PYTHON CODES

—>Basic problems

```
1.#ifelse loop
height=float(input())
weight=int(input())
gender=input()
if(height>5.8):
    if(weight<50):
        if(gender=="female"):
            print("selected")
        else:
            print("not selected")
    else:
            print("not selected")
else:
            print("not selected")




2.#leapyear
year=int(input())
if year%4==0:
    if year%100==0:
        if year%400==0:
            print("it's a leap year")
        else:
            print("it's not a leap year")
    else:
            print("it's a leap year")
else:
```

```
            print("it's not a leap year")
```

```
3.#forlopp(prime)
n=int(input())
for i in range(2,n):
    if n%i==0:
        print("it is not a prime")
        break
else:
    print("it is a prime")
```

```
4.#forloopusing flag(prime)
n=int(input())
flag=0
for i in range(2,n//2):
    if n%i==0:
       flag=1
       break
if flag==1:
    print("not prime")
else:
    print("it is a prime")
```

```
5.#gcd
a=int(input())
b=int(input())
m=min(a,b)
g=1
```

```
for i in range(1,m+1):
    if a%i==0 and b%i==0:
        g=i
print(g)
```

```
6.#lcm
a=int(input())
b=int(input())
m=max(a,b)
l=m
for i in range(m,(a*b)+1):
    if i%a==0 and i%b==0:
        l=i
        break
print(l)
```

```
7.#rev a num
n=int(input())
rev=0
while n>0:
    rem=n%10
    rev=rev*10+rem
    n=n//10
print(rev)
```

```
8.#sum of digits in a nums
n=int(input())
sum=0
while n>0:
```

```
    rem=n%10
    sum=sum+rem
    n=n//10
print(sum)



9.#mul each digitd in a num
n=int(input())
mul=1
while n>0:
    rem=n%10
    mul=mul*rem
    n=n//10
print(mul)



10.#palindrome of a num
n=int(input())
m=n
temp=0
while n>0:
    rem=n%10
    temp=temp*10+rem
    n=n//10
print(temp)
if m==temp:
    print("It is palindrome")
else:
    print("not a palindrome")



11.#count the np.of digitd in a num
n=int(input())
```

```
m=n
count=0
while n>0:
    rem=n%10
    count=count+1
    n=n//10
print(count)
```

12.#finding no.of even and odd digits in anum

```
n=int(input())
count=0
even=0
odd=0
while n>0:
    rem=n%10
    if rem%2==0:
        even=even+1
    else:
        odd=odd+1
    n=n//10
print(even)
print(odd)
```

13.factorial of a num

```
n=int(input())
fact=1
for i in range(1,n+1):
    if n!=0:
        fact=fact*i
print(fact)
```

14.factorial of a num(reverse)

```
n=int(input())
fact=1
for i in range(n,0,-1):
    if n!=0:
        fact=fact*i
print(fact)


15.#fibonacci series
n=int(input())
a=0
b=1
print(a,b,end=" ")
for i in range(3,n+1):
    c=a+b
    print(c,end=" ")
    a=b
    b=c
```

—>Pattern programs

```
1.#matrix type pattern1
n=int(input())
for i in range(1,n+1):
    for j in range(1,n+1):
        print(j,end=" ")
    print()


2.#patter2
n=int(input())
for i in range(1,n+1):
    for j in range(i,n+i):
        print(j,end=" ")
    print()
```

```
3.#patter3
n=int(input())
for i in range(1,n+1):
    for j in range(i,n+i):
        print(chr(64+j),end=" ")
    print()


4.#pattern4
n=int(input())
for i in range(1,n+1):
    for j in range(0,n):
        print(chr(64+j),end=" ")
    print()

5.#pattern5
n=int(input())
for i in range(1,n+1):
    for j in range(1,i+1):
        print(j,end=" ")
    print()

6.#patter6
n=int(input())
for i in range(1,n+1):
    for j in range(i,i+i):(or)for j in range(i,2*i):
        print(j,end=" ")
    print()


7.#pattern7
n=int(input())
for i in range(n):
    for j in range(n,i,-1):
```

```
        print(j,end=" ")
    print()


8.#patter8
n=int(input())
for i in range(n):
    for j in range(n):
        print("*",end=" ")
    print()



9.#patter9
n=int(input())
for i in range(n):
    if i==0 or i==n-1:
        for j in range(n):
            print("*",end=" ")
        print()
    else:
        for j in range(n):
            if j==0 or j==n-1:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()

10.#patter10
n=int(input())
for i in range(n):
    if i==0 or i==1 or i==n-1 or i==n-2:
        for j in range(n):
            print("*",end=" ")
        print()
    else:
```

```
        for j in range(n):
            if j==0 or j==1 or j==n-1 or j==n-2:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
```

