All Cs Programs

June 7, 2020

[1]:
```
#1.1 A program to input a number and print its cube.
num=float(input("Enter a number :"))
cube=num**3
print("The cube of the number",num,"is",cube)
```

```
Enter a number :3
The cube of the number 3.0 is 27.0
```

[6]:
```
#1.2 A program to input a number and print its square root.
num=float(input("Enter a number :"))
sqrt=num**0.5
print("The square root of the number",num,"is",sqrt)
```

```
Enter a number :25
The square root of the number 25.0 is 5.0
```

[1]:
```
#1.3 A program that inputs an integer in range 0-999 and prints if the integer␣
 ↪entered is a 1/2/3 digit number.
num=int(input("Enter a number(0...999) :"))
if num<0:
    print("Invalid Entry")
elif num<10:
    print("Single digit number")
elif num<100:
    print("Double digit number")
elif num<=999:
    print("Triple digit number")
else:
    print("Invalid Entry")
```

```
Enter a number(0…999) :87
Double digit number
```

[3]:
```
#1.4 A program that inputs an integer in range 0-999 and prints if the integer␣
 ↪entered is a 1/2/3 digit number.
num=int(input("Enter a number(0...999) :"))
if num<0 or num>999:
    print("Invalid Entry")
```

```
else:
    if num<10:
        print("Single digit number")
    else:
        if num<100:
            print("Double digit number")
        else:
            if num<=999:
                print("Triple digit number")
```

```
Enter a number(0…999) :505
Triple digit number
```

[4]:
```
#1.5 A program to print cubes of numbers in the range 15 to 20.
for i in range(15,21):
    print("The cube of number",i,"is",i**3)
```

```
The cube of number 15 is 3375
The cube of number 16 is 4096
The cube of number 17 is 4913
The cube of number 18 is 5832
The cube of number 19 is 6859
The cube of number 20 is 8000
```

[7]:
```
#1.6 A program to print square root of every alternate number in the range 1 to␣
 ↪10.
for i in range(1,10,2):
    print("The square root of number",i,"is",i**0.5)
```

```
The square root of number 1 is 1.0
The square root of number 3 is 1.7320508075688772
The square root of number 5 is 2.23606797749979
The square root of number 7 is 2.6457513110645907
The square root of number 9 is 3.0
```

[9]:
```
#1.7 A program that multiplies two integers without using the * operator, using␣
 ↪repeted addition.
x=int(input("Enter the first number :"))
y=int(input("Enter the second number :"))
product=0
count=x
while count > 0:
    count-=1
    product+=y
print("The product of",x,"and",y,"is",product)
```

```
Enter the first number :5
Enter the second number :5
```

```
The product of 5 and 5 is 25
```

[7]:
```
#2.1 A program that reads a line and prints its statistics.
line=input("Enter a line :")
lowercount=uppercount=0
digitcount=alphacount=0
for a in line:
    if a.islower():
        lowercount+=1
    elif a.isupper():
        uppercount+=1
    elif a.isdigit():
        digitcount+=1
    if a.isalpha():
        alphacount+=1
print("Number of uppercase letters are",uppercount)
print("Number of lowercase letters are",lowercount)
print("Number of alphabets are",alphacount)
print("Number of digits",digitcount)
```

```
Enter a line : 0y8y 08HU 97T 0F88yfsy fsy gaqY8GG Y g8YUg so8yugXu
Number of uppercase letters are 11
Number of lowercase letters are 20
Number of alphabets are 31
Number of digits 12
```

[1]:
```
#2.2 A program to create a dictonary containing names of competition winner␣
↪students as key and number of their wins as values.
n=int(input("How many students?"))
CompWinners={}
for i in range (n):
    key=input("Name of the student:")
    value=int(input("Number of cometitions won.:"))
    CompWinners[key]=value
print("The diconary now is :")
print(CompWinners)
```

```
How many students?4
Name of the student:Manoj
Number of cometitions won.:4
Name of the student:Dhruva
Number of cometitions won.:6
Name of the student:Yash
Number of cometitions won.:2
Name of the student:Aniket
Number of cometitions won.:1
The diconary now is :
{'Manoj': 4, 'Dhruva': 6, 'Yash': 2, 'Aniket': 1}
```

[2]:
```
#2.3 A program that adds list2 and list3 to list1 as single element each. The␣
↪resultant list should me in the order off list3, elements of list1,list2.
list1=['a','b','c']
list2=['h','i','t']
list3=['0','1','3']
print("Orginally:")
print("List1=",list1)
print("List2=",list2)
print("List3=",list3)

list1.append(list2)
list1.insert(0,list3)
print("After adding two elements as individual elements,list now is :")
print(list1)
```

```
Orginally:
List1= ['a', 'b', 'c']
List2= ['h', 'i', 't']
List3= ['0', '1', '3']
After adding two elements as individual elements,list nnow is :
[['0', '1', '3'], 'a', 'b', 'c', ['h', 'i', 't']]
```

[4]:
```
#2.4 A program that adds individual elements of list2 and list3 to list1. The␣
↪resultant list should me in the order off list3, elements of list1,list2.
list1=['a','b','c']
list2=['h','i','t']
list3=['0','1','3']
print("Orginally:")
print("List1=",list1)
print("List2=",list2)
print("List3=",list3)

list3.extend(list1)
list3.extend(list2)
print("After adding elements of two lists individually, list now is :")
print(list3)
```

```
Orginally:
List1= ['a', 'b', 'c']
List2= ['h', 'i', 't']
List3= ['0', '1', '3']
After adding elements of two lists individually, list now is :
['0', '1', '3', 'a', 'b', 'c', 'h', 'i', 't']
```

[8]:
```
#2.5 A program that finds an element's index/position in tuple WITHOUT using␣
↪index().
tuple1=('a','p','p','l','e',)
```

```
char=input("Enter a single letter without quotes.:")
if char in tuple1:
    count=0
    for i in tuple1:
        if i != char:
            count+=1
        else:
            break
    print(char,"is at index",count,"in",tuple1)
else:
    print(char,"is NOT in",tuple1)
```

```
Enter a single letter without quotes.:l
l is at index 3 in ('a', 'p', 'p', 'l', 'e')
```

[10]:
```
#2.6 A program that checks for the presence of a value inside a dictonary and␣
 ↪prints its key.
info={'Riya':'CSc.','Mark':'Eco','Ishpreet':'Eng','Kamaal':'Env.Sc'}
inp=input("Enter values to be searched for :")
if inp in info.values():
    for i in info:
        if info[i]==inp:
            print("The key of the given value is",i)
            break
else:
    print("Given value does not exist in dictonary")
```

```
Enter values to be searched for :eng
Given value does not exist in dictonary
```

[7]:
```
#2.7 A program that checks for the presence of a value inside a dictonary and␣
 ↪prints its key, even if the case is different, i.e., match the two values␣
 ↪ignored their case.
info={'Riya':'CSc.','Mark':'Eco','Ishpreet':'Eng','Kamaal':'Env.Sc'}
inp=input("Enter values to be searched for :")
for i in info:
    if info[i].upper()==inp.upper():
            print("The key of the given value is",i)
            break
else:
    print("Given value does not exist in dictonary")
```

```
Enter values to be searched for :eng
The key of the given value is Ishpreet
```

