**Variables and Operators – LC**

The Cargo Coordinates

lct=input()

lc=input()

cn=input()

row=lct/lc

totalRowCount=row\*row

z=cn/totalRowCount

xtemp=cn%totalRowCount

ytemp=xtemp%row

y=((xtemp-ytemp)/row)

if(xtemp == 0):

x=row-1

else :

x=xtemp%row-1

finalOut="("+str(x)+","+str(y)+","+str(z)+")"

print(finalOut)

Volume of the container

l=input()

w=input()

print(l\*w)

Automatic volume detection

l=input()

w=input()

h=input()

print(l\*w\*h)

**Decision Making – LC**

Damaged Cargo

lc=input()

lc1=input()

c=input()

d=lc/lc1

nc=d\*d

import decimal

z=c/nc

if(c-(nc\*z))%d==0:

x=(c-(nc\*z))/d

x=x+2

else:

x=(c-(nc\*z))%d

x=x-1

if(c-(nc\*z))%d==0:

y=(c-(nc\*z))/d

y=y-1

else:

y=(c-(nc\*z))/d

print ("("+str(x)+","+str(y)+","+str(z)+")")

if z in (0,4):

print "Yes"

elif x==0:

if y in (0,1,2,3,4):

print "Yes"

elif y ==0 or y==4:

if x in (0,1,2,3,4):

print "Yes"

elif y ==1 or y== 2 or y==3:

if x in (0,4):

print "Yes"

else:

print "No"

else:

print "No"

Locate the Container

a1=input()

a2=input()

a3=input()

a4=input()

total=input()

sum= a1+a2+a3+a4

if sum<=total:

print "Not Possible"

elif total==a1 and total < (a1+a2):

print 2

elif total >= (a1+a2) and total < (a1+a2+a3):

print 3

elif total>=(a1+a2+a3) and total < (a1+a2+a3+a4):

print 4

elif total>=(a1+a2+a3+a4):

print 5

Identify the Container type

val=input()

if val>=0 and val <=1000:

print "Saver"

elif val>1000 and val <= 10000:

print "Economy"

else :

print "Flexi"

**Creating and Executing Loops / LC**

The Damaged Cargo

r=input()

if (r-3)%2==0:

p=(r-3)/2

if p%2==0:

print p

else:

print "Safe"

elif (r-4)%2==0:

p=(r-4)/2+1

if p%2!=0:

print p

else:

print "Safe"

else:

print "Safe"

The Weight detection sensor

tw=input()

j=0

while True:

r=input()

if r<tw:

tw=tw-r

j+=1

elif r==tw:

j+=1

break

else:

break

print j

The Cargo arrangement

count=input()

c=1

d=count-1

for x in range(count):

print('#' \* d + '\*' \* c + '+' \* d)

c=c+2

d=d-1

**Working with Lists / LC**

Hashing the Cargos

data=raw\_input()

myList=data.split(",")

inp=input()

result=list()

for x in range(len(myList)):

if int(myList[x])%inp == 0:

result.append(inp)

else:

result.append(int(myList[x])%inp)

print '[%s]' % ",".join(str(y) for y in result)

[Sequencing the Cargos](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4444)

data=raw\_input()

myList=data.split(",")

pos=input()

sortData= sorted(myList,key=int,reverse=True)

value=0

for x in range(pos):

x=int(sortData[x])

value=value+x

print value

[The Lounge Numbers](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4443)

data=raw\_input()

myList=data.split(",")

srch=raw\_input()

indices=[i for i,x in enumerate(myList) if x==srch]

if len(indices)>0:

for y in indices:

print (y+1)

else:

print 0

**Working with Strings / LC**

Validation of Employee Email ID

import re

email=raw\_input()

index=list()

tld=email[email.rfind('.')+1:len(email)]

dname=email[email.rfind('@')+1:email.rfind('.')]

un=email[0:email.rfind('@')]

etld=['com' , 'in' , 'edu']

if tld not in etld:

index.append(1)

if len(dname)<4:

index.append(2)

if un[0] in ('.','-','\_') or un[-1] in ('.','\_','-') or un.isupper() or (not re.match("^[a-z0-9.\_-]\*$",un)):

index.append(3)

if len(index)>0:

print "Invalid"

print '\n'.join(str(x) for x in index)

else:

print "Valid"

Employee Username

f=raw\_input()

m=raw\_input()

l=raw\_input()

un=f[0:3]+m[0:1]+l[0:2]

print un

**Working with Tuples / LC**

List of Tuples

import datetime

n=input()

mytuple=[]

for x in range(n):

ui1=tuple(map(str,raw\_input().split(',')))

mytuple.append(ui1)

date1=raw\_input()

print mytuple

for itm in mytuple:

fildat=datetime.datetime.strptime(date1,'%d-%m-%Y')

itmdat=datetime.datetime.strptime(itm[1],'%d-%m-%Y')

if itmdat>fildat:

print itm[0]

Know Your Customer Wing

data=raw\_input()

tup1=()

tup1=data.split(",")

for x in range(len(tup1)/7):

print "Door-no: "+tup1[0]

print "Street: "+tup1[1]

print "Area: "+tup1[2]

print "City: "+tup1[3]

print "State: "+tup1[4]

print "Zipcode: "+tup1[5]

print "Country: "+tup1[6]