11.#patter11

```
n=int(input())
for i in range(n):
    if i==0 or i==1 or i==n-1 or i==n-2:
        for j in range(n):
            print("*",end=" ")
        print()
    elif i==n//2:
        for j in range(n):
            if j==0 or j==n//2 or j==n-1:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
    else:
        for j in range(n):
            if j==0 or j==1 or j==n-1 or j==n-2:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
```

12.#pattern12

```
n=int(input())
for i in range(1,n+1):
    for j in range(1,n-i+1):
```

```
        print(" ",end=" ")
    for j in range(0,2*i-1):
        print("*",end=" ")
    print()


13.#pattern13
n=int(input())
for i in range(1,n+1):
    if i==1 or i==n:
         for j in range(1,n-i+1):
             print(" ",end=" ")
         for j in range(0,2*i-1):
             print("*",end=" ")
         print()
    else:
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            if j==0 or j==2*i-2:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()


14.#pattern14
n=int(input())
for i in range(1,n+1):
         for j in range(1,n-i+1):
             print(" ",end=" ")
         for j in range(0,2*i-1):
             print("*",end=" ")
         print()
for i in range(n-1,0,-1):
         for j in range(1,n-i+1):
```

```
            print(" ",end=" ")
         for j in range(0,2*i-1):
            print("*",end=" ")
         print()



15.#pattern15
n=int(input())
for i in range(1,n+1):
    if i==1:
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        print()
    else:
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            if j==0 or j==2*i-2:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
for i in range(n-1,0,-1):
    if i==1:
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        print()
    else:
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
```

```
            if j==0 or j==2*i-2:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()



16.#pattern16
n=int(input())
for i in range(1,n+1):
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        for j in range(1,(n-i+1)*2):
            if j!=(n-i+1)*2-1:
                    print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        print()




17.#pattern17
n=int(input())
for i in range(1,n+1):
        for j in range(1,n-i+1):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        for j in range(1,(n-i+1)*2):
            if j!=(n-i+1)*2-1:
                    print(" ",end=" ")
        for j in range(0,2*i-1):
```

```
            print("*",end=" ")
        print()
for i in range((2*n-1),0,-1):
        for j in range(1,(2*n)-i):
            print(" ",end=" ")
        for j in range(0,2*i-1):
            print("*",end=" ")
        print()

18.#pattern18
n=int(input())
for i in range(n):
    if i==0 or i==n-1:
        for j in range(n):
            print("*",end=" ")
        print()
    else:
        for j in range(n):
            if j==0 or j==n-1:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
for i in range(1,n):
    if  i==n-1:
        for j in range(n):
            print("*",end=" ")
        print()
    else:
        for j in range(n):
            if j==0 or j==n-1:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
```

```
19.#pattern19
n=int(input())
for i in range(1,2*n):
    if i==1 or i==(2*n)//2:
        for j in range(n+1):
            print("*",end=" ")
        print()
    else:
        for j in range(n+1):
            if j==0 or j==n:
                print("*",end=" ")
            else:
                print(" ",end=" ")
        print()
```

—>List programs

```
1.#reversinglist
l=[1,2,3,4,5]
print(l[::-1])


2.#reversinglist
l=[1,2,3,4,5]
#print(l[::-1])
for i in range(len(l)-1,-1,-1):
    print(l[i])


3.#finding largest and smallest in a list
l=[5,6,3,4,7]
```

```
m=l[0]
for i in l:
    if i>m:(largest)  if i<m:(smallest)
        m=i
print(m)



4.#finding second largest and smallest in a list
#list
l=[5,6,3,4,7]
p=set(l)
q=list(p)
print(q[1])(largest)print(q[-2])(smallest)


5.#removing duplicate elements in a list
l=[1,1,2,3,3,4,5,5]
room=[]
for i in l:
    if i not in room:
        room.append(i)
print(room)


5.#counting no.of elemts in a list
l=[1,1,2,3,3,4,5,5]
room=[]
for i in l:
    if i in room:
        room.append(i)
    else:
        if i in l:
            count=0
            for j in l:
                if i==j:
```

```
                    count+=1
        print(room,"-",count)


6.#find similr elements in two lists
l=[1,2,3,4]
l2=[4,5,6,7,2]
for i in l:
    for j in l2:
        if i==j:
            print(i)



7.#printing non-similar eleemnts in alist
l1=[1,2,3,4]
l2=[4,5,6,7]
l=[]
for i in l1:
    if i not in l2:
      l.append(i)
for i in l2:
   if i not in l1:
      l.append(i)
print(l)
```