[12]:
```
#2.8 A program to sort using bubble sort.
aList=eval(input("Enter Your List:"))
```

```
print("Orginal list is:",aList)
n=len(aList)
for i in range(n):
    for j in range (0,n-i-1):
        if aList[j]>aList[j+1]:
            aList[j],aList[j+1]=aList[j+1],aList[j]
print("List after sorting:",aList)
```

```
Enter Your List:[15,6,13,22,3,52,2]
Orginal list is: [15, 6, 13, 22, 3, 52, 2]
List after sorting: [2, 3, 6, 13, 15, 22, 52]
```

[14]:
```
#2.9 A program to sort sequence using insertion sort.
aList=eval(input("Enter Your List:"))
print("Orginal list is:",aList)
n=len(aList)
for i in range(1,n):
    key=aList[i]
    j=i-1
    while j >= 0 and key <aList[j]:
        aList[j+1]=aList[j]
        j-=1
    else:
        aList[j+1]=key
print("List after sorting:",aList)
```

```
Enter Your List:[15,6,13,22,3,52,2]
Orginal list is: [15, 6, 13, 22, 3, 52, 2]
List after sorting: [2, 3, 6, 13, 15, 22, 52]
```

[4]:
```
#3.1 A program to add two numbers through a function.
def calcSum(x,y):
    return (x+y)
#__main__
x=int(input("Enter the first number :"))
y=int(input("Enter the second number :"))
a=calcSum(x,y)
print("The sum of two numbers is",a)
```

```
Enter the first number :5
Enter the second number :5
The sum of two numbers is 10
```

[4]:
```
#3.2 Program to calculate compound interest.
def intrest(principal,time=2,rate=0.10):
    return (principal*time*rate)
#__main__
```

```
principal=float(input("Enter the principal amount:"))
print("Simple interst with default ROI and time values is : Rs.
 ↪",intrest(principal))
time=int(input("Enter time in years :"))
rate=float(input("Enter the rate of intrest(ROI) :"))
x=intrest(principal,time,rate/100)
print("The amount after",rate,"years is",x,"rupees.")
```

```
Enter the principal amount:50000
Simple interst with default ROI and time values is : Rs. 10000.0
Enter time in years :10
Enter the rate of intrest(ROI) :8
The amount after 8.0 years is 40000.0 rupees.
```

[3]:
```
#3.3 A program that recives two nummbers in a function and returns the results␣
 ↪of all arthemetic operations (+,-,*,/,%) on these numbers.
def arCalc(x,y):
    return x+y,x-y,x*y,x/y,x%y
#__main__
x=int(input("Enter the first number.:"))
=int(input("Enter the second number.:"))
add,sub,mult,div,mod = arCalc(x,y)
print("Sum of given numbers is",add)
print("Substraction of given numbers is",sub)
print("Product of given nimbers is",mult)
print("Division of given numbers is",div)
print("Modulo of given number is",mod)
```

```
Enter the first number.:13
Enter the second number.:7
Sum of given numbers is 20
Substraction of given numbers is 6
Product of given nimbers is 91
Division of given numbers is 1.8571428571428572
Modulo of given number is 6
```

[6]:
```
#4.1 A program that reads then converts it into binary, octal and hexadecimal␣
 ↪equivalent number using built-in functions of python.
num=int(input("Enter a number.:"))
print("The number entered is",num)
bnum=bin(num)
onum=oct(num)
hnum=hex(num)
print("Binary conversion is",bnum)
print("Octal conversion is",onum)
print("Hedxadecimal conversion is",hnum)
```

```
Enter a number.:17
The number entered is 17
Binary conversion is 0b10001
Octal conversion is 0o21
Hedxadecimal conversion is 0x11
```

[8]:
```
#4.2 A program that inputs a real number and convets it to nearest integer␣
↪using two  different built-in functions. It also displays the given number␣
↪rounded off  to three decimal palces.
num=float(input("Enter a real number.:"))
tnum=int(num)
rnum=round(num)
print("Number",num,"converted to integer in two ways as",tnum,"and",rnum)
rum=round(num,3)
print(num,"rounded off to three decimal places after decimal is",rum)
```

```
Enter a real number.:5.555628
Number 5.555628 converted to integer in two ways as 5 and 6
5.555628 rounded off to three decimal places after decimal is 5.556
```

[6]:
```
#4.3 A program that encrypts a string then decrypts it.
def encrypt(sttr,enkey):
    return enkey.join(sttr)
def decrypt(sttr,enkey):
    return sttr.split(enkey)
#__main__
sttr=input("Enter main string.:")
enkey=input("Enter encrypytion key.:")
enStr=encrypt(sttr,enkey)
deLst=decrypt(enStr,enkey)
deStr="".join(deLst)
print("The encrypted string is",enStr)
print("The decrypted string is",deStr)
```

```
Enter main string.:Manoj
Enter encrypytion key.:*&$#!
The encrypted string is M*&$#!a*&$#!n*&$#!o*&$#!j
The decrypted string is Manoj
```

[8]:
```
#4.4 A program to get request information from url and opens within the program.
import urllib
import webbrowser
weburl=urllib.request.urlopen('https://www.starwars.com/')
html=weburl.read()
data=weburl.getcode()
url=weburl.geturl()
hd=weburl.headers
inf=weburl.info()
```

```
print("The url is",url)
print("HTTP status code is data.:",data)
print("Headers returned:\n",hd)
print("The info() returned.:\n",inf)
print("Now opeaning the url",url)
webbrowser.open_new(url)
```

```
The url is https://www.starwars.com/
HTTP status code is data.: 200
Headers returned:
 Content-Type: text/html;charset=utf-8
Cache-Control: public, max-age=1800
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-UA-Compatible: IE=edge,chrome=1
ETag: W/"3866cce1768384bdd7c19e64f0b8d0ee"
X-Server: px-mh-ha-f
Vary: Accept-Encoding
Date: Sat, 25 Apr 2020 08:38:11 GMT
Transfer-Encoding: chunked
Connection: close
Connection: Transfer-Encoding


The info() returned.:
 Content-Type: text/html;charset=utf-8
Cache-Control: public, max-age=1800
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-UA-Compatible: IE=edge,chrome=1
ETag: W/"3866cce1768384bdd7c19e64f0b8d0ee"
X-Server: px-mh-ha-f
Vary: Accept-Encoding
Date: Sat, 25 Apr 2020 08:38:11 GMT
Transfer-Encoding: chunked
Connection: close
Connection: Transfer-Encoding


Now opeaning the url https://www.starwars.com/
```

[8]: True

[3]:
```
#5.1 A program to display the size of a file in bytes.
myfile=open("D:\\mytext.txt",'r')
st=myfile.read()
size=len(st)
```

```
print("Size of the given file is",size,"bytes.")
myfile.close()
```

```
Size of the given file is 469 bytes.
```

