1. arr=list(map(int,input().split()))

n=int(input())

count=0

for i in range(0,n):

if(arr[i]%3==0):

count=count+(arr[i]/3)

elif(arr[i]%3!=0):

count=count+(arr[i]//3)+1

print(int(count))

1. def age\_calculation(n):

return n\*7

n=int(input())

print(age\_calculation(n))

1. # Enter your code here

str= "b h ar th ku m a r"

print(str.count(' '))

count = 0

for i in str:

if(i==' '):

count=count+1

print(count)

1. str= “kadapa"

print(str.count(' '))

count = 0

for i in str:

if(i==' '):

count=count+1

print(count)

print(" malli"\*5)

1. s=input()

count\_hyphen = s.count('-')

new\_str='-'\*count\_hyphen + s.replace('-',' ')

print(new\_str)

print("count of extra spaces")

count\_spaces=new\_str.count(' ')

new\_str=new\_str.replace(' ',' ')

print(new\_str)

1. class Node:

def \_\_init\_\_(self,data):

self.left=None

self.right=None

self.data=data

class bst:

def \_\_init\_\_(self):

self.root=None

def insert(self,data,root):

if root is None:

return Node(data)

if(data<root.data):

root.left=self.insert(data,root.left)

elif(data>root.data):

root.right=self.insert(data,root.right)

return root

def inorder\_traversal(self,root):

if root:

self.inorder\_traversal(root.left)

print(root.data,end=" ")

self.inorder\_traversal(root.right)

def preorder\_traversal(self,root):

if root:

print(root.data,end=" ")

self.preorder\_traversal(root.left)

self.preorder\_traversal(root.right)

def postorder\_traversal(self,root):

if root:

self.postorder\_traversal(root.left)

self.postorder\_traversal(root.right)

print(root.data,end=" ")

bst\_tree=bst()

root=None

root=bst\_tree.insert(20,root)

root=bst\_tree.insert(10,root)

root=bst\_tree.insert(900,root)

bst\_tree.inorder\_traversal(root)

1. class Node:

def \_\_init\_\_(self,data):

self.data=data

self.next=None

class LinkedList:

def \_\_init\_\_(self):

self.head=None #starting point of LL

def insert(self,data):

new\_node=Node(data)

if self.head is None:

self.head=new\_node

else:

temp=self.head

while temp.next:

temp=temp.next

temp.next=new\_node

def display(self):

temp=self.head

while temp:

print(temp.data,end="->")

temp=temp.next

print("None")

ll=LinkedList()

ll.insert(10)

ll.insert(20)

ll.insert(30)

ll.insert(40)

ll.display()