—>Set problems

```
1.#counting no.of elemts in a set
l=[1,1,2,3,3,4,5,5]
s=set(l)
room=[]
for i in s:
```

```
    if i in l:
        count=0
        for j in l:
            if i==j:
                count+=1
        print(i,"-",count)
2.#sum of all numbers in aset
l=[1,2,3,4]
s=set(l)
even=0
odd=0
for i in s:
    if i%2==0:
        even+=i
    else:
        odd+=i
print("sum of even numbers ",even)
print("sum of odd numbers ",odd)
```

—>Different topic codes

```
1.#finding prime numbers in a range
n=int(input())
m=int(input())
for i in range(n,m+1):
    if i%2!=0:
        print(i," is a prime number")



2.#using def
def isprime(x):
    for i in range(2,x//2+1):
```

```
        if x%i==0:
            return 0
        else:
            return 1
n=int(input())
m=int(input())
for i in range(n,m+1):
    a=isprime(i)
    if a==1:
        print(i)

3.#convert decimal to bimary
n=int(input())
s=""
while n>0:
    rem=n%2
    s=s+str(rem)
    n=n//2
print(s[::-1])



4.#finding perfect square
n=int(input())
i=1
while i**2<n:
    i+=1
if i**2==n:
    print(n," is a perect square")
else:
    print(n," is not a perfect square")
```

```
5.#power of 2
n=int(input())
i=1
while 2**i<n:
    i+=1
if 2**i=n:
    print(n," is a poweer of 2")
else:
    print(n," is not a power of 2")
```

```
6.#armstrong
n=int(input())
d=len(str(n))
a=0
t=n
while t>0:
    rem=t%10
    a=a+(rem**d)
    t=t//10
if a==n:
    print("armstrong")
else:
    print("not an armstrong")
```

```
7.#perfect number
n=int(input())
r=0
i=1
for i in range(1,n):
        if n%i==0:
```

```
                r=r+i
if r==n:
        print("it is a perfect number")
else:
        print("not a perfect number")
```

```
8.#slicing
s=[1,2,3,4,5,6,7,8,9,10]
l=[]
even=s[1:11:2]
odd=s[-2::-2]
l.extend(even)
l.extend(odd)
print(l)
```

```
9.#slicing(2)
s=[1,2,3,4,5,6,7,8,9,10]
l=s[2::3]+s[1::2]
print(l)
```

```
10.#leftrotating
n=2
s=[1,2,3,4,5]
a=s[-n::]+s[0:-n]
print(a)
```

```
11.#rightrotation
n=3
s=[1,2,3,4,5]
```

```
a=s[n::]+s[0:n]
if n>len(a):
        n=n%len(a)
print(a)
```

12.whether a number is vald or not

```
n=str(input())
if len(n)==10:
     if n[0]=='6' or n[0]=='7' or n[0]=='8' or n[0]=='9':
        if n.isdigit():
                print("It is a valid number")
                exit
        else:
             print("invalid")
else:
    print("It is not a valid number")
```

