NAME: JAY KUSHWAHA

CLASS: 12th A

ROLL No: 6

CS PROGRAM FILE

FUNCTIONS

Check whether a no is prime or not.

```
def isprime(n,d=2):
    if n==d:
        return "prime"
    if n%d==0:
        return "not prime"
    elif d<n:
        return isprime(n,d+1)
print(isprime(7))
```

#Output:

```
Prime
```

Write a program using rec.fun to print fibonacci series upto nth term.

```
def fib(n):
    if n==1:
        return 0
    elif n==2:
        return 1
    else:
        return fib(n-1)+fib(n-2)
n=int(input("enter a no:"))
for w in range(1,n+1):
    print(fib(w),end=',')
print('...')
```

#Output:

```
enter a no:8
0,1,1,2,3,5,8,13,...
```

TEXT FILE

Read a txt file and display the no of words starting with alphabet 'a' or 'A'.

```
f=open("rt.txt","r")
f.seek(0)
data=f.read()
lst=data.split()
s=0
for w in lst:
    if w[0]=='a' or w[0]=='A':
        s+=1
print("Number of words:- ",s)
f.close()
```

OUTPUT:

```
Number of words:- 6
```

Def a fun reading and writing few lines from the user.Display the length of all the lines containing word 'nope'.

```
def fun(fn):
    f=open(fn,"r+")
    print("empty line to end input :")
    while True:
        s=input()
        if len(s)==0:
            break
        else:
            f.write(s+'\n')
    f.seek(0)
    lst=f.readlines()
    for w in lst:
        if 'nope' in w:
            print(len(w))
    f.close()
```

OUTPUT:

```
5
6
```

Read a txt file and replace each alpahabet with its next alphabet and 'z' with 'a'.

```
f=open("rt.txt","r+")
data=f.read()
print(data)
f.seek(0)
while True:
    n=f.tell()
    ch=f.read(1)
    if len(ch)==0:
        break
    if ch=='z':
        f.seek(n)
        f.write('a')
    elif ch.isalpha():
        f.seek(n)
        f.write(chr(ord(ch)+1))
f.seek(0)
r=f.read()
print(r)
f.close()
```

'''LIST'''

Write a program to print a list in ##separate lines along with elements both ##indexes(+ve and -ve).

```
lst=['a','a','r','t','i']
l=len(lst)
for a in range(l):
    print("At
indexes",a,"and",(a-l),"element:",lst[a])
```

#Output:

```
At indexes 0 and -5 element: a
At indexes 1 and -4 element: a
At indexes 2 and -3 element: r
At indexes 3 and -2 element: t
At indexes 4 and -1 element: i
```

Write a program to calculate the mean of list of nos.

```
lst=[1,2,3,4,5,6]
s,m=0,0
for w in lst:
    s+=w
    m=s/len(lst)
print("mean is:",m)
```

#Output:

```
mean is: 3.5
```

STACK

```
def push(s,x):
    global top
    s.append(x)
    top=len(s)-1
def pop(s):
    global top

    if len(s)==0: #if top==None:
        print(Underflow)
    else:
        x=s.pop( )
        print(poped;,x)
    if len(s)==0:
        top=None
    else:
        top=len(s)-1
def display(s):
    global top
    if len(s)==0:
        print(stack is empty)
    else:
        print(Stack elements....)
        ## for a in range (len(s)-1,-1,-1):
        ## print(s[a])
    for a in range (top,-1,-1):
        print(s[a])

stack=[]
top=None
while True:
    print(\nStack operations)
    print(1.Push&quot;)
    print(2.Pop&quot;)
    print(3.Display&quot;)
    print(4.Exit&quot;)
    print(top is,top)
```

```
ch=int(input(Enter choice : ))
if ch==1:
    item=int(input(Enter data :))
    push(stack,item)
elif ch==2:
    pop(stack)
elif ch==3:
    display(stack)
else:
    break
```

Output:

```
Stack operations
1.Push
2.Pop
3.Display
4.Exit
top is None
Enter choice : 1
Enter data : 111
Stack operations
1.Push
2.Pop
3.Display
4.Exit
top is 0
Enter choice : 1
Enter data : 222
Stack operations
1.Push
2.Pop
3.Display
4.Exit
top is 1
Enter choice : 1
Enter data : 333
Stack operations
1.Push
```

```
2.Pop
3.Display
4.Exit
top is 2
Enter choice : 3
Stack elements....
333
222
111
```

