PROGRAM:11

Write a function EUCount() in C++,which should reach each character of a text file IMP.TXT, should count and display the occurrence of alphabets E and U(including small cases e and u too).
Example:

- If the file content is as follows:

Updated information
is simplified by official websites.

The EUcount() function should display the output as

E:4
U:1

SOLUTION:

```
void EUCount()
{
     ifstream fin;
     fin.open("IMP.TXT",ios::in);
    char ch;
    int ecount =0,ucount =0;
    while(!fin.eof())
   {
        fin.get(ch);
        if(ch== 'e' || ch == 'E')
                ecount++;
        else if(ch== 'u'|| ch== 'U')
              ucount++;
}
cout<< " E:"<<ecount<<endl;
cout<< "U:"<<ucount<<endl;
fin.close();
}
```

PROGRAM:12

Write function definition for TOWER() in C++ to read the content of a file WRITEUP.TXT,count the presence of word TOWER and display the number of occurrences of this word.
Note.

- The word TOWER should be an independent word
- Ignore type cases(i.e., lower/upper case)

Example:

If the content of the file WRITEUP>TXT is as follows:

Tower of Hanoi is an interesting problem. Mobile phone tower is away from here. Views from EIFFEL TOWER are amazing.

The function TOWER() should display the following:
3

SOLUTION:

```
void TOWER( )
{
  ifstream fin;
fin.open("WRITEUP.TXT");
char word[10];
int count =0;
```

```
while(!fin.eof())
{
    fin>>word;
  if(strcmpi(word, "TOWER")== 0)
   count++;
}
cout<<count;
fin.close();
}
```

PROGRAM:13

Write a function in C++ to search for a camera from a binary file " CAMERA.DAT" containing the objects of class CAMERA(as defined below). The user should enter the Model No and the function should search and display the details of the camera.

```
Class CAMERA
{
     long ModelNo;
    float MegaPixel;
    int Zoom;
   char Details[120];
public:
      void Enter()
{
     cin>>ModelNO>>MegaPixel>>Zoom;
     gets(Details):
}
Void Display( )
{
    cout<<ModelNO<<MegaPixel<< Details<<endl;
}
long GetModelNO( )
{
     return ModelNo;
}
};
```

SOLUTION:

```
void search(long MNo)
{
     ifstream ifile("CAMERA.DAT", ios::in|ios::binary);
if(! ifile)
{
   cout<<" could not open CAMERA.DAT file";
      exit(-1);
}
else
{
    CAMERA c ;
     int found = 0;
while(ifile.read((char *)&c, sizeof(c )))
{
```

```
    if(c.GetModelNo( )==MNo)
{
   c.Display( );
   found=1;
   break;
}
}
}
if (found ==0)
 cout<< " Given ModelNo not found";
}
```

PROGRAM:14

Write a function in C++ which accepts an array of Student records and its size as arguments / parameters and arrange all records in ascending order of the field total. Apply selection Sort technique. The description of the student record is given below:

```
struct student
{
int roll;
int total;
};
```

SOLUTION:

```
#include<iostream.h>

struct student
{
int roll;
int total;
};

void selsort(student ob[], int n)
{
int min, pos;
student temp;
for(int i=0;i<n-1;i++)
{
min=ob[i].total;
pos=i;
for(int j=i+1;j<n;j++)
if(ob[j].total<min)
{
min=ob[j].total;
pos=j;
}
temp=ob[i];
ob[i]=ob[pos];
ob[pos]=temp;
}
}

void show(student ob[], int n)
{
cout<<"\n showing data....\n";
```

```
for(int i=0;i<n;i++)
cout<<ob[i].roll<<" "<<ob[i].total<<"\n";
}
void main()
{
student ob[5]={ {101,98}, {102, 56}, {103,90}, {104,78},{105,65}};
show(ob,5);
selsort(ob,5);
show(ob,5);
}
```

Program:15

Write a user-defined function mergesort() that takes two arrays of integers A and B and their sizes as parameters/arguments. Assume array A and array B are sorted in ascending and descending order respectively. Apply mergesort technique to combine array A and array B into array C.

SOLUTION:

```
#include<iostream.h>
#include<string.h>

void show(int a[], int n)
{
cout<<"\n showing data....\n";
for(int i=0;i<n;i++)
cout<<a[i]<<" ";
cout<<endl;
}

void mergesort(int a[], int b[], int m, int n)
{
int i, j, k;
i=k=0;
j=m-1;
int c[30];
while(i<m && j>=0)
{
if(a[i]<b[j])
c[k++]=a[i++];
else
c[k++]=b[j--];
}
while(i<m)
c[k++]=a[i++];
while(j>=0)
c[k++]=b[j--];
show(c,k);
}

void main()
{
int a[]={ 11,22,33,44, 55};
int b[]={ 50,40,30,20,10};
show(a,5);
```

```
show(b,5);
mergesort(a,b,5,5);
}
```

PROGRAM:16

Write a function in C++ which accepts a 2-D array of integers and its size as arguments and prints the elements present on the left and right diagonals.