13.#bankaccount

```
def withdraw(account, amount):
    if amount > account['balance']:
        print("Insufficient funds!")
    else:
        account['balance'] -= amount
        account['transactions'].append(f"Withdrawal: ${amount}"
        print(f"Withdrawal successful. Remaining balance: ${acc

def deposit(account, amount):
    account['balance'] += amount
    account['transactions'].append(f"Deposit: ${amount}")
    print(f"Deposit successful. Remaining balance: ${account['ba

def get_balance(account):
    return account['balance']
```

```
def get_transaction_history(account):
    return account['transactions']

# Create an account dictionary
account = {
    'balance': 1000,
    'transactions': []
}

# Dictionary to map user choices to functions
choices = {
    '1': deposit,
    '2': withdraw,
    '3': get_balance,
    '4': get_transaction_history
}

while True:
    print("\n1. Deposit")
    print("2. Withdraw")
    print("3. Check Balance")
    print("4. Transaction History")
    print("5. Exit")

    choice = input("Enter your choice: ")

    if choice == '5':
        print("Exiting program.")
        break

    if choice in choices:
        if choice == '1' or choice == '2':
            amount = float(input("Enter amount: "))
            choices[choice](account, amount)
        else:
```

```
            print(choices[choice](account))
    else:
        print("Invalid choice. Please try again.")



14.#reversing a sentence
s="hey how are you"
l=list(s.split(" "))
l=l[::-1]
ans=" ".join(l)
print(ans)



15.#reversig the words in a sentence
s="hey how are you"
l=list(s.split(" "))
s=""
for i in l:
    s=s+i[::-1]+" "
print(s)



16.#checking if a string is a palindrome or not
n="Madan"
n1=n.upper()
temp=0
temp=n1[::-1]
if n1==temp:
    print("It is palindrome")
else:
    print("not a palindrome")
```

```
17.#Anagram
a=input()
b=input()
s=set(a)
s2=set(b)
if s==s2:
    for i in s:
        if a.count(i)!=b.count(i):
            print("not anagram")
            break
    else:
        print("ANAGRAM")


18.#class poegram ex
class Mango:
    def _init_(self):
        print("this is what?")
    def balaji(self):
        print("this is without para")
    def shetty(self,a,b):
        print(a+b,"this is with para")
    def magicalprime(self,a):
        print('check it magical primr or not')
man=Mango()
man.balaji()
man.shetty(10,20.5)
man.magicalprime(102)



19.#recursion
def recur():
    print("hey")
```

```
    recur()
recur()
```

```
20.#recursion
def recur(x):
    print(x)
    x=x+1
    recur(x)
recur(1)
```

```
21.#factorial of a nnum using recursion
def fact(x):
    if x==1:
        return x
    return x*fact(x-1)
n=int(input())
a=fact(n)
print(a)
```

```
22.#factorial of a nnum using recursion(reverse order)
def fact(a,x):
    if a==x:
        return x
    return a*fact(a+1,x)
n=int(input())
a=fact(1,n)
print(a)
```

```
23.#fibonacci sseries using recursion
def fib(x):
    if x<=1:
        return x
    return fib(x-1)+fib(x-2)
n=int(input())
a=fib(n)
print(a)


24.#reversing a num using recursion
def rev(x,res):
    if x<=0:
        return res
    rem=x%10
    res=res*10+rem
    return rev(x//10,res)
n=int(input())
res=0
a=rev(n,res)
print(a)




25.#finding if the num is power of 3 or not
def pow(i,x):
    if 3**i==x:
        return True
    elif 3**i>=x:
        return False
    else:
        return pow(i+1,x)
n=int(input())
a=pow(1,n)
```

```
print(a)


26.#class example
class person:
    nickname="chintu"
    roll="56"
    height="6"
    def run(self):
        print("i can run"  +self.nickname  +self.roll)
harsha=person()
harsha.run()



27.#constructor exmaple
class person:
    def __init__(self,x,y,z):
        self.nickname=x
        self.roll=y
        self.height=z
    def run(self):
        print("i can run",self.nickname,self.roll)
harsha=person("chintu",78,6)
harsha.run()



28.#Abstrct method and class
class mobile:
    def functions(self):
        pass
class iphone(mobile):
    def functions(self):
        print("Hey!! i am IPHONE")
class samsung(mobile):
```

```
    def functions(self):
        print("Hey!! i am SAMSUNG")
iphone13=iphone()
iphone13.functions()
samsungm31s=samsung()
samsungm31s.functions()



29.#encapsulation
class car:
    _engine="v8"
    _wires="blue"
    def getter(self):
        print(self._engine)
        print(self._wires)
    def setter(self,engine,wires):
        self._engine=engine
        self._wires=wires
bmw=car()
bmw.setter("v9","red")
bmw.getter()



