbers is: ",h)
print("Modulus of two numbers is: ",i)
```

#Practical 1 Basics 2

```
r = int(input("Radius is : "))
area = r*r*3.14
print("Area of circle is : ",area)
```

Practical 1 Basics 3

```
l = int(input("Enter length : "))
w = int(input("Enter width : "))
area = l*w
print("Area of rectangle : ",area)
```

Practical 1 Basics 4

```
h = int(input("Height : "))
b = int(input("Base : "))
area = 1/2*b*h
print("Area of triangle is ",area)
```

```
#Practical 1 Basics 5
s = int(input("Side of a square : "))
area = s*s
print("Area of a Square : ",area)

Practical 1 Basics 6
r = int(input("Radius is : "))
vol = (4/3)*(3.14)*(r*r*r)
print("Volume of sphere : ",vol)
```

```
#Practical 2 Conditional Statements 1
num1 =  int(input("Enter the first number: "))
num2 =  int(input("Enter the second number: "))
if(num1 > num2):
print("The largest number is ",num1)
elif(num1 < num2):
print("The largest number is ",num2)
else:
print("Both numbers are equal")

#Practical 2 Conditional Statements 2
num1 =  int(input("Enter the first number: "))
num2 =  int(input("Enter the second number: "))
num3 =  int(input("Enter the third number: "))
if(num1 <= num2 and num1 <= num3):
    print("The smallest number is ",num1)
elif(num2 <= num1 and num2 <= num3):
    print("The smallest number is ",num2)
else:
    print("the smallest number is ",num3)
```

```
#Practical 2 Conditional Statements 3
a = int(input("Enter the a number: "))
if(a>0):
    print("positive number")
elif(a<0):
    print("negative number")
else:
    print("Zero")

#Practical 2 Conditional Statements 4
a = int(input("Enter the a number: "))
if( a%3==0 ):
    print("The Number is divisible by three")
else:
    print("The Number is not divisible by three")
```

```
#Practical 2 Conditional Statements 5
a = int(input("Enter the a number: "))
if( a%2==0 ):
    print("Even Number")
else:
    print("Odd Number")

#Practical 2 Conditional Statements 6
a = int(input("Enter the a number: "))
if( a%3==0 ):
    if( a%2==0 ):
        print("The number is even and Divisible by three")
    else:
        print("The number is odd and Divisible by three")
else:
    print("The number is not divisible by three")
```

```
#Practical 3 While Loop 1
n = int(input("Enter the a number: "))
i=1
while(i<=n):
	print(i)
	i+=1

#Practical 3 While Loop 2
n = int(input("Enter the a number: "))
i=1
sum=0
while(i<=n):
	sum+=i
	i+=1
average=sum/n
print("Sum is : ",sum,"\nAverage is : ",average)
```

```
#Practical 3 While Loop 3
n = int(input("Enter the a number: "))
while(n>0):
	print(i)
	n-=1
```

Practical 3 While Loop 4

```
i=1
while(i<=30):
	if(i%3==0):
		print(i)
	i+=1
```

```
#Practical 3 While Loop 5
n = int(input("Enter the a number: "))
i=1
while(i<=n):
	if(i%2==0):
		print(i)
	i+=1
```

```
#Practical 3 While Loop 6
n = int(input("Enter the a number: "))
i=1
sum=0
count=0
while(i<=n):
    if(i%3==0):
        print(i)
        sum+=i
        count+=i
    i+=1

if(count>0):
    average=sum/count
    print("Sum of odd numbers : ",sum,"Average of all numbers : ",average)
else:
    print("No odd numbers found")
```

```
#Practical 3 While Loop 7
n = int(input("Enter the a number: "))
factorial=1
i=1
while(i<=n):
	factorial*=i
	i+=1
print("The Factorial of ",n,"is : ",factorial)
```

```
#Practical 4 For Loop 1

n = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
for i in n:
        print(i)


#Practical 4 For Loop 2

n = int(input(“Enter a number: “))
for i in range(1,n):
        print(i)


#Practical 4 For Loop 3

n = int(input(“Enter a number: “))
for i in range(1,n+1):
        if(i%2 == 0)
                print(i)
        else:
                continue
```

```
#Practical 4 For Loop 4
n = int(input("Enter a number: "))
for i in range(num, 0, -1):
    if(i%2 == 1):
        print(i)
    else:
        continue