SOLUTION:

```
#include<iostream.h>
void showdiagonal(int a[10][10],int r, int c)
{
cout<<"\n diagonal one :";
for(int i=0;i<r;i++)
{
for(int j=0;j<c;j++)
{
if(i==j)
cout<<a[i][j]<<"\t";
}
}
cout<<"\n\n diagonal two :";
for(i=0;i<r;i++)
{
for(int j=0;j<c;j++)
{
if(i+j==(r-1))
cout<<a[i][j]<<"\t";
}
}
}

void main()
{
int x[10][10]={ { 5,4,3}, {6,7,8}, {1,2,9}};
showdiagonal(x,3,3);
}
```

PROGRAM:17

Write a function in C++ which accepts a 2-D array of integers and its size as arguments. Assuming array as square matrix i.e. MxM dimensions, exchanges elements of first column with the elements of the last column.

SOLUTION:

```
#include<iostream.h>
#include<string.h>

void exchangecol(int a[10][10],int r, int c)
{
        int first=0,last=c-1,temp;
        for(int j=0;j<r;j++)
        {
                temp=a[j][first];
```

```
                a[j][first]=a[j][last];
                a[j][last]=temp;

        }
}

void show(int a[10][10], int r, int c)
{
        cout<<"\n show ....\n";
        for(int i=0;i<r;i++)
        {
                for(int j=0;j<c;j++)
                        cout<<a[i][j]<< " ";
                cout<<endl;

        }
}
                void main()
                {
                        int x[10][10]={ { 5,4,3}, {6,7,8}, {1,2,9}};
                        show(x,3,3);
                        exchangecol(x,3,3);
                        show(x,3,3);

                }
```

PROGRAM:18

Given below is the description of class implementing array based Stack. Define function push() and pop(). Test the class in main().

SOLUTION:

```
#include<iostream.h>
#include<string.h>
#define MAX 6
class Stack
{
        int stk[MAX];
        int top;
public:
        Stack() { top=-1; }
        void push(int x);
        void pop();
        void show();
};

void Stack::push(int x)
{
        if(top==MAX-1)
        {
                cout<<"\n stack overflows..";
                cout<<"\n stack full...";
                return;
        }
        top=top+1;
        stk[top]=x;
```

```
}void Stack::pop()
{
        if(top==-1)
        {
                cout<<"\n stack underflows...";
                cout<<"stack empty...";
                return;
        }
        int val=stk[top];
        top--;
        cout<<"\n value popped.."<<val;
}

void Stack::show()
{
        if(top==-1)
        {cout<<"\n stack is empty....\n";
                return;
        }
        cout<<"\n  stack : ";
        for(int i=top;i>=0;i--)
                cout<<stk[i]<<" ";
}



void main()
{
        Stack ob;
        ob.push(1);  ob.push(2);  ob.push(3);  ob.show();
        ob.pop(); ob.pop(); ob.pop(); ob.pop(); ob.show(); ob.push(4) ;
        ob.push(5); ob.show();
        ob.pop();        ob.show();
}
```

PROGRAM:19

Given below is the description of class implementing array based Queue. Define function add() and del(). Test the class in main().

SOLUTION:

```
#include<iostream.h>
#include<string.h>
#define MAX 6

class Queue
{
        int Q[MAX];
        int front,rear;
public:
        Queue() { front=rear=-1; }
        void add(int x);
        void del();
```

```
        void show();
};

void Queue::add(int x)
{
        if(rear==MAX-1)
        {
                cout<<"\n queue overflows..";
                cout<<"\n queue is full...";
                return;
        }
        rear=rear+1;
        Q[rear]=x;


}

void Queue::del()
{
        if(front==rear)
        {
                cout<<"\n queue underflows...";
                cout<<"queue empty...";
                return;
        }
        front=front+1;
        int val=Q[front];
        cout<<"\n value removed.."<<val;
}


void Queue::show()
{
        if(front==rear)
        {cout<<"\n queue is empty....\n";
                return;
        }
        cout<<"\n showing queue status : ";
        for(int i=front+1;i<=rear;i++)
                cout<<Q[i]<<" ";
}



void main()
{
        Queue ob;
        ob.add(1);   ob.add(2);   ob.add(3);   ob.show();
        ob.del();          ob.del();          ob.del();          ob.del();          ob.show();
        ob.add(4);  ob.add(5); ob.show();
        ob.del();          ob.show();
}
```

PROGRAM:20

Given below is the description of class implementing array based Circular Queue. Define function add() and del(). Test the class in main().