QUEUE

```
front=rear=None
def insert(q,x):
    global front,rear
    q.append(x)
    if len(q)==1:
        front=rear=0
    else:
        rear+=1
def delete(q):
    global front,rear
    if len(q)==0:
        return
    else:
    item=q.pop(0) # 0 is must ie FIFO
    if len(q)==0:
        front=rear=None
    else:
        rear-=1 #rear=rear-1
    return item
def dis(q):
    if len(q)==0:
        print(Nothing to display;)
    else:
        print(Displaying queue elements ****)
```

```
    for x in q:
        print(x)

def fr():
    global front,rear
    print("Front is",front,"Rear is",rear)
q=[]
while True:
    fr()
    print(\n\n\nQUEUE \n1.Insert\n2.Delete\n3.Dispaly)
    print(4.Front Rear Index\n5.Exit)
ch=int(input(Enter choice :))
if ch==1:
        insert(q,input(Enter data to insert))
elif ch==2:
    x=delete(q)
    if x==UNDERFLOW:
print(UNDERFLOW)
else:
print(x,deleted)
elif ch==3:
    dis(q)
elif ch==4:
    fr()
else:
    break
```

Output

```
QUEUE
1.Insert
2.Delete
3.Dispaly
4.Front Rear Index
5.Exit
Enter choice : 1
Enter data to insert111
Front is 0 Rear is 0

QUEUE
1.Insert
2.Delete
3.Dispaly
4.Front Rear Index
5.Exit
Enter choice : 1
Enter data to insert222
```

SORTING

Write a program that arranges a list using Bubble sort.

```
lst=[2,5,6,78,8,1,0]
n=len(lst)-1
for i in range(n):
    for j in range(0,n-i-1):
        if lst[j]>lst[j+1]:
            lst[j],lst[j+1]=lst[j+1],lst[j]
print("sorted list is",lst)
```

#Output:

sorted list is [1, 2, 5, 6, 8, 78, 0]

Write a program that arranges a list using Bubble sort.

```
lst=[5,9,2,1,4,3,60,0]
l=len(lst)
for i in range(1,l):
    key=lst[i]
    j=i-1
    while j>=0 and key<lst[j]:
        lst[j+1]=lst[j]
        j-=1
    lst[j+1]=key
print("modified list is:",lst)
```

#Output:

```
modified list is: [0, 1, 2, 3, 4, 5, 9, 60]
```

'''**TUPLES**'''

Write a program to print the names of the cars in the index range 1 to 4.

```
cars=('AustinMartin','Ferrari','Porshe','Audi','BMW','
Mercedes')
no=('Zero','One','Two','Three','Four','Five')
for w in range(1,5):
    print(no[w],cars[w])
```

#Output:

```
One Ferrari
Two Porshe
Three Audi
Four BMW
```

Replace four elements of list with four elements of tuple.

```
t=(1,2,3,4)
l=['We','Are','Coming','Soon','Takecare','till','then'
]
print("Before...",  l)
l[0],l[1],l[2],l[3]=t
print("After... ",  l)
```

#Output:

```
Before... ['We', 'Are', 'Coming', 'Soon', 'Takecare',
'till', 'then']
After...  [1, 2, 3, 4, 'Takecare', 'till', 'then']
```

'''STRINGS'''

Program that reads a line and replaces each uppercase alphabet with its previous one and lowercase into into its next.

If its a digit change it into $

```
l=input("enter a line:")
n=''
for w in l:
    if w.isupper():
        n+=chr(ord(w)-1)
    elif w.islower():
        n+=chr(ord(w)+1)
    elif w.isdigit():
        n+='$'
    else:
        n+=w
print("modified line is:",n)
```

#Output:

```
enter a line:What Are yoU doINg Here 8547.
modified line is: Vibu @sf zpT epHMh Gfsf $$$$.
```

Define a fun having a list of strings as its argument and the fun is displaying the str(s) having highest freq of alphabet 'a' and returning the sum of len of all the strings

```
def fun(lst):
    s=0
    for st in lst:
        s+=len(st)
    h=0
    for str in lst:
        freq=0
        for ch in str:
            if ch=='a':
                freq+=1
        if freq>h:
            h=freq
    for str in lst:
        freq=0
        for ch in str:
            if ch=='a':
                freq+=1
        if freq==h:
            print(str)
    return s

print(fun(['aarti','aaaaart','a452ahb','aaa']))
```

##Output:

```
aaaaart
##22
```

'''DICTIONARIES'''

Create a dictionary containing names of competition winners as key and no of wins as valuee

```
n=int(input("enter no. of students:"))
c={}
for a in range(n):
    key=input("Name of student:")
    value=int(input("No of wins:"))
    c[key]=value
print("The dictionary now is :")
print(c)
```