[4]:
```
#5.2 A program to display the number of lines in a file.
myfile=open("D:\\mytext.txt",'r')
st=myfile.readlines()
size=len(st)
print("Number of lines in the given file is",size,".")
myfile.close()
```

```
Number of lines in the given file is 35 .
```

[3]:
```
#5.3 A program to to get roll numbers, names and marks of the students of a␣
 ↪class (from the user) and store these details in a file called "Marks.txt".
count=int(input("How many sudents are there in the class? :"))
fileout=open("D:\\Marks.txt","w")
for i in range(count):
    print("Enter details of students",(i+1),"below.")
    rollno=int(input("Enter the Roll Number of the student.:"))
    name=input("Enter the name of the student.:")
    marks=float(input("Enter the marks obtained by the student.:"))
    rec=str(rollno)+","+name+","+str(marks)+"\n"
    fileout.write(rec)
fileout.close()
```

```
How many sudents are there in the class? :3
Enter details of students 1 below.
Enter the Roll Number of the student.:001
Enter the name of the student.:Krunal
Enter the marks obtained by the student.:81.17
Enter details of students 2 below.
Enter the Roll Number of the student.:002
Enter the name of the student.:Manoj
Enter the marks obtained by the student.:97.82
Enter details of students 3 below.
Enter the Roll Number of the student.:003
Enter the name of the student.:Neda
Enter the marks obtained by the student.:84.73
```

[4]:
```
#5.4 A program to to add two more students' details to the file created "Marks.
 ↪txt".
fileout=open("D:\\Marks.txt","a")
for i in range(2):
    print("Enter details of students",(i+1),"below.")
    rollno=int(input("Enter the Roll Number of the student.:"))
```

```
    name=input("Enter the name of the student.:")
    marks=float(input("Enter the marks obtained by the student.:"))
    rec=str(rollno)+","+name+","+str(marks)+"\n"
    fileout.write(rec)
fileout.close()
```

```
Enter details of students 1 below.
Enter the Roll Number of the student.:004
Enter the name of the student.:Shabnam
Enter the marks obtained by the student.:74.87
Enter details of students 2 below.
Enter the Roll Number of the student.:005
Enter the name of the student.:Shruti
Enter the marks obtained by the student.:999.99
```

[5]:
```
#5.5 A program that displays the content in file "Marks.txt".
fileout=open("D:\\Marks.txt","r")
while str:
    str=fileout.readline()
    print(str)
fileout.close()
```

```
1,Krunal,81.17

2,Manoj,97.82

3,Neda,84.73

4,Shabnam,74.87

5,Shruti,999.99
```

[6]:
```
#6.1 A recursive program that computes the sum of numbers 1...n; get the value␣
 ↪from the user.
def compute(num):
    if num==1:
        return 1
    else:
        return (num+compute(num-1))
#__main__
num=int(input("Enter a number.:"))
print("The sum of the series from 1 . . .",num,"is",compute(num))
```

```
Enter a number.:5
The sum of the series from 1 . . . 5 is 15
```

[2]:
```
#6.2 A recursive program to print a string backwards.
def bp(sg,n):
    if n>0:
        print(sg[n],end='')
        bp(sg,n-1)
    elif n==0:
        print(sg[0])
#__main__
sg=input("Enter a string.:")
bp(sg,len(sg)-1)
```

```
Enter a string.:MANOJ
JONAM
```

[10]:
```
#6.3 A recursive program to print a string backwards.
def bp(sg,n):
    if n>0:
        k=len(sg)-n
        bp(sg,n-1)
        print(sg[k],end='')
    elif n==0:
        return
#__main__
sg=input("Enter a string.:")
bp(sg,len(sg))
```

```
Enter a string.:MANOJ
JONAM
```

[2]:
```
#6.4 A program to show the use of recursion in calculation of a^b.
def power(a,b):
    if b==0:
        return 1
    else:
        return a*power(a,b-1)
#__main__
print("Enter only positive numbers.")
num=int(input("Enter the base number.:"))
p=int(input("Raised to the power.:"))
print(num,"raised to the power",p,"is",power(num,p))
```

```
Enter only positive numbers.
Enter the base number.:5
Raised to the power.:4
5 raised to the power 4 is 625
```

[3]:
```
#6.5 A program to calculate a^b usinng iterative code.
def power(a,b):
    res=1
    if b==0:
        return 1
    else:
        for i in range (b):
            res*=a
    return res
#__main__
print("Enter only positive numbers.")
num=int(input("Enter the base number.:"))
p=int(input("Raised to the power.:"))
print(num,"raised to the power",p,"is",power(num,p))
```

```
Enter only positive numbers.
Enter the base number.:5
Raised to the power.:4
5 raised to the power 4 is 625
```

[8]:
```
#6.6 Program to calculate the factorial of an integer using recursion.
def factorial(n):
    if n < 2:
        return 1
    return n*factorial(n-1)
#__main__
n=int(input("Enter a number.:"))
print("Factorial of ",n,"is",factorial(n))
```

```
Enter a number.:5
Factorial of  5 is 120
```

[4]:
```
#6.7 Program to print Fibonacci series using recursion.
def fib(n):
    if n==1:
        return 0
    elif n==2:
        return 1
    else:
        return fib(n-1)+fib(n-2)
#__main__
n=int(input("Enter the last term required:"))
for i in range(1,n+1):
    print(fib(i),end=',')
print ("...")
```

```
Enter the last term required:9
0,1,1,2,3,5,8,13,21,…
```

[9]:
```
#6.8 Program for binary search.
def binarysearch(ar,key):
    low=0
    high=len(ar)-1
    while low<=high:
        mid=int((low+high)/2)
        if key==ar[mid]:
            return mid
        elif key<ar[mid]:
            high=mid-1
        else:
            low=mid+1
    else:
        return-999
#__main__
ar=eval(input("Enter a list of integers:" ))
item=int(input("Enter search item:"))
res=binarysearch(ar,item)
if res>=0:
    print(item,"found at index",res,".")
else:
    print("Sorry",item,"not found in array.")
```