SOLUTION:

```
#include<iostream.h>
#define N 5
#define pr cout<<"\n"<<

class Queue
{
        int *queue;
        int MAX;
        int front,rear;
public:
        Queue(int size) { queue=new int[MAX=size]; front=0;rear=0;}
        ~Queue() { delete [] queue; }

        void insertq(int item)
        {
                pr "front="<<front <<" rear ="<<rear;
                if( (rear+1)%MAX==front)
                {
                        pr "queue overflows";
                        return;
                }

                queue[rear]=item;

                rear=(rear+1) % MAX;

        }

        int deleteq()
        {
                if(front==rear)
                {
                        pr "queue underflows";
                        return -1;

                }
                int value=queue[front];
                front=(front+1)%MAX;

                pr "deleted value :"<<value;
                //      pr "front="<<front <<" rear ="<<rear;
                return value;
        }

        void delitem(int item)
        {
```

```
                int *temp=new int[MAX];
                int i;
                i=front; int k=0;

                while(i!=rear)
                {
                        if(item!=queue[i])
                                temp[k++]=queue[i];
                                                i=(i+1)%MAX;
                }
                pr "value of k :"<<k;
                i=front;int j=0;
                front=j; rear=k;
                while(j <k)
                                queue[j]=temp[j++];




        }
        void traverse()
        {
                int i;
                i=front;
                pr "traversing array \n";
                while(i!=rear)
                {
                        cout<<queue[i]<<" ";
                        i=(i+1)%MAX;
                }
        }
};

void main()
{
Queue Q(6);
Q.insertq(1); Q.insertq(2); Q.insertq(3);
Q.insertq(4);Q.insertq(5); Q.insertq(6);
Q.traverse();
Q.deleteq(); Q.deleteq();
Q.traverse();
Q.insertq(11); Q.insertq(12); Q.insertq(13);
Q.traverse();
}
```

PROGRAM:21

Consider the following structure of node, which implements a linked list based Stack of names. Write the definition of function PUSH( ), to insert a new node in the stack and definitions of function POP(), to delete a node from the stack

```
struct node
{
char na[30];
node *next;
};
```

SOLUTION:

```
#include<iostream.h>
#include<conio.h>
#define tab '\t'
#define pr cout<<"\n"<<

struct node
{
char na[30];
node *next;
};

node *top;

void push(char *str)
{
node *ptr=new node;
if(ptr==NULL)
{
pr "Node cannot be created :\n";
return;
}

strcpy(ptr->na,str);
ptr->next=NULL;

if(top==NULL)
top=ptr;
else
{
ptr->next=top;
top=ptr;
}
}

void pop()
{
node *temp;

if(top==NULL)
{
pr "No Node exists :\n";
return;
}
temp=top;
top=top->next;
```

```
pr " Node deleted :"<<temp->na<<endl;
delete temp;
}
void traverse()
{
node *temp;
if(top==NULL)
return;
pr " Traversing List \n";
temp=top;
while(temp!=NULL)
{
cout<< temp->na<<tab;
temp=temp->next;
}
}
void main()
{
top=NULL;
push("India");
push("Japan");
push("China");
push("England");
traverse();
pop();
pop();
traverse();
}
```

PROGRAM:22

Consider the following structure of node, which implements a linked list based Queueof names. Write the definition of function INSERT( ), to insert a new node in the stack and definitions of function REMOVE(), to delete a node from the queue. struct node

{ char na[30]; node *next; };

SOLUTION:

```
#include<iostream.h>
#include<conio.h>
#define tab '\t' #define pr
cout<<"\n"<< struct
node
{
char na[30];
node  *next;
};
node *front,*rear;
void insert(char *str)
{
node *ptr=new node;
if(ptr==NULL)
{
pr "Node cannot be created :\n";
return;
```

```
}
strcpy(ptr->na,str);
ptr->next=NULL;
if(front==NULL)
front=rear=ptr;
else
{
rear->next=ptr;;
rear=ptr;
}
}

void remove()
{
node *temp;
if(rear==NULL)
{
pr "No Node exists :\n";
return;
}
temp=front;
front=front->next;
pr " Node deleted :"<<temp->na<<endl;
delete temp;
}

void traverse()
{
node *temp;
if(front==NULL)
return;
pr " Traversing List \n";
temp=front;
while(temp!=NULL)
{
cout<< temp->na<<tab;
temp=temp->next;
}
}
void main()
{
front=rear=NULL;
insert("India");
insert("Japan");
insert("China");
insert("England");
traverse();
remove();
remove();
traverse();
}
```