#Output:

```
The dictionary now is :
{'Aarti': 10, 'Jay': 1}
```

Program to count the frequency of a list element using dict

```
import json
s='''This boy sitting next to me
is a psycopath.So beware of him cause
This boy can be a murderer as well.'''
w=s.split()
d={}
for a in w:
    key=a
    if key not in d:
        c=w.count(key)
        d[key]=c
print("Counting frequencies \n",w)
print(json.dumps(d,indent=1))
```

Menu Driven

```
import mysql.connector
d=mysql.connector.connect(host='localhost',user='root'
,password='aarti')
c=d.cursor()
c.execute("create database HHW")
c.execute("use HHW")
c.execute("create table cwc(sno int,Team
char(20),Bestplayer char(20),won int,lost int)")
while True:
    x=int(input("Menu:\n 1.Delete all \n 2.Add Records
\n 3.Display Records \n 4.Exit"))
    if x==1:
        c.execute("delete from cwc;")
        d.commit()
        print("All records have been deleted")
    elif x==2:
        n=int(input("enter no. of records:"))
        for w in range(n):
            s=int(input("enter sno:"))
            t=input("enter name of country:")
            b=input("enter name of best player:")
            w=int(input("enter no of matches won:"))
            l=int(input("enter no of matches lost:"))
            k="insert into cwc
values({},'{}','{}',{},{})".format(s,t,b,w,l)
            c.execute(k)
            d.commit()
    elif x==3:
        c.execute("select * from cwc;")
        r=c.fetchall()
        for x in r:
            print(x)
    elif x==4:
        d.commit()
        break
```

#Output:

```
Menu:
 1.Delete all
 2.Add Records
 3.Display Records
 4.Exit3
(1, 'India', 'Virat Kohli', 8, 1)
(2, 'England', 'Jason Roy', 6, 3)
(3, 'Australia', 'David Warner', 8, 1)
(4, 'Afghanistan', 'Rashid Khan', 0, 9)
Menu:
 1.Delete all
 2.Add Records
 3.Display Records
 4.Exit4
```

MYSQL WITH PYTHON

```
import mysql.connector as sqlc

co=sqlc.connect(host="localhost",user="root",password=
"123456",charset='utf8')


if co.is_connected():

    print("Connection ok")

else:

    print("Fail")


print("connectedddddd",co.is_connected())   #True /
False


##curobj=co.cursor()

curobj=co.cursor(buffered=True)


#alows new data in the cursor even if #Unread result
found

curobj.execute('create database if not exists
employee')

curobj.execute('use employee')

curobj.execute('create table if not exists emp(eno
int,name char (20))')

curobj.execute("insert into emp
values(14,'madhuri'),(12,'dhruv')")

curobj.execute("commit")
```

```
curobj.execute("select * from emp")
rec=curobj.fetchone() #list of tuple
print(rec)
print("Total no of records affected ",curobj.rowcount)

curobj.reset()

curobj.execute("select * from emp")
rec=curobj.fetchone() #list of tuple
print(rec)
print("Total no of records affected ",curobj.rowcount)

recs=curobj.fetchall() #list of tuple
for r in recs:
   print(r)
print("Total no of records are ",curobj.rowcount)
```

OUTPUT:

```
Connection ok
connecteddddddd True
rrrrrrrrr 2
records are  [(4, 'madhuri'), (2, 'dhruv')]
(4, 'madhuri') (2, 'dhruv')
```

```
curobj.execute("desc emp")
print('Heading ------> ',curobj.column_names) #tuple

ds=curobj.fetchall() #list of tuple
print(ds)

'''
Heading ------>  ('Field', 'Type', 'Null', 'Key',
'Default', 'Extra')
[('eno', 'int(11)', 'YES', '', None, ''), ('name',
'char(20)', 'YES', '', None, '')]

'''
print("\n"*3)
print("1.Executing command : show databases")
curobj.execute('show databases')
for w in range(7):
   seq=curobj.fetchone()
   print(1,seq)
   print("rowcountttt",curobj.rowcount)
   print()

print('stmt------> ',curobj.statement)
```

```
print('column names ------> ',curobj.column_names)
#tuple


print("\n"*3)
print("2Executing command : show databases")
curobj.execute('show databases')
seq=curobj.fetchall()
print('ALLLLLL- > ',seq)
print("rowcountttt",curobj.rowcount)
print()



print("\n"*3)
print("3Executing command : show databases")
curobj.execute('show databases')
seq=curobj.fetchmany(3)
print('ALLLLLL- > ',seq)
print("rowcountttt",curobj.rowcount)
print()



print("rowcountttt",curobj.rowcount)
print('stmt------> ',curobj.statement)
print('stmt------> ',curobj.column_names) #tuple
vv=curobj.fetchall()
```

```
print(vv)
print("rowcountttt",curobj.rowcount)
```