#Practical 4 For Loop 5
fruit = "apple"
for i in fruit:
    print(i)

#Practical 4 For Loop 6
f = int(input("Enter a number: "))
for i in range(1,f):
    if(i>=2):
        f*=i
    else:
        print(1)
print(f)
```

#Practical 4 For Loop 7.a

```
n = int(input(“Enter rows: “))  #5
for i in range(1,n+1):
    for j in range(1, i):
        print(j, end = “ “)
    print(“ “)
```

#Practical 4 For Loop 7.b

```
n = int(input(“Enter rows: “))  #5
for i in range(1,n+1):
    for j in range(1, i):
        print(i, end = “ “)
    print(“ “)
```

```
#Practical 4 For Loop 7.c
n = int(input(“Enter rows: “))  #5
count = 1
for i in range(1,n+1):
    for j in range(1, i+i):
        print(count, “ “, end = “ “)
        count += 1
    print(“ “)
```

```
#Practical 5 Functions 1
def add(a, b):
    c = a+b
    return c
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
sum = add(a,b)
print("Addition of : ", a , " + ", b , " = ", sum)
```

```
Practical 5 Functions 2
def area(r):
    area = r*r*(22/7)
    return area
r = int(input("Enter the radius: "))
Area = areaR(r)
print("Area of circle : ", Area , " sq.units ")
```

```
#Practical 5 Functions 3
def largest(a, b, c):
    return max(a, b, c)
a = int(input("Enter the 1st number: "))
b = int(input("Enter the 2nd number: "))
b = int(input("Enter the 3rd number: "))
large = largest(a, b, c)
print("The largest number from ", a , " , ", b , " , " ,
" and ", c , " is " , large)

#Practical 5 Functions 4
def fact(n):
    res = 1
    for i in range(1, n+1):
        res *= i
    return res
num= int(input("Enter the Number: "))
factorial = fact(num)
print("Factorial of ", num , " = ", factorial)
```

```
#Practical 5 Functions 5
def factorial(n):
    if (n<=1):
        return 1
    else:
        return n*factorial(n-1)

def nCr (n, r)
    n1 = factorial(n)
    r1 = factorial(r)
    c = n1 / r1 * (factorial(n-r))
    return c
Solution = nCr(5,2)
print("nCr = ", Solution)
```

```
#Practical 5 Functions 6
def factorial(n):
    if (n<=1):
        return 1
    else:
        return n*factorial(n-1)

num= int(input(“Enter a Number: “))
fact = factorial(num)
print(“Factorial of “, num , “ is “, fact)
```

```
#Practical 6 Regular Expression 1

import re

mystr = input(“Enter a line: “)

myptr = input(“Enter a pattern: “)

      x = re.findall(myptr, mystr)

print(“The pattern present in my string is = “)
print(x)


#Practical 6 Regular Expression 2

import re

mystr = input(“Line: “)

x = re.search(“\s”, mystr)

print(“The first white space character is located at
position= “, x.start())


#Practical 6 Regular Expression 3

import re

mystr = input(“Enter a Line: “)

x = re.split(“\s”, mystr)

print(“The entered text is splitted as: “)
print(x)
```

```
#Practical 6 Regular Expression 4

import re

mystr = input(“Enter a Line: “)

x = re.sub(“\s”, “ 3 “ , mystr)

print(“The input string after sub function is = “)
print(x)
```

```
#Practical 7 File Handling 1
with open(“textfile.txt” , “r” ) as f:
        data = f.read()
print(data)

#Practical 7 File Handling 2
with open(“days.txt” , “r” ) as f:
        for day in f:
                print(data)

#Practical 7 File Handling 3
f = open(“cities.txt” , “ r “)
city = f.read()
f.close()
print(“Cities: “)
print(city)
```

```
#Practical 7 File Handling 4
with open(country.txt” , “r” ) as f:
    while true:
        line = f.readline()
        if not line:
            break
        print(line, end = “*”)

#Practical 7 File Handling 5
with open(“friends.txt” , “w” ) as f:
    for i in range(5):
        name = input(“Name: “)
        f.write(name + ‘\n’)
```

pract8 - 1: Write a python program that implements the stack using list .