SQL COMMANDS

(1) Given the following student relation:

NO.	NAME	AGE	DEPARTMENT	DATEofadm	FEE	SEX
1.	Pankaj	24	Computer	10/01/97	120	M
2.	Shalini	21	History	24/03/98	200	F
3.	Sanjay	22	Hindi	12/12/96	300	M
4.	Sudha	25	History	1/7/99	400	F
5.	Rakesh	22	Hindi	5/9/97	250	M
6.	Shakel	30	Histoty	27/6/98	300	M
7.	Surya	34	Computer	25/2/97	210	M
8.	Shikha	23	Hindi	31/7/97	200	F

Write SQL commands for (a) to (f) and write output for (g)

(a) To show all information about the student of History department

(b) To list the names of female students who are Hindi department

(c) To list names of all students with their date of admission in ascending order.

(d) To display student's Name, Fee, Age for male Students only.

(e) To count the number of student with Age < 23

(f) To insert a new row in the STUDENT table with the following data:
9, "Zaheer",36, "Computer",{12/03/97},230, "M"

(g) Give the output of following SQL statements:

i. Select COUNT(distinct department) from STUDENT;

ii. Select MAX(Age) from STUDENT where SEX= "F";

iii. Select AVG(Fee) from STUDENT where Dateofadm<{01/01/98};

iv. Select SUM(Fee) from STUDENT where Dateofadm<{01/01/98};

Sol:

(a) SELECT * FROM Student
WHERE Department = "History";

(b) SELECTB Name FROM Student
WHERE sex= "F" and Department = " Hindi";

(c) SELECT name FROM Student ORDER BY Dateofadm;

(d) SELECT Name, Fee,Age FROM Student WHERE sex = "M";

(e) SELECT COUNT(*)FROM Student WHERE Age <23;

(f) INSERT INTO Student
VALUES(9,"Zaheer","Computer","12/03/97",230,"M");

(g) (i) 3 (ii) 25 (iii) 216 (iv) 1080

(2) Given the following tables for a database LIBRARY:

Table : BOOKS

BOOK_ID	BOOK_NAME	AUTHOR_NAME	PUBLISHERS	PRICE	TYPE	QTY
C0001	Fast Cook	Lata Kapoor	EPB	355	Cookery	5
F0001	The Tears	William Hopkins	First Publ.	650	Fiction	20
T0001	My First C++	Brian & Brooke	EPB	350	Text	10
T0002	C++Brainworks	A.W Rossaine	TDH	350	Text	15
F0002	Thunderbolts	Anna Roberts	First Publ.	750	Fiction	50

Table : ISSUED

BOOK_ID	QUANTITY_ISSUED
T0001	4
C0001	5
F0001	2

Write SQL queries for (a) to (f) :

(a) To show Book name, Author name and Price of books of First Publ. publishers.

(b) To list the names from books of Text type.

(c) To display the names and price from books in ascending order of their price.

(d) To increase the price of all books of EPB Publishers by 50.

(e) To display the Book_Id, Book_Name and Quantity_Issued for all books which have been issued. (The query will require contents from both the tables.)

(f) To insert a new row in the table Issued having the following data :" F0003",1

(g) Give the output of the following queries based on the above tables :

i) SELECT COUNT(*) FROM Books ;

ii) SELECT MAX(Price) FROM Books WHERE Quantity >=15;

iii) SELECT Book_Name, Author_Name FROM Books WHERE Publishers='EPB";

iv) SELECT COUNT (DISTINCT Publishers) FROM Books WHERE Price >=400;

Sol:

a) SELECT Book_Name, Author_Name,Price
FROM Books
WHERE Publishers= "First Publi." ;

b) SELECT Book_Name
FROM Books
WHERE Type=”Text”;

c) SELECT Book_Name, Price
FROM Books
ORDER By Price;

d) UPDATE Books
SET Price =Price+50
WHERE Publishers = “EPB”;

e) SELECT Books.Book_ID, Book_Name, Quantity_Issued
FROM Books,Issued
WHERE Books.Book_ID = Issued.Book_ID;

f) INSERT INTO Issued
VALUES(“F0003”,1);

g) i) 5 ii) 750 iii) Fast Cook Lata Kapoor iv) 1
My First C++ Brian & Brooke

(3) **Table :** Stock

ItemNo	Item	Dcode	Qty	UnitPrice	StockDate
5005	Ball Pen 0.5	102	100	16	31-Mar-10
5003	Ball Pen 0.25	102	150	20	01-Jan-10
5002	Gel Pen Premium	101	125	14	14-Feb-10
5006	Gel Pen Classic	101	200	22	01-Jan-09
5001	Erraser Small	102	210	5	19-Mar-09
5004	Eraser Big	102	60	10	12-Dec-09
5009	Sharpener Classic	103	160	8	23-Jan-09