OUTPUT

```
1.Executing command : show databases
1 ('information_schema',)
rowcountttt 1

1 ('employee',)
rowcountttt 2

1 ('mysql',)
rowcountttt 3

1 ('sch',)
rowcountttt 4

1 ('test',)
rowcountttt 5

1 None
rowcountttt 5

1 None
rowcountttt 5
```

```
stmt------>  show databases
column names ------>  ('Database',)

2Executing command : show databases
ALLLLLL- >  [('information_schema',), ('employee',),
('mysql',), ('sch',), ('test',)]
rowcounttttt 5

3Executing command : show databases
ALLLLLL- >  [('information_schema',), ('employee',),
('mysql',)]
rowcounttttt 3

rowcounttttt 3
stmt------>  show databases
stmt------>  ('Database',)
[('sch',), ('test',)]
rowcounttttt 5
```

```
curobj=dbobj.cursor()
##curobj=dbobj.cursor(buffered=True)
#alows new data in the cursor even if #Unread result
found
curobj.execute('create database if not exists wooden')
curobj.execute('use wooden')
x=3
qry="select * from abc1 where price>{}".format(x)
curobj.execute(qry)
recc=curobj.fetchall()
print('6666',recc)
##input()

curobj.execute('show databases')
seq=curobj.fetchone()
print(1,seq)
print("rowcountttt",curobj.rowcount)
print('stmt------> ',curobj.statement)
print('stmt------> ',curobj.column_names) #tuple
curobj.fetchall()
##input()

x=3
qry="select * from abc1 where price>{}".format(x)
print("ok4")
```

```
#curobj.execute(qry)
###mysql.connector.errors.InternalError:
###Unread result found

#cur2=dbobj.cursor()
###mysql.connector.errors.InternalError:
###Unread result found
##curobj.reset()
##seq=curobj.fetchone()
##print(1,seq)
##print("rowcountttt",curobj.rowcount)

##curobj.close()

curobj.execute(qry)

seq=curobj.fetchone()
print(2,seq) #None is no data / tuple is record found
seq=curobj.fetchone()
print(3,seq)
seq=curobj.fetchone()
print(4,seq)
```

```
seq=curobj.fetchmany(3) #empty list if no data / or
list of tuple
print(567,seq)  #list of tuple


print("rowcountttt",curobj.rowcount)
```

```
import mysql.connector
d=mysql.connector.connect(host='localhost',user='root'
,password='aarti')
c=d.cursor()
c.execute("create database HHW")
c.execute("use HHW")
c.execute("create table cwc(sno int,Team
char(20),Bestplayer char(20),won int,lost int)")
while True:
    x=int(input("Menu:\n 1.Delete all \n 2.Add Records
\n 3.Display Records \n 4.Exit"))
    if x==1:
        c.execute("delete from cwc;")
        d.commit()
        print("All records have been deleted")
    elif x==2:
        n=int(input("enter no. of records:"))
        for w in range(n):
            s=int(input("enter sno:"))
            t=input("enter name of country:")
            b=input("enter name of best player:")
            w=int(input("enter no of matches won:"))
            l=int(input("enter no of matches lost:"))
            k="insert into cwc
values({},'{}','{}',{},{})".format(s,t,b,w,l)
            c.execute(k)
            d.commit()
    elif x==3:
        c.execute("select * from cwc;")
        r=c.fetchall()
        for x in r:
            print(x)
    elif x==4:
        d.commit()
        break
```

#Output:

Menu:

```
 1.Delete all
 2.Add Records
 3.Display Records
 4.Exit3
(1, 'India', 'Virat Kohli', 8, 1)
(2, 'England', 'Jason Roy', 6, 3)
(3, 'Australia', 'David Warner', 8, 1)
(4, 'Afghanistan', 'Rashid Khan', 0, 9)
Menu:
 1.Delete all
 2.Add Records
 3.Display Records
 4.Exit4
```