```
mystack = []
print("Stack")
print(mystack)
# append()
mystack.append('a')
mystack.append('b')
mystack.append('c')

# pop()
print("The Elements popped out of the stack :")
print(mystack.pop())
print("The Elements popped out of the stack :")
print(mystack.pop())
print("The Elements popped out of the stack :")
print(mystack.pop())

print("The elements finally in the stack :")
print(mystack)
```

pract8 - 2 : Write a python program that implements Stack using Function.

```
def create():
    mystack = list()
    return mystack

def isempty(mystack):
    return len(mystack) == 0

def push(mystack,n):
    mystack.append(n)
    print("pushed item : ",n)

def pop(mystack):
    if(isempty(mystack)):
        return "mystack is empty"
    else:
        return mystack.pop()

def show(mystack):
```

```
    print("The elements in the stacks are :")
    for i in mystack:
        print(i)

mystack = create()
push(mystack,str(10))
push(mystack,str(20))
push(mystack,str(30))
push(mystack,str(40))

print("The elements in stack is :")
show(mystack)

print("The poped element in stack is : ",pop(mystack))
show(mystack)
print("The poped element in stack is : ",pop(mystack))
show(mystack)
print("The poped element in stack is : ",pop(mystack))
show(mystack)
```

pract8 - 3 : Write a python program that implements Stack using class.

```
class myStack:
    def __init__(self):
        self.items = list()
        self.size  = 0

    def isEmpty(self):
        if(self.size == 0):
            return True
        else:
            return False

    def length(self):
        if self.isEmpty():
            print("Stack is Empty")
        else:
            print("Number of Elements in List are :
",self.size)

    def peek(self):
```

```
        if self.isEmpty():
            print("Stack is Empty")
        else:
            print("Top most Element :",self.items[self.size -
1])

    def pop(self):
        if self.isEmpty():
            print("Stack is Empty")
        else:
            print("The popped element is :",self.items.pop())
            self.size -= 1

    def push(self,item):
        self.items.append(item)
        self.size += 1

    def show(self,items):
        print("Stack :",self.items)
```

```
s = myStack()
print("MENU:")
print("1 = push operation")
print("2 = pop operation")
print("3 = peek operation")
print("4 = length operation")
print("5 = Show operation")
print("0 = Quit")

ch = -1
while(ch != 0 or s.size <= len(s.items)):
    ch =int(input("Choose an Option :"))
    if ch == 1:
        value = int(input("Enter value:"))
        s.push(value)
    elif ch == 2:
        s.pop()
    elif ch == 3:
        s.peek()
    elif ch == 4:
```

```
    s.length()
elif ch == 5:
    s.show(s.items)
elif ch == 0:
    print("End of Program.")
    break
```

#pract9-1 :Write a python program that implements queue using list with numeric type of data

Coding:

```
class Queue:
    def _init_(self):
        self.queue = []

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

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

    def peek(self):
        if not self.is_empty():
```

```
            return self.queue[0]
        else:
            return None

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

# Example usage with numeric data:
queue = Queue()
queue.enqueue(10)
queue.enqueue(20)
queue.enqueue(30)

print(queue.peek())  # Output: 10

while not queue.is_empty():
    print(queue.dequeue())  # Output: 10 20 30
```

#pract9-2 :Write a python program that implements queue using list with string type of data

Coding:

```
class Queue:
    def _init_(self):
        self.queue = []

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

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

    def peek(self):
        if not self.is_empty():
```

```
            return self.queue[0]
        else:
            return None

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

# Example usage with string data:
queue = Queue()
queue.enqueue("hello")
queue.enqueue("world")
queue.enqueue("!")

print(queue.peek())  # Output: hello

while not queue.is_empty():
    print(queue.dequeue())  # Output: hello world !
```

```
# pract9 - 3 : Write a python program that implements queue using class with different methods - enqueue, dequeue, display, size
"""
- where ,
enqueue is inserting element in queue,
dequeue is removing element from queue,
display is showing the queue elements,
size is giving the number of elements in queue.
"""

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

    # Add an Element:
    def enqueue(self,item):
        self.queue.append(item)

    # Remove an Element:
    def dequeue(self):
```

```
        if len(self.queue) < 1:
            return None
        return self.queue.pop(0)

    # Display the queue:
    def display(self):
        print(self.queue)

    # Size of queue:
    def size(self):
        return len(self.queue)

q =Queue()

cnt = int(input("Enter how many element to insert in queue : "))
for i in range(1 , cnt + 1):
    x = int(input("Enter actual Element : "))
    q.enqueue(x)
```