Table : Dealers

Dcode	Dname
101	Reliable Stationers
103	Classic Plastics
102	Clear Deals

(a1) Write SQL commands for the following statements:

i) To display details of all Items in the Stock table in ascending order of StockDate.

ii) To Display ItemNo and Item name of those items from Stock table whose UnitPrice is more than Rupees 10.

iii) To display the details of those items whose dealer code (Dcode) is 102 or Quantity in Stock(Qty) is more than 100 from the table Stock..

iv) To display Maximum UnitPrice of items for each dealer individually as per Dcode from the table Stock.

Sol: (a1)

i)
```
SELECT *
FROM Stock
ORDER BY StockDate;
```

ii)
```
SELECT ItemNo,Item
FROM Stock
WHERE Unitprice > 10;
```

iii)
```
SELECT *
FROM Stock
WHERE Dcode = 102 OR Qty > 100;
```

iv)
```
SELECT Dcode, MAX ( UnitPrice)
FROM Stock
GROUP BY Dcode;
```

(a2) (i) 3 (ii) 4400 (iii) Eraser Big Clear Deals (iv)01-Jan-09

(4) Consider the following tables EMPLOYEE and SALGRADE and answer (A1) and (A2) parts of this question:

Table:EMPLOYEE

ECODE	NAME	DESIG	SGRADE	DOJ	DOB
101	**Abdul Ahmed**	**EXECUTIVE**	**S03**	**23-Mar-2003**	**13-Jan-1980**
102	**Ravi Chander**	**HEAD –IT**	**S02**	**12-Feb-2010**	**22-Jul-1987**
103	**John Ken**	**RECEPTIONIST**	**S03**	**24-Jun-2009**	**24-Feb-1983**
105	**Nazar Ameen**	**GM**	**S02**	**11-Aug-2006**	**3-Mar-1984**
108	**Priyam Sen**	**CEO**	**S01**	**29-Dec-2004**	**19-Jan-1982**

Table: SALGRADE

SGRADE	SALARY	HRA
S01	**56000**	**18000**
S02	**32000**	**12000**
S03	**24000**	**8000**

(A1) Write SQL commands for the following statements:

(i) To display the details of all EMPLOYEEs in descending order of DOJ.

(ii) To display NAME and DESIG those EMPLOYEEs, whose SALGRADE is either S02 or S03

(iii) To display the content of all the EMPLOYEEs table whose DOJ is in between '09-Feb-2006' and '08-Aug-2009'

(iv) To add a new row with the following: 19, 'Harish Roy', 'HEAD-IT', 'S02', '09-Sep-2007' , '21-Apr-1983'

(A2) Give the output of the following SQL queries:

(i) SELECT COUNT (SGRADE),SGRADE FROM EMPLOYEE GROUP BY SGRADE;

(ii) SELECT MIN(DOB),MAX(DOJ) FROM EMPLOYEE;

(iii)SLECT NAME,SALARY FROM EMPLOYEE E, SALGRADE S WHERE E.SGRADE=S.SGRADE AND E.ECODE<103;

(iv) SELECT SGRADE,SALARY +HRA FROM SALGRADE WHERE SGRADE= 'S02';

SOL : (A1)

(i)SELECT * FROM EMPLOYEE ORDER BY DOJ DESC;

(ii) SELECT NAME,DESIG FROM EMPLOYEE WHERE SALGRADE IN('S02', 'S03');

(iii)SLECT *FROM EMPLOYEE WHERE DOJ BETWEEN '09-Feb-2006' AND '08-Aug-2009';

(iv) INSERT INTO EMPLOYEE VALUES (19, 'Harish Roy', 'S02', '09-Sep-2007', '21-Apr-1983');

(A2)

(i)

COUNT	SGRADE
2	S03
2	S02
1	S01

(ii)

13-Jan-1980	12-Feb-2010

(iii)

NAME	SALARY
Abdul Ahmed	24000
Ravi Chander	32000

(iv)

SGRADE	SALARY+HRA
S02	44000