**Declaring Functions and Passing Values / LC**

[Loading Expressions](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4468)

import re

st=raw\_input()

cargo=["{","}"]

cont=["[","]"]

box=["(",")"]

if st[0]<>'{' or st[len(st)-1] <>'}':

print "Invalid"

exit()

if st.count('{',0,len(st)) <> st.count('}',0,len(st)):

print "Invalid"

exit()

string1=st.split("}+")

for x in range(0,len(string1)):

if re.search("{",string1[x]):

if string1[x].count("{",0,len(string1[x]))>1 or string1[x].count("}",0,len(string1[x]))>1:

print "Invalid"

exit()

elif re.search("[",string1[x]):

string2=string1[x].split("]+")

for y in range(0,len(string2)):

if string2[y].count("[",0,len(string2[y]))>1 or string2[y].count("]",0,len(string2[y]))>1:

print "Invalid"

exit()

elif re.search("(",string1[x]):

string3=string1[x].split(")+")

for y in range(0,len(string3)):

if string3[y].count("(",0,len(string3[y]))>1 or string3[y].count(")",0,len(string3[y]))>1:

print "Invalid"

exit()

print "Valid"

Palindromic Cargo

import re

def checkPalindrome(param):

val=re.sub('[^a-zA-Z0-9]','',param)

reverseval=val.lower()[::-1]

if val.lower()==reverseval:

print "Yes"

else:

print "No"

inp=raw\_input()

checkPalindrome(inp)

**Working with Dictionary / LC**

Tracking the Shipment Dates

import datetime

day=raw\_input()

noof=input()

shipdate=raw\_input()

ship=datetime.datetime.strptime(shipdate,"%d-%m-%Y")

nxtdt=ship+datetime.timedelta(days=1)

nxtdt=nxtdt.strftime("%d-%m-%Y")

flag="Cont"

i=1

while i<noof:

if day.lower() =="monday":

nxtdt=ship+datetime.timedelta(days=1)

day="tuesday"

nxtdt=nxtdt.strftime("%d-%m-%Y")

print nxtdt

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="tuesday":

nxtdt=ship+datetime.timedelta(days=1)

day="wednesday"

nxtdt=nxtdt.strftime("%d-%m-%Y")

print nxtdt

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="wednesday":

nxtdt=ship+datetime.timedelta(days=1)

day="thursday"

nxtdt=nxtdt.strftime("%d-%m-%Y")

print nxtdt

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="thursday":

nxtdt=ship+datetime.timedelta(days=1)

day="friday"

nxtdt=nxtdt.strftime("%d-%m-%Y")

print nxtdt

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="friday":

day="saturday"

nxtdt=ship+datetime.timedelta(days=1)

noof=noof+1

nxtdt=nxtdt.strftime("%d-%m-%Y")

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="saturday":

day="sunday"

nxtdt=ship+datetime.timedelta(days=1)

noof=noof+1

nxtdt=nxtdt.strftime("%d-%m-%Y")

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

elif day.lower() =="sunday":

nxtdt=ship+datetime.timedelta(days=1)

day="monday"

nxtdt=nxtdt.strftime("%d-%m-%Y")

print nxtdt

shp=nxtdt

ship=datetime.datetime.strptime(shp,"%d-%m-%Y")

i+=1

Expected Date of Delivery

from datetime import datetime,timedelta,date,time

n=input()

val=raw\_input()

day,month,year=val.split('-')

val=day+'-'+month+'-'+'20'+year

date=datetime.strptime(val,"%d-%m-%Y")

newval= date+timedelta(days=n)

print datetime.strftime(newval,'%d-%m-%Y')

**File Handling / LC**

File Copy

f1=open('file\_in.txt','r')

f2=open('file\_out.txt','w')

myLine=f1.readlines()

f1.close()

for i in range (len(myLine)):

f2.write(myLine[i])

f2.close()

Heaviest Cargos

fo=open('readnLines.txt','r')

myLine=map(str.rstrip,fo.readlines())

fo.close()

print "Enter the n value"

a=input()

for i in range(len(myLine)-a,len(myLine)):

print myLine[i]

[CSV to XML](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4494)

import csv

csvFile='product.csv'

xmlFile='product.xml'

csvData=csv.reader(open(csvFile))

xmlData=open(xmlFile,'w')

xmlData.write('<productList>'+'\n')

rowNum=0

for row in csvData:

if rowNum==0:

tags=row

else:

xmlData.write('<product>'+'\n')

for i in range(len(tags)):

if tags[i]=='productname':

tags[i]='productName'

xmlData.write('<'+tags[i]+'>'+str(row[i])+'</'+tags[i]+'>'+'\n')

xmlData.write('</product>'+'\n')

rowNum+=1

xmlData.write('</productList>'+'\n')

xmlData.close()

[Total cost from XML](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4495)

import xml.etree.ElementTree as ET

tree=ET.parse('product.xml')

root=tree.getroot()

amount=0

for product in root.findall('product'):

amount=amount+int(product.find('cost').text)

print 'Total Cost : '+str(amount)

[XML to CSV](https://qa2qe.cognizant.e-box.co.in/problem/showDescription/4493)

import xml.etree.ElementTree as ET

tree =ET.parse('product.xml')

root=tree.getroot()

print 'id,productName,cost,weight'

for product in root.findall('product'):

print product.find('id').text+','+product.find('productName').text+','+product.find('cost').text+','+product.find('weight').text