```
print("Elements in the queue are : ")
q.display()

s = q.size()
print("Total elements in queue are =",s)

elem = q.dequeue()
print("The removed element from queue is ",elem)

print("After removing an element the queue is ")
q.display()
```

pract9 - 4 : Write a python program that implements queue using collection.dequeue

```
from collections import deque
myq = deque()

print("Elements are insert in queue : ")
myq.append(100)
myq.append(200)
myq.append(300)
myq.append(400)
myq.append(500)

print("The elements in queue are :")
print(myq)
# removes first element from queue
x = myq.popleft()

print("The removed element from queue is =",x)
print("The queue elements are = ",myq)
```

```
# pract9 - 5 : Write a python program that implements queue using 'queue.Queue'

import queue

q = queue.Queue()

c = int(input("Enter how many elements to insert in queue : "))

for i in range(1 , c + 1):

    x = int(input("Enter element to insert in queue : "))

    q.put(x)


q.put(11)

q.put(12)

q.put(13)

q.put(14)

q.put(15)

print("The queue elements are : ")

print(q.queue)


x = q.get()

print("Removed element from queue is = ",x)
```

```
print("The rest element in queue are = ",q.queue)

x = int(input("Enter element to insert in queue : "))
q.put(x)
print("Final queue is = ",q.queue)

x = q.get()
print("Removed element from queue is = ",x)
print("The rest element in queue are = ",q.queue)
```

pract10 - 1. Write a Python Program to implement singly Linked List using class.

```
class node:
    def __init__(self,data,next=None):
        self.data = data
        self.next = next

n1 = node(10)
n2 = node(20)
n3 = node(30)
n4 = node(40)

n1.next = n2
n2.next = n3
n3.next = n4

print("The Linked List created is:")
curr = n1
while curr:
    print(curr.data,end=" -> ")
```

```
    curr = curr.next
print(None)
```

pract10-2. Write a Python Program that implements linked list and insert a new node at the beginning.

```
class node:
    def __init__(self,data,next=None):
        self.data = data
        self.next = next

n1 = node(10)
n2 = node(20)
n3 = node(30)
n4 = node(40)

n1.next = n2
n2.next = n3
n3.next = n4

print("The Linked List created is:")
curr = n1
while curr:
```

```
    print(curr.data,end=" -> ")
    curr = curr.next
print(None)

# new node insert at beginning
head = n1
newnode = node(int(input("Enter new element: ")))
newnode.next = head
head = newnode

while head:
    print(head.data,end=" -> ")
    head = head.next
print(None)
```

pract10 - 3. Write a Python Program that implements linked list and insert a new node at the end.

```
class node:
    def __init__(self,data,next=None):
        self.data = data
        self.next = next

n1 = node(10)
n2 = node(20)
n3 = node(30)
n4 = node(40)

n1.next = n2
n2.next = n3
n3.next = n4

print("The Linked List created is:")
curr = n1
while curr:
```

```
    print(curr.data,end=" -> ")
    curr = curr.next
print(None)

# node insert at end
head = n1
newn = node(int(input("Enter new element:")))
current = head
while current.next:
    current = current.next
current.next = newn

print("The Linked List after inserting at end:")
curr = n1
while curr:
    print(curr.data,end=" -> ")
    curr = curr.next
print(None)
```

pract10-4. Write a Python Program that implements linked list and insert a new node in between.

```
class node:
    def __init__(self,data,next=None):
        self.data = data
        self.next = next

n1 = node(10)
n2 = node(20)
n3 = node(30)
n4 = node(40)

n1.next = n2
n2.next = n3
n3.next = n4

print("The Linked List created is:")
curr = n1
while curr is not None:
```

```
    print(curr.data,end=" -> ")
    curr = curr.next
print(None)


head = n1
elemnew = int(input("Enter new Element to insert in between : "))
newnd = node(elemnew)


elemafter = int(input("Enter element after which to insert : "))


current = head
while current is not None and current.data != elemafter:
    current = current.next


if current is None:
    print("Not Found.")
else:
    newnd.next = current.next  #doubt
```

```
    current.next = newnd


print("The resultant linked list is : ")
current = head
while current is not None:
    print(current.data , end = " -> ")
    current = current.next
print(None)
```