In [ ]:
```
#Read no of array element
n=input().split()
n[0],n[1]=int(n[0]),int(n[1])

a = input().split()
sum = []

#cumulative sum
for i in range(0,n[0]):
    if i==0:
        sum.append(int(a[i]))
    else:
        sum.append(int(sum[i-1])+int(a[i]))
del a
#Read each query and calculate the average
#print(sum[n[0]-1])
for k in range(0,n[1]):
    inq = input().split()
    i=int(inq[0])
    j=int(inq[1])
    if i>1:
        print(sum[j-1]-sum[i-2])//(j-i+1))
    else:
        print(sum[j-1]//(j-i+1))
```

#special number

A special number is defined as a number which has at least

write a program to determine whether a number N is a special

inputformat first line

In [8]:
```
#Function to determine if a number in a list
def isSpecialNumber(n,p):
    if numberprimeFactors(n) >= p:
        return True
    return False

# Function to check if number is prime

def isprime(n):
    flag=1
    if n==2:
        return True
    for i in range(3,n//2+1):
        if n%i==0:
            flag=0
            return False
    if flag==1:
        return True
#Function to check if number of prime factors  for a given number

def numberprimeFactors(n):
    if isprime(n):
        return 1
    count = 0
    for i in range(2,n//2+1):
        if isprime(i) and n%i == 0:
            count+=1
    return count

#numberprimeFactors(30)
isSpecialNumber(8,2)
```

Out[8]: True

In [11]:
```
def solution2():
    p=int(input())
    t=int(input())
    for i in range(0,t):
        n=int(input())
        if isSpecialNumber(n,p):
            print("YES")
        else:
            print("NO")
solution2()
```

```
2
3
6
YES
7
NO
8
YES
```

In [ ]:
```
#special numbers in the list
def isprime(num):
    for i in range(2,num+1):
        if num%i==0
           return False
        return True
def specialnumber(n):
    for i in range(1,n):
        if n%i==0:
            if isprime(n)
```

Tuples

t1=()

li=[]

Differnce between Lists and Tuples

Lists are mutable -can be changed/modified

    - used to Access ,Modify, Add, Delete

Tuples are immutable -cannot be changed

- Used to access data only
- All slicing work

In [14]:
```
t1=(1,2,8,6,0)

t1[3]   # Accessing the fourth element

t1[8:]# Accessing all elements from middle to last

t1[len(t1)//2:] #Accessing all elements
```

Out[14]:
```
(8, 6, 0)
```

In [15]:
```
type(t1)
```

Out[15]:
```
tuple
```

Dictionaries

It Works on the concept of set

Unique Data

Keys,values

key is the unique identifier for a value

value is data that can be accessed with key

In [24]:
```
d1={"k1":"value1","k2":"value2"}

d1["k2"]          #Accessing the value with key "k2"
d1.keys()         #Accessing list of keys
d1.values()       #Accessing list of all values
d1.items()        #return list of tuples of keys and values
d1["k3"] = "value3"   # adding the numbers to dictionaries

d1["k3"] = "value4"   #Updating an elements
d1.pop("k3")    #Removing an element


d1
```

Out[24]:
```
{'k1': 'value1', 'k2': 'value2'}
```

contacts Applications : Examples for Dictionaries

- Add Contacts
- Search for contact
- List all contact
  - name1 : phone1
  - name2 : phone2
- Modify contact
- Remove contact
- import contact

In [26]:
```
contacts = {}

def addcontacts(name,phone):
    #verify that the contact does not already exist
    if name not in contacts:
        contacts[name] = phone
        print("contact %s added")
    else:
        print("contact %s already exists" %name)
    return
addcontacts("name1","9774462523")
```

```
contact %s added
```

In [30]:
```
def searchcontacts(name):
    if name in contacts:
        print(name,":",contacts[name])
    else:
        print("%s does not exists " %name)
    return
searchcontacts("name1")
```

```
name1 : 9774462523
```

In [ ]:
```
def Listcontact(list):
```

In [33]:
```
def importcontact(newcontacts):
    contacts.update(newcontacts)
    print(len(newcontacts.keys()), "contact added")
    return
newcontacts = {"name2" :975323454,"name3" :8634567898}

importcontact(newcontacts)
```

```
2 contact added
```

Pakages and Modules

Pakage -> collection of Modules (Python File.py)

Sub pakage ->

module -> A single python file containing function

pakage -> Subpakages -> Modules -> Functions

In [35]:
```
import math

math.floor(123.456)

math.pi
```

Out[35]:
```
3.141592653589793
```

In [44]:
```
from math import floor as fl

fl(123.456)
```

Out[44]:
```
123
```

In [53]:
```
import random

def GenerateNRandomNumbers(n,lb,ub):
    for i in range(0,n):
        print(random.randint(lb,ub),end=" ")

GenerateNRandomNumbers(10,0,100)
```

```
26 25 23 90 10 4 44 38 56 74
```

In [ ]:
```

```

In [1]:
```
from Packages import numerical

numerical.isprime(30)
```

Out[1]:
```
False
```

In [ ]:
```
i
```