30.#single iheritance
class parents:
    def coolness(self):
        print("parents are cool")
class child(parents):
            def coding(self):
              print("i know coding")
rahul=child()
rahul.coolness()
rahul.coding()
```

```
31.#multilevel inherutance
class parents:
    def coolness(self):
        print("parents are cool")
class child(parents):
            def coding(self):
              print("i know coding")
class child2(child):
      def dancing(self):
            print("i know dancing")
rahul=child2()
rahul.coolness()
rahul.coding()
rahul.dancing()


32.#multiple inheritance
class parents:
    def coolness(self):
        print("parents are cool")
class child:
            def coding(self):
              print("i know coding")
class child2(child,parents):
      def dancing(self):
            print("i know dancing")
rahul=child2()
rahul.coolness()
rahul.coding()
rahul.dancing()


33.#hierariachial in heritance
class parents:
    def coolness(self):
```

```
        print("parents are cool")
class child(parents):
            def coding(self):
              print("i know coding")
class child2(parents):
      def dancing(self):
            print("i know dancing")
rahul=child2()
anjali=child()
rahul.coolness()
anjali.coding()
rahul.dancing()


34.#hybrid inheritance
class parents:
    def coolness(self):
        print("parents are cool")
class child(parents):
            def coding(self):
              print("i know coding")
class child2(parents):
      def dancing(self):
            print("i know dancing")
class child3(child,child2):
      def singing(self):
            print("I can sing")
sahethi=child3()
sahethi.coolness()
sahethi.coding()
sahethi.dancing()
sahethi.singing()


35.#overriding
class addition:
```

```
    def add(self,x,y):
        print(x+y)
class child(addition):
    def add(self,x,y,z):
        print(x+y+z)
i=child()
i.add(5,6,7)
```

```
36.#linear search
l=[1,2,3,4,2,3,2,4,5,4,4]
for i in l:
    if i==4:
        print("found")
        break
else:
    print("not found")
```

```
37.#sorting
l=[1,2,3,4,2,3,2,4,5,4,4]
l.sort()
print(l)
```

```
38.#binary search
l=[1,2,3,4,2,3,2,4,5,4,4]
l.sort()
i=0
s=2
j=len(l)-1
while i<j:
    mid=(i+j)//2
    if l[mid]==s:
```

```
        print(mid, "found")
        break
    elif l[mid]>s:
        j=mid-1
    else:
        i=mid+1
else:
    print("not found")



39.#bubble sort
l=[1,2,3,4,2,3,2,4,5,4,4]
for i in range(0,len(l)-1):
    for j in range(0,len(l)-i-1):
        if l[j]<l[j+1]:
            l[j],l[j+1]=l[j+1],l[j]
print(l)


40.#selection sort
b=[6,5,4,3,2]
for i in range(0,len(b)-1):
    m=i
    for j in range(i+1,len(b)):
        if b[m]>=b[j]:
            m=j
    b[i],b[m]=b[m],b[i]
print(b)



41.#insertion sort
b=[6,5,4,3,2]
for i in range(1,len(b)):
    j=i-1
```

```
    a=b[i]
    while j>=0 and b[j]>a:
        b[j+1]=b[j]
        j-=1
    b[j+1]=a
print(b)
```

```
42.#merge sort
def merge(arr,beg,mid,end):
     n1=mid-beg+1
     n2=end-mid#temp array size
     i=j=0
     left=arr[:mid+1]
     right=arr[mid+1:end+1]
     k=beg
     while i<n1 and j<n2:
        if left[i]<right[j]:
            arr[k]=left[i]
            i+=1
        else:
             arr[k]=right[j]
             j+=1
        k+=1
    while i<n1:
          arr[k]=arr[i]
          k+=1
          i+=1
    while j<n2:
          arr[k]=arr[j]
          k+=1
          j+=1
def mergesort(arr,beg,end):
    if beg<end:
```

```
        mid=(beg+end)//2
        #recursion takes place
        mergesort(arr,beg,mid)
        mergesort(arr,mid+1,end)#left part
        merge(arr,beg,mid,end)
a=[8,7,6,4,3,2,1]
b=0
e=len(a)-1
mergesort(a,b,e)
print(a)



43.#quick sort
def partition(arr,low,high):
    pivot=arr[low]
    start=low+1
    end=high
    while True:
        while start<=end and arr[start]<=pivot:
            start+=1
        while start<=end and arr[end]>pivot:
            end-=1
        if start<end:
            arr[start],arr[end]=arr[end],arr[start]
        else:
            break
    arr[low],arr[end]=arr[end],arr[low]
    return end
def quicksort(arr,beg,end):
    if beg<end:
        p=partition(arr,beg,end)
        quicksort(arr,beg,p-1)
        quicksort(arr,p+1,end)
a=[8,7,6,4,5,2,1,3]
b=0
```

```
e=len(a)-1
quicksort(a,b,e)
print(a)




44.#stack
class stack:
    def _init_(self):
        self.top=-1
        self.size=5
        self.list=[]
    def push(self,data):
        if len(self.list)==5:
            print("Full Stack")
            return 0