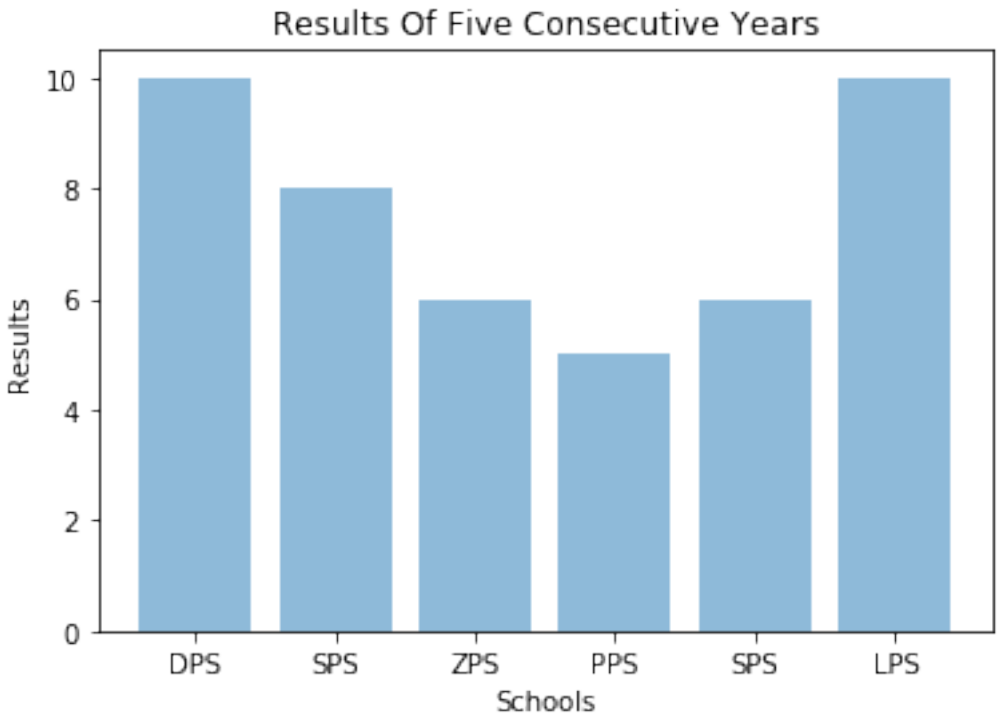
```
Enter a list of integers:[12,15,21,25,28,32,33,36,43,45]
Enter search item:32
32 found at index 5 .
```

[12]:
```
#6.9 A recursive function to implement binary search algorithm.
def binarysearch(ar,key,low,high):
    if low > high:
        return -999
    mid=int((low+high)/2)
    if key==ar[mid]:
        return mid
    elif key < ar[mid]:
        high=mid-1
        return binarysearch(ar,key,low,high)
    else:
        low=mid+1
        return binarysearch(ar,key,low,high)
#__main__
ar=eval(input("Enter a list of integers:" ))
item=int(input("Enter search item:"))
res=binarysearch(ar,item,0,len(ar)-1)
if res>=0:
    print(item,"found at index",res,".")
else:
```

```
    print("Sorry",item,"not found in array.")
```

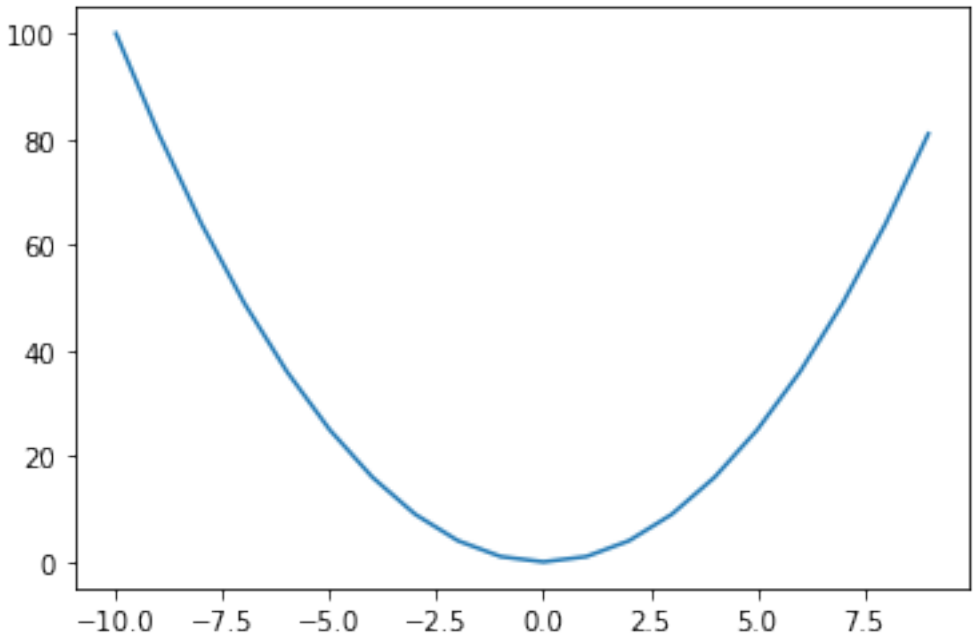
```
Enter a list of integers:[12,15,21,25,28,32,33,36,43,45]
Enter search item:32
32 found at index 5 .
```

[13]:
```
# 8.1 A program to plot a bar chart on to display the result of a school for
#    five consecutive year.
import matplotlib.pyplot as plt
import numpy as np
objects = ('DPS', 'SPS', 'ZPS', 'PPS', 'SPS', 'LPS')
y = np.arange(len(objects))
performance = [10,8,6,5,6,10]
plt.bar(y, performance, align='center', alpha=0.5)
plt.xticks(y, objects)
plt.ylabel('Results')
plt.xlabel('Schools')
plt.title('Results Of Five Consecutive Years')
plt.show()
```

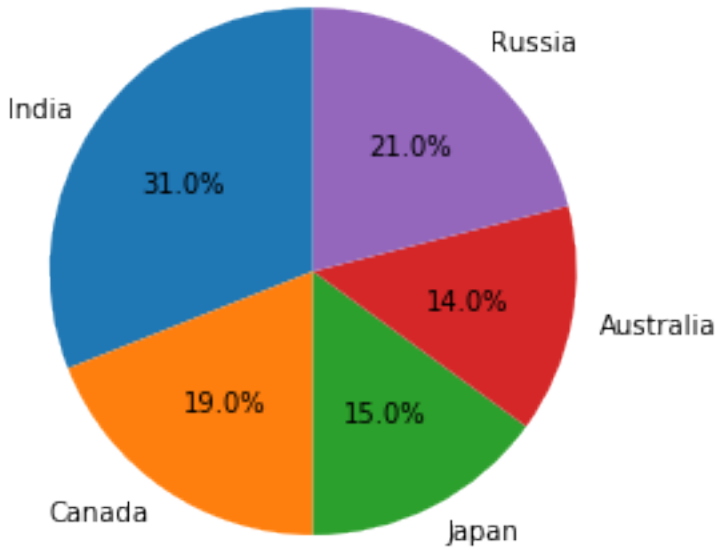


[3]:
```
#8.2 A program to plot a graph for the function y=x^2.
import matplotlib.pyplot as plt
import numpy as np
```