BINARY FILE

```
import pickle
file=open("binf.dat", 'wb')
intv=3
floatv=4.5
listv=[5,6,7]
dictv={2:22,3:33}
pickle.dump(intv,file)
#pickle.dump(variable,fileObject)
pickle.dump(floatv,file)
pickle.dump(listv,file)
pickle.dump(dictv,file)
file.close()
file=open("binf.dat", 'rb')
v=pickle.load(file)
print(v)
v=pickle.load(file)
print(v)
v=pickle.load(file)
print(v)
v=pickle.load(file)
print(v)
file.close()
```

output

```
3
4.5
[5, 6, 7]
{2: 22, 3: 33}
```

```
from pickle import *
file=open("binf.dat", 'wb+')
intv=3
floatv=4.5
listv=[5,6,7]
dictv={2:22,3:33}
dump(intv,file)
dump(floatv,file)
dump(listv,file)
dump(dictv,file)
#pickle.dump(dictv,file) #NameError: name 'pickle' is
not defined
##import pickle
##pickle.dump(dictv,file) #no error
file.seek(0)
v=load(file)
print(v)
v=load(file)
print(v)
v=load(file)
print(v)
v=load(file)
print(v)
file.close()
```

output

```
3
4.5
[5, 6, 7]
{2: 22, 3: 33}
```

CSV FILE

```
import csv
f=open('csv5.csv','r')
r=csv.reader(f)
print(r) #object address
print(list(r))
#reads entire file and makes nested list
f.seek(0)
for w in r:
   print("***",w)
f.close()
```

Output

```
<_csv.reader object at 0x0000006FD22B9B40>
[['rno', 'name', 'sec'], ['4', 'ram', 'c'], ['5',
'raheem', 'c'], ['7', 'kareem', 'b'], ['1', 'shyam',
'a'], ['2', 'sundar', 'a'], ['3', 'mohan', 'a'], ['8',
'ghanshyam', 'a']]
*** ['rno', 'name', 'sec']
*** ['4', 'ram', 'c']
*** ['5', 'raheem', 'c']
*** ['7', 'kareem', 'b']
*** ['1', 'shyam', 'a']
*** ['2', 'sundar', 'a']
*** ['3', 'mohan', 'a']
```

```
import csv
def f():
##    import csv
   f= open('CSV5.csv', 'r')
   print("start")
   x=csv.reader(f)
   s=0
   p=next(x) #function next reads 1 record
   print("header--",p)
   for w in x:
       print(w) #list form
       if w[2]=='a':
           s+=int(w[0])
   print(s)
   print("end")
   f.close()
```

OUTPUT

```
start
header-- ['rno', 'name', 'sec']
['4', 'ram', 'c']
['5', 'raheem', 'c']
['7', 'kareem', 'b']
['1', 'shyam', 'a']
['2', 'sundar', 'a']
```

```
['3', 'mohan', 'a']
['8', 'ghanshyam', 'a']
14
end
```

SEARCHING

```
def linear_Search(list1, n, key):

    # Searching list1 sequentially
    for i in range(0, n):
        if (list1[i] == key):
            return i
    return -1



list1 = [1 ,3, 5, 4, 7, 9]
key = 7


n = len(list1)
res = linear_Search(list1, n, key)
if(res == -1):
    print("Element not found")
else:
    print("Element found at index: ", res)
```

OUTPUT

```
Element found at index:  4
```

```
# Binary Search in python


def binarySearch(array, x, low, high):

    # Repeat until the pointers low and high meet each
other
    while low <= high:

    mid = low + (high - low)//2

    if array[mid] == x:
        return mid

    elif array[mid] < x:
        low = mid + 1

    else:
        high = mid - 1

    return -1


array = [3, 4, 5, 6, 7, 8, 9]
x = 4

result = binarySearch(array, x, 0, len(array)-1)

if result != -1:
    print("Element is present at index " +
str(result))
else:
    print("Not found")
```

MYSQL QUERIES

```
create database db1;

use db1;


CREATE TABLE customers

(id int(10),

 name varchar(50),

 city varchar(50),

 PRIMARY KEY (id )

);


ALTER TABLE customers

ADD age varchar(50);

insert into customers values(101,'rahul','delhi');

update customers set name='bob', city='london' where
id=101;


delete from customers where id=101;

SELECT * from customers;


truncate table customers;

drop table customers;
```

```
SELECT officers.officer_name, officers.address,
students.course_name

FROM officers

INNER JOIN students

ON officers.officer_id = students.student_id;

SELECT  officers.officer_name, officers.address,
students.course_name

FROM officers

LEFT JOIN students

ON officers.officer_id = students.student_id;

SELECT columns

FROM table1

RIGHT [OUTER] JOIN table2

ON table1.column = table2.column;

SELECT officers.officer_name, officers.address,
students.course_name, students.student_name

FROM officers

RIGHT JOIN students

ON officers.officer_id = students.student_id;

SELECT *

FROM customers

CROSS JOIN contacts;
```