        self.top+=1
        self.list.append(data)
    def pop(self):
        if len(self.list)==0:
            print("Empty stack")
            return 0
        self.top-=1
        self.list.pop()
    def peek(self):
        print(self.list)
        if len(self.list)==0:
            print("Empty stack")
            return 0
        elif self.top>5:
            print("Out of index")
        else:
            print(self.list[self.top])
s=stack()
s.push(1)
```

```
s.pop()
s.push(2)
s.push(3)
s.push(4)
s.push(7)
s.peek()
```

```
45.#queue
class queue:
    def __init__(self):
        self.front=self.rear=-1
        self.size=5
        self.list=[]
    def enque(self,data):
        if len(self.list)==5:
            print("full queue")
            return 0
        self.rear+=1
        self.list.append(data)
        if self.front==-1:
            self.front+=1
    def dqueue(self):
        if len(self.list)==0:
            print("empty stack")
            return 0
        self.list.pop()
        self.front-=1
    def display(self):
        if len(self.list)==0:
            print("empty stack")
            return 0
        print(self.list)
q=queue()
```

```
q.enque(1)
q.dqueue()
q.enque(2)
q.dqueue()
q.enque(3)
q.dqueue()
q.enque(4)
q.dqueue()
q.enque(5)
q.dqueue()
q.display()
```

```
46.#evaluation of postfix expression
l="5678+-*"
a=[]
for i in l:
    if i.isdigit():
        a.append(int(i))
    else:
        op2=a.pop()
        op1=a.pop()
        if i=="+":
            a.append(op1+op2)
        elif i=="-":
            a.append(op2-op1)
        elif i=="*":
            a.append(op1*op2)
        elif i=="/":
            a.append(op1/op2)
print(a)
```

```
47.#linked list
class Node:
    def __init__(self,value):
        self.data=value
        self.next=None
class linkedlist:
    def __init__(self):
        self.head=None
    def insertatbeg(self,value):
        newnode=Node(value)
        if self.head==None:
            self.head=newnode
        else:
            newnode.next=self.head
            self.head=newnode
    def insertatend(self,value):
        newnode=Node(value)
        if self.head==None:
            self.head=newnode
        else:
            curr=self.head
            while curr.next!=None:
                curr=curr.next
            curr.next=newnode
    def searching(self,key):
        curr=self.head
        while curr!=None:
            if curr.data==key:
                print(key, "is found")
                break
            curr=curr.next
        else:
            print(key, "is not found")
    def count(self):
        count=0
        if self.head==None:
```

```
            print("no.of nodes=0")
        else:
            curr=self.head
            while curr!=None:
                count+=1
                curr=curr.next
        print(count)
    def insertinganywhere(self,value,key):
        newnode=Node(key)
        curr=self.head
        while curr!=None:
            if curr.data==value:
                newnode.next=curr.next
                curr.next=newnode
                break
            curr=curr.next
        else:
            print("element not found")
    def insertatmid(self,value):
        newnode=Node(value)
        count=0
        if self.head.next==None:
            self.head=newnode
        elif self.head.next==None:
            self.head.next=newnode
        else:
            curr=self.head
            while curr!=None:
                count+=1
                curr=curr.next
            curr=self.head
            for i in range(count//2):
                curr=curr.next
            newnode.next=curr.next
            curr.next=newnode
    def middle(self,value):
```

```
        newnode=Node(value)
        count=0
        if self.head.next==None:
            self.head=newnode
        elif self.head.next==None:
            self.head.next=newnode
        else:
            fast=self.head
            slow=self.head
            while fast.next!=None and fast.next.next!=None:
                fast=fast.next.next
                slow=slow.next
            newnode.next=slow.next
            slow.next=newnode
    def printlist(self):
        curr=self.head
        while(curr!=None):
            print(curr.data,"->",end=" ")
            curr=curr.next
        print("null")
l=linkedlist()
l.insertatbeg(1)
l.insertatbeg(2)
l.insertatbeg(3)
l.insertatbeg(4)
l.insertatend(5)
l.insertatend(6)
l.searching(7)
l.insertinganywhere(3,8)
l.insertatmid(9)
l.middle(7)
l.count()
l.printlist()
```