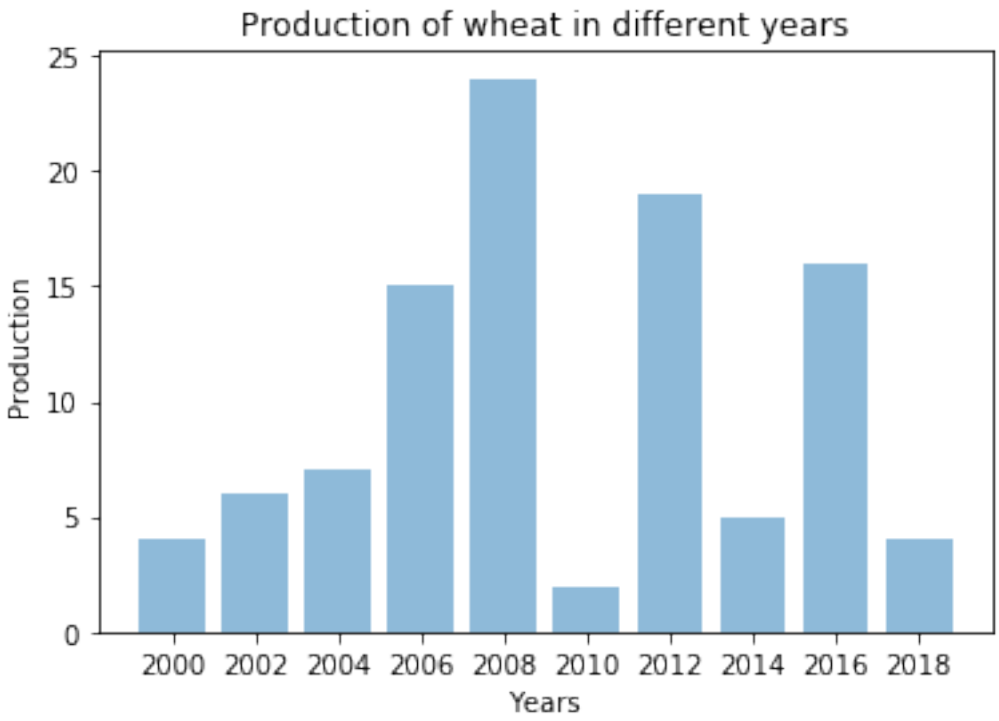
```
x = np.arange( -10 , 10 )
y = np.square( x )
plt.plot( x , y )
plt.show()
```



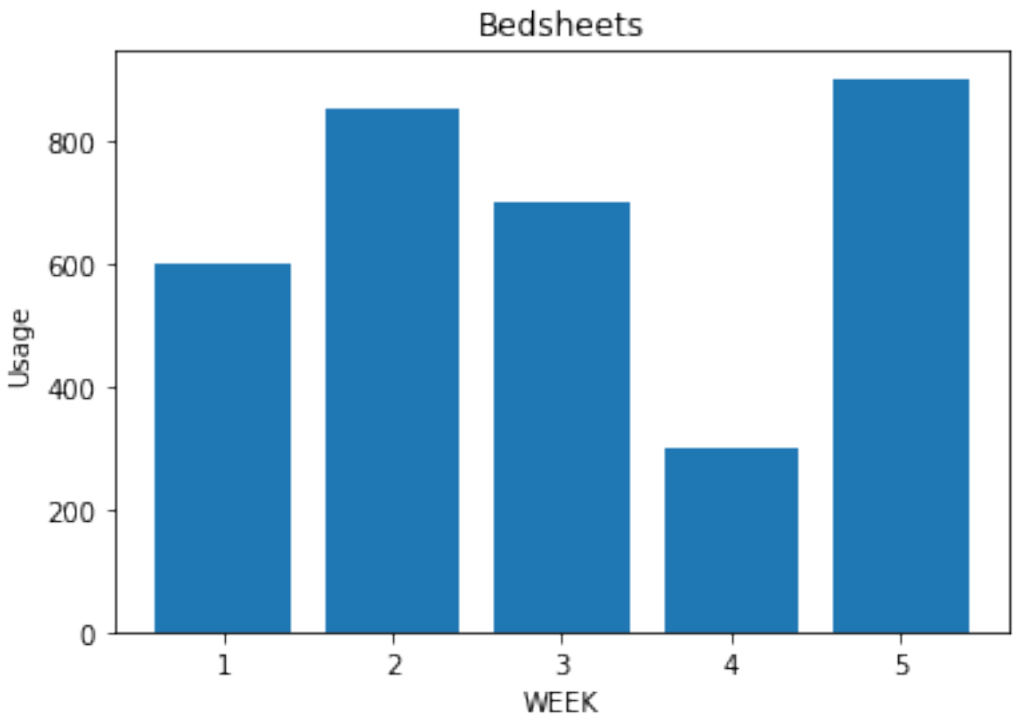
[15]:
```
#8.3 A program to plot a pie chart on consumption of water in daily life.
import matplotlib.pyplot as plt
labels = ['India', 'Canada', 'Japan', 'Australia', 'Russia']
sizes = [31, 19, 15, 14, 21] # Add upto 100%
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
plt.axis('equal')
plt.show()
```



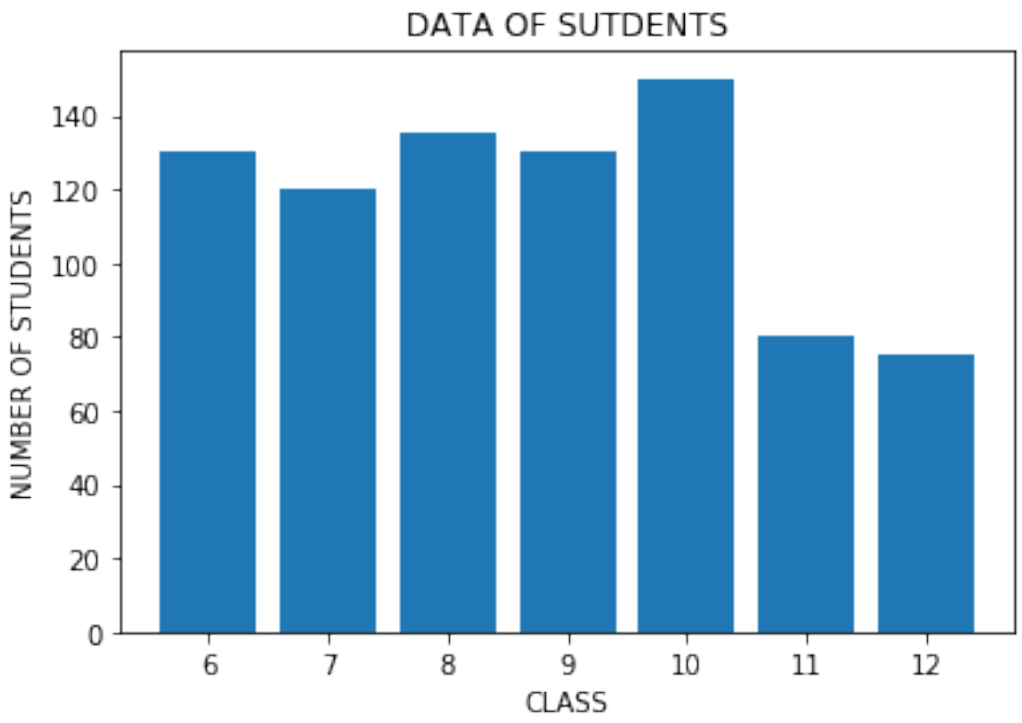
[16]:
```
#8.4 A graph based on the production of wheat in different years.
import matplotlib.pyplot as plt
import numpy as np
objects = (2000,2002,2004,2006,2008,2010,2012,2014,2016,2018)
y = np.arange(len(objects))
performance = [4,6,7,15,24,2,19,5,16,4]
plt.bar(y, performance, align='center', alpha=0.5)
plt.xticks(y, objects)
plt.ylabel('Production')
plt.xlabel('Years')
plt.title('Production of wheat in different years')
plt.show()
```



[17]:
```
#8.5 A bar graph representing bed-sheets manufactured by a factory.
import matplotlib.pyplot as plt
import numpy as np
x=[1,2,3,4,5]
y=[600,850,700,300,900]
plt.bar(x,y)
plt.xlabel("WEEK")
plt.ylabel("Usage")
plt.title("Bedsheets")
plt.show()
```

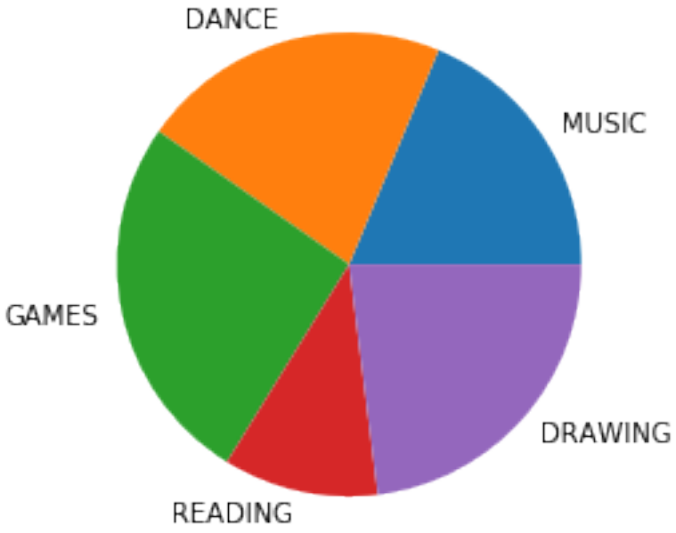


[18]:
```
#8.6 A program for graph of the give data.
import matplotlib.pyplot as plt
import numpy as np
a=[6,7,8,9,10,11,12]
b=[130,120,135,130,150,80,75]
plt.bar(a,b)
plt.xlabel("CLASS")
plt.ylabel("NUMBER OF STUDENTS")
plt.title("DATA OF SUTDENTS")
plt.show()
```

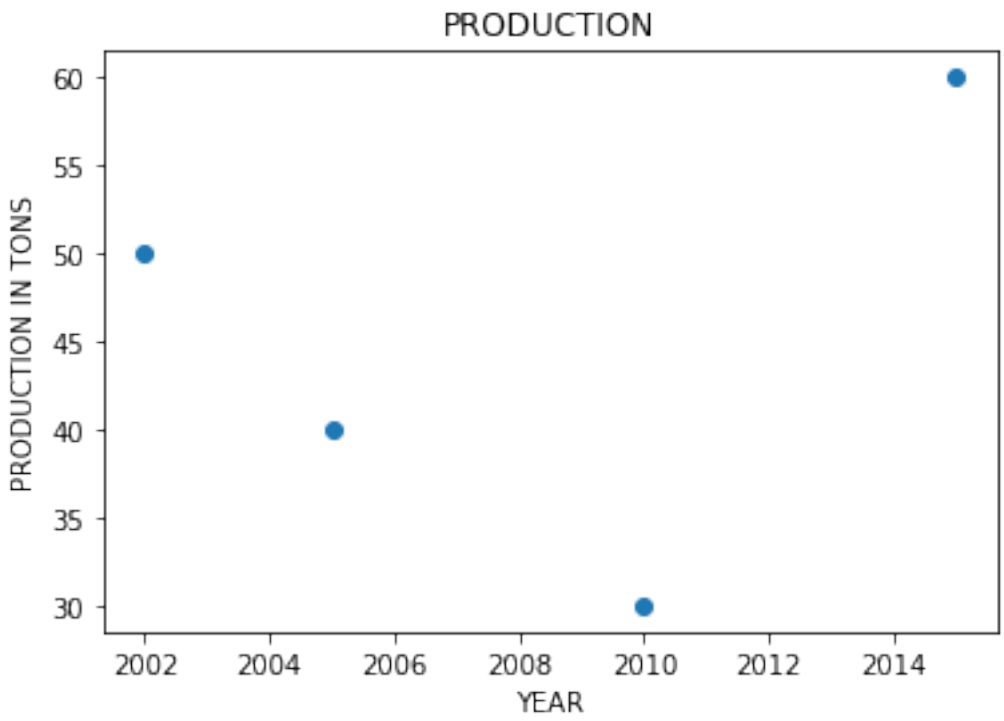


[19]:
```
#8.7 A program to pie graph of the given data.
import matplotlib.pyplot as plt
labels = ['MUSIC','DANCE', 'GAMES', 'READING','DRAWING']
number_of_students = [130,150,180,75,160]
plt.pie(number_of_students,labels=labels)
```

[19]:
```
([<matplotlib.patches.Wedge at 0x82af308>,
  <matplotlib.patches.Wedge at 0x8443dc8>,
  <matplotlib.patches.Wedge at 0x8448848>,
  <matplotlib.patches.Wedge at 0x844e248>,
  <matplotlib.patches.Wedge at 0x8453048>],
 [Text(0.9154789352338673, 0.6098346654160165, 'MUSIC'),
  Text(-0.30665142637607196, 1.0563923999639153, 'DANCE'),
  Text(-1.0773212340458904, -0.22221376796643377, 'GAMES'),
  Text(-0.23437408375699953, -1.0747412660092042, 'READING'),
  Text(0.824628636069477, -0.7280024811593669, 'DRAWING')])
```



[20]:
```
#8.8 A program to make a graph of the given data.
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
N = 500
x = [2002,2005,2010,2015]
y = [50,40,30,60]
colors = (0,0,0)
plt.title('PRODUCTION')
plt.xlabel('YEAR')
plt.ylabel('PRODUCTION IN TONS')
plt.scatter(x, y)
plt.show()
```



[2]:
```
#9.1 Linear searching in array(linear list).
def Lsearch(ar,items):
    i  = 0
    while i < len(ar) and ar[i] != item:
        i+=1
    if i < len(ar):
        return i
    else:
        return False
#__main__
n=int(input("Enter desired linear-list (max. 50)..."))
print("\nEnter elements for linear list\n")
ar=[0]*n
for i in range(n):
    ar[i]=int(input("Element "+str(i)+":"))
item=int(input("\nEnter element to be searched for..."))
index=Lsearch(ar,item)
if index:
    print("\nElement found at index :",index,",Position :",(index+1))
else:
    print("\nSorry!! Given element could not be found.\n")
```

```
Enter desired linear-list (max. 50)…7
```

```
Enter elements for linear list

Element 0:88
Element 1:77
Element 2:44
Element 3:33
Element 4:22
Element 5:11
Element 6:10

Enter element to be searched for…11

Element found at index : 5 ,Position : 6
```

[1]:
```python
#9.2 Binary Searching in an arry.
def binarysearch(ar,key):
    low=0
    high=len(ar)-1
    while low<=high:
        mid=int((low+high)/2)
        if key==ar[mid]:
            return mid
        elif key<ar[mid]:
            high=mid-1
        else:
            low=mid+1
    else:
        return-999
#__main__
ar=eval(input("Enter a list of integers:" ))
item=int(input("Enter search item:"))
res=binarysearch(ar,item)
if res>=0:
    print(item,"found at index",res,".")
else:
    print("Sorry",item,"not found in array.")
```

```
Enter a list of integers:[11,15,18,21,23,25,27,29]
Enter search item:29
29 found at index 7 .
```

[8]:
```python
#9.3 Insertion an element in a sorted array using traditional algorithm.
def FindPos (ar,item):
    size=len(ar)
    if item < ar[0]:
        return 0
    else:
```

```
        pos=-1
    for i in range (size-1):
        if(ar[i] <= item and item < ar[i+1]):
            pos=i+1
            break
    if(pos == -1 and 1 <= size-1):
        pos=size
    return pos
def Shift(ar,pos):
    ar.append(None)
    size=len(ar)
    i=size-1
    while i >= pos:
        ar[i]=ar[i-1]
        i-=1
#__main__
print("Enter the list in sorted order.")
ar=eval(input("Enter a list.:"))
item=int(input("Enter a new element to be inserted.:"))
position=FindPos(ar,item)
Shift(ar,position)
ar[position]=item
print("The list after insertion",item,"is",ar)
```

```
Enter the list in sorted order.
Enter a list.:[11,22,33,44,66,77,88,99]
Enter a new element to be inserted.:55
The list after insertion 55 is [11, 22, 33, 44, 55, 66, 77, 88, 99]
```

[9]:
```
#9.4 Insertion in sorted array using bisect module.
import bisect
print("Enter the list in sorted order.")
ar=eval(input("Enter a list.:"))
item=int(input("Enter a new element to be inserted.:"))
ind=bisect.bisect(ar,item)
bisect.insort(ar,item)
print(item,"inserted at index",ind)
print("The list after insertion is",ar)
```

```
Enter the list in sorted order.
Enter a list.:[11,22,33,44,66,77,88,99]
Enter a new element to be inserted.:55
55 inserted at index 4
The list after insertion is [11, 22, 33, 44, 55, 66, 77, 88, 99]
```

[10]:
```
#9.5 Deletion of an element from a sorted list.
def binarysearch(ar,key):
```

```
    low=0
    high=len(ar)-1
    while low<=high:
        mid=int((low+high)/2)
        if key==ar[mid]:
            return mid
        elif key<ar[mid]:
            high=mid-1
        else:
            low=mid+1
    else:
        return False
#__main__
print("Enter the list in sorted order.")
ar=eval(input("Enter a list.:"))
item=int(input("Enter a new element to be inserted.:"))
pos=binarysearch(ar,item)
if pos:
    del ar[pos]
    print("The list after deletion",item,"is",ar)
else:
    print("SORRY!! No such element in the list.")
```

```
Enter the list in sorted order.
Enter a list.:[11,22,33,44,55,66,77,88,99]
Enter a new element to be inserted.:55
The list after deletion 55 is [11, 22, 33, 44, 66, 77, 88, 99]
```

[3]:
```
#9.6 Traversing a linear list.
def traverse(ar):
    size=len(ar)
    for i in range(size):
        print(ar[i],end=' ')
#__main__
size=int(input("Enter the size of linear list to be input.:"))
ar=[None]*size
print("Enter elements for linear list.")
for i in range(size):
    ar[i]=int(input("Emlement "+str(i)+":"))
print("Traversing the list.")
traverse(ar)
```

```
Enter the size of linear list to be input.:6
Enter elements for linear list.
Emlement 0:12
Emlement 1:23
Emlement 2:34
Emlement 3:45
```

```
Emlement 4:56
Emlement 5:67
Traversing the list.
12 23 34 45 56 67
```

[2]:
```
## 9.7 Linear List Implimentation. A charity organization conducts camps in␣
 ↪various locations of a city. Each camp is located at a location at a␣
 ↪sepcific date. The organization maintains a montl1ly list of camp planned in␣
 ↪this format:
#    03 New Cly, 18 T Nagar, 25 K Pura.
#    Where first two digits of every entry signify the date on which the camp␣
 ↪is to be conducted. Write a program to inplement this along with the␣
 ↪following point:
#(i)   A linear list, planned, stores the camps that are to be conducted. A␣
 ↪camp's detils are added to planned list once NOC is obtained.
#(ii)  As soon as a camp is conducted, its details are moved to list conducted␣
 ↪and removed from planned list.
#(iii) Each camp servers some people.
#(iv)  The program should be able to provide options for adding to planned␣
 ↪list, getting conducted camp's details, searching for a camp, a report of␣
 ↪how many camps we are conducted so far and how many people were served and␣
 ↪display the linear lists planned and conducted.
def addloc(cmp) :
    dd=cmp[0:2]
    ln=len(planned)
    if ln==0:
        planned.append(cmp)
    else:
        last=planned[ln-1]
        if int(dd) >= int(last[0:2]):
            planned.append(cmp)
        else:
            for i in range(ln):
                cp = planned[i]
                if int(dd) <= int(cp[0:2]):
                    planned.insert(i,cmp)
                    break
def conductCamp(cmp):
    conducted.append(cmp)
    planned.remove(cmp)
def search ( cmp, lst) :                    # linear search technique
    ln = len(lst)
    for i in range(ln):
        if cmp in lst[i]:
            return lst[i]
    else:
        return False
```

```
def Report():
    lenp = len(planned)
    lenc = len (conducted)
    print("\tREPORT")
    print("-------------------------")
    print ( "Camps conducted so far:" ,lenc)
    print ("People served so far",ppl)
    print ( "Camps to be conducted:",lenp)
    print("-------------------------")
def display():
    print ("\nCamps Planned : ", end = ' ')
    for i in planned:
        print(i, end=', ')
    print("...!!")
    print ("\nCamps Conducted so far : ",end = ' ')
    for i in conducted:
        print(i, end=', ')
    print("...!!")
#__main__
planned= []
conducted = []
ppl = 0
ch= 0
while (ch != 6) :
    print("\t----")
    print("\tMENU")
    print("\t----")
    print( "1. Add Camp Location ")
    print ( "2. Camp Conducted" )
    print ( "3. Look for a Camp")
    print("4. Report")
    print ( "5. Display List " )
    print ( "6. Exit " )
    ch = int(input("Enter your choice (1 - 6 ): "))
    if ch==1:
        cm = input ("Enter Camp location : ")
        dd = input ("Enter date of the month (only dd) : ")
        cmp = dd+" " +cm
        addloc(cmp)
    elif ch == 2 :
        cm = input ( "Camp conducted at location?")
        p = int ( input ( "How many people are served at this camp?"))
        ppl = ppl + p
        result = search ( cm, planned)
        if result == False :
            print ("Sorry no such camp in the list")
        else :
```

```
            conductCamp(result)
    elif ch== 3:
        cm = input ("Enter camp location : ")
        r1 = search ( cm, planned)                          # result 1
        if r1 == False :
            r2 = search ( cm, conducted)                    # result 2
            if r2 == False:
                print( "Sorry no such camp in our list")
            else:
                dd = r2[0:2]
                print( cm, "was conducted on date", dd, "of this month")
        else :
            dd = r1[0:2]
            print( cm, "camp is to be conducted on date", dd, "of this month.")
    elif ch== 4 :
        Report()
    elif ch== 5:
        display()
    elif ch !=6:
        print ( "Wrong choice! Enter choice from 1 to 6 only")
else:
    print ( "THANK YOU")
```

```
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 1
Enter Camp location : K Pur
Enter date of the month (only dd) : 09
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 1
Enter Camp location : D Pur
Enter date of the month (only dd) : 27
```

```
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 1
Enter Camp location : B Nagar
Enter date of the month (only dd) : 03
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 1
Enter Camp location : Y Vihar
Enter date of the month (only dd) : 15
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 5

Camps Planned :  03 B Nagar, 09 K Pur, 15 Y Vihar, 27 D Pur, …!!

Camps Conducted so far :  …!!
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 5
```

```
Camps Planned :  03 B Nagar, 09 K Pur, 15 Y Vihar, 27 D Pur, …!!

Camps Conducted so far :  …!!
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 3
Enter camp location : D Pur
D Pur camp is to be conducted on date 27 of this month.
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 4
        REPORT
-------------------------
Camps conducted so far: 0
People served so far 0
Camps to be conducted: 4
-------------------------
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 2
Camp conducted at location?B Nagar
How many people are served at this camp?320
        ----
        MENU
        ----
1. Add Camp Location
```

```
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 2
Camp conducted at location?K Pur
How many people are served at this camp?412
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 4
        REPORT
-------------------------
Camps conducted so far: 2
People served so far 732
Camps to be conducted: 2
-------------------------
        ----
        MENU
        ----
1. Add Camp Location
2. Camp Conducted
3. Look for a Camp
4. Report
5. Display List
6. Exit
Enter your choice (1 - 6 ): 6
THANK YOU
```

[1]:
```python
# 10.1 Proagram to perform the operation on stack in python.
def cls():
    print("\n"*5)
def empty (stk):
    if stk==[]:
        return True
    else:
        return False
def push (stk,item):
    stk.append(item)
    top=len(stk)-1
```

```
def pop(stk):
    if empty(stk):
        return "Underflow"
    else:
        item=stk.pop()
        if len(stk)==0:
            top=None
        else:
            top=len(stk)-1
        return item
def peek(stk):
    if empty(stk):
        return "Underflow"
    else:
        top=len(stk)-1
        return stk[top]
def display(stk):
    if empty(stk):
        print("Empty stack")
    else :
        top=len(stk)-1
        print(stk[top],"<-top")
        for i in range(top-1,-1,-1):
            print(stk[i])
#__main__
Stack=[]
top = None
while True:
    cls()
    print("STACK OPERATION")
    print("1.PUSH")
    print("2.POP")
    print("3.PEEK")
    print("4.DISPLAY")
    print("5.EXIT")
    ch=int(input("ENTER YOUR CHOICE FROM 1 TO 5: "))
    if ch == 1:
        item=int(input("Enter item:"))
        push(Stack,item)
    elif ch==2:
        item= pop(Stack)
        if item=="Underflow":
            print("Underflow! The stack is empty")
        else:
            print("Popped item is ",item)
    elif ch==3:
        item=peek(Stack)
```

```
        if item=="Underflow":
            print("Underflow! Stack is empty")
        else:
            print ("Top most item is",item)
    elif ch==4:
        display(Stack)
    elif ch==5:
        break
    else:
        print("Invalid choice!")
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 1
Enter item:6
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 1
Enter item:8
```

```
STACK OPERATION
1.PUSH
```

```
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 1
Enter item:2
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 1
Enter item:4
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 4
4 <-top
2
8
6
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
```

```
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 3
Top most item is 4
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 4
4 <-top
2
8
6
```

```
STACK OPERATION
1.PUSH
2.POP
3.PEEK
4.DISPLAY
5.EXIT
ENTER YOUR CHOICE FROM 1 TO 5: 5
```

[1]:
```python
#10.2 Program to implement Queue operation.
def cls():
    print("\n"*5)
def empty(qu):
    if qu==[]:
        return True
    else:
        return False
def enqueue(qu,item):
    qu.append(item)
    if len(qu)==1:
        front=rear=0
    else:
```

```
        rear=len(qu)-1
def dequeue(qu):
    if empty(qu):
        return"Underflow"
    else:
        item=qu.pop(0)
    if len(qu)==0:
        front=rear=None
    return item
def peek(qu):
    if empty(qu):
        return"Underflow"
    else:
        front=0
    return qu[front]
def display(qu):
    if empty(qu):
        print("Queue Empty!")
    elif len(qu)==1:
        print(qu[0],"<== front, rear")
    else:
        front=0
        rear=len(qu)-1
        print(qu[front],"<-front")
        for a in range(1,rear):
            print(qu[a])
        print(qu[rear],"<-rear")
#__main__
queue=[]
front=None
while True:
    cls()
    print("QUEUE OPREATIONS")
    print("1.Enqueue")
    print("2.Dequeue")
    print("3.Peek")
    print("4.Display queue")
    print("5.Exit")
    ch=int(input("Enter your choice from 1 to 5: "))
    if ch == 1:
        item=int(input("Enter item:"))
        enqueue(queue,item)
        input("Press enter to continue...")
    elif ch == 2:
        item=dequeue(queue)
        if item=="Underflow":
            print("Underflow! The queue is empty.")
```

```
        else:
            print("Dequeue-ed item is",item)
        input("Press enter to continue...")
    elif ch==3:
        item=peek(queue)
        if item=="Underflow":
            print("Underflow! queue is empty")
        else:
            print ("Foremost most item is",item)
        input("Press enter to continue...")
    elif ch==4:
        display(queue)
        input("Press enter to continue...")
    elif ch==5:
        break
    else:
        print("Invalid choice!")
        input("Press enter to continue...")
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 1
Enter item:5
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 4
```

```
5 <== front, rear
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 1
Enter item:7
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 3
Foremost most item is 5
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 1
Enter item:9
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 4
5 <-front
7
9 <-rear
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 2
Dequeue-ed item is 5
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 2
Dequeue-ed item is 7
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 2
Dequeue-ed item is 9
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 2
Underflow! The queue is empty.
Press enter to continue…
```

```
QUEUE OPREATIONS
1.Enqueue
2.Dequeue
3.Peek
4.Display queue
5.Exit
Enter your choice from 1 to 5: 5
```