FS LAB PROGRAMS

NOTE:

To create and execute a programs use these commands on UBUNTU:

gedit program_name.cpp [ex: gedit p1.cpp]

g++ program_name.cpp [ex: gedit p1.cpp]

./a.out

PROGRAM 1a

```
#include<iostream>

#include<string>

#include<stdlib.h>


using namespace std;

int main()

{

   string name , rev;

   int count , i , j;

   system("clear");

   cout<<"\nEneter the number of names: \n";

   cin>>count;

   for( i = 0; i < count; i++)

   {

      cout<<"\n Enter name: ";

      cin>>name;

      rev.erase();

      for( j = name.length()-1; j >= 0; j--)

      {

         rev += name[j];

      }

      cout<<"\n Reversed "<<rev <<endl;


   }
   return 0;

}
```

Output:

p1a output

```
Eneter the number of names: 2

Enter name: Avinash

Reversed hsanivA

Enter name: Chauhan

Reversed nahuahC
```

PROGRAM 1b

```
#include<iostream>

#include<string.h>

#include<fstream>

#include<stdlib.h>


using namespace std;


int main()

{

   string name , rev;

   char infile[30] , outfile[30];

   fstream fpinp , fpoutp;


   int j;

   system("clear");
```

```
    cout<<"Enter the input filename\n";

    cin>>infile;

    cout<<"Enter the output filename\n";

    cin>>outfile;

    fpinp.open(infile , ios::in);

    fpoutp.open(outfile , ios::out);

    if(!fpinp || !fpoutp)

    {

        cout<<"FATAL ERROR! Unable to open the files";

        exit(0);

    }


    while(fpinp){

        getline(fpinp , name);

        rev.erase();

        for(j= name.length()-1 ; j>= 0 ; j--)

        {

            rev += name[j];

        }

        fpoutp<<rev<<endl;

    }

    fpinp.close();

    fpoutp.close();

    return 0;

}
```

Output:

```
Enter the input filename
name.txt
Enter the output filename
out.txt
```

PROGRAM 2

Write a C++ program to read and write and student objects with fixed-length records and the fields delimited by "|" . implement pack(), unpack(), modify() and search() methods.

```
#include<fstream>
#include<string>
#include<iostream>
#include<cstring>
using namespace std;
class student
{
        public:
                string usn;
                string name;
                string sem;
                string Buf;
                char buf[100];
                void pack();
                void write_f(fstream &);
                void unpack();
                void print(ostream &);
                void read_f(fstream &);
};
void student::pack()
{
Buf=usn+"|"+name+"|"+sem+"\n";
}
void student::write_f(fstream &fp)
{
        fp<<Buf;
```

```
}
void student::print(ostream &stream)
{
        stream<<"student:\n"
       <<"\t usn'"<<usn<<"'\n"
                <<"\t name'"<<name<<"'\n"
                <<"\t sem'"<<sem<<"'\n";
}
void student::unpack()
{
        char stg[100];
        int pos=0,count=0,k;
        while(count<3)
        {
                k=0;
                for(int i=pos;i<strlen(buf);i++,k++)
                {
                        stg[k]=buf[i];
                        pos++;
                        if(buf[i]=='|')
                                break;
                }

                stg[k]='\0';
                count++;
                if(count==1) usn=stg;
                if(count==2) name=stg;
                if(count==3) sem=stg;
        }
}
void student::read_f(fstream &fp)
{
        fp.getline(buf,100,'\n');
```

```
}
int main()
{
        int ch;
        fstream fp;
        void search();
        student s;
        system("clear");
        do
        {
                cout<<"enter your choice\n";
                cout<<"1.insert a record\n"
                        <<"2.search and modify a record\n"
                        <<"3.exit\n";
                cin>>ch;
                switch(ch)
                {
                        case 1:fp.open("in.txt",ios::out|ios::app);
                                cout<<"enter usn\n";
                                cin>>s.usn;
                                cout<<"enter name\n";
                                cin>>s.name;
                                cout<<"enter sem\n";
                                cin>>s.sem;
                                s.pack();
                                s.write_f(fp);
                                fp.close();
                                break;
                        case 2:search();
                                break;
                        case 3:exit(1);
                }
        }
        while(ch<=3);
}
void search()
{
        int c=0,choice;
        string usn;
        student s[100];
        fstream fp1;
        fp1.open("in.txt",ios::in);
        cout<<"enter the usn of the student to be searched and modified\n";
        cin>>usn;
        int cnt=0;
        int i=0;
        while(fp1)
        {
                s[i].read_f(fp1);
                s[i].unpack();
                i++;
        }
        fp1.close();
        cnt=i-1;
        for(i=0;i<cnt;i++)
        {
                if(s[i].usn==usn)
                {
                        c++;
                        break;
                }
        }
        if(c==0)
        {
                cout<<"record not found\n";
                return;
```

```
}
else
{
        cout<<"record found\n";
        s[i].print(cout);
        do
        {
                cout<<"\n\t enter your choice of field to  be modified";
                cout<<"\n\n\t usn=>\t"<<s[i].usn
    <<"\n\n\t 1.name=>\t"<<s[i].name
                <<"\n\n\t 2.semester=>\t"<<s[i].sem
            <<"\n\n\t3.exit";
                cout<<"\n\n\t choice=>";
                cin>>choice;
                switch(choice)
                {
                        case 1:cout<<"enter the name=>";
                                cin>>s[i].name;
                                break;
                        case 2:cout<<"enter the semester=>";
                                cin>>s[i].sem;
                                break;
                        case 3:break;
                        default:cout<<"\n\t\t\t invalid entry!"<<endl;
                                break;
                }
        }
        while(choice!=3);
        fp1.open("in.txt",ios::out);
        for(i=0;i<cnt;i++)
        {
                s[i].pack();
                s[i].write_f(fp1);
                }
                fp1.close();
        }
}
```

Output:

```
enter your choice
1.insert a record
2.search and modify a record
3.exit

1

enter usn
1234
enter name
chiru
enter sem
6

enter your choice
1.insert a record
2.search and modify a record
3.exit

2

enter the usn of the student to be searched and modified
1234
record found
```

```
student:
            usn '1234'
            name 'chiru'
            sem '6'

            enter your choice of field to  be modified

            usn=>  1234

            1.name=>        chiru

            2.semester=>   6

           3.exit

            choice=>1

enter the name=>chiranthan

            enter your choice of field to  be modified

            usn=>  1234

            1.name=>        chiranthan

            2.semester=>   6

           3.exit

            choice=>3
enter your choice
```

```
1.insert a record

2.search and modify a record

3.exit
```

PROGRAM 3

Write a C++ program to read and write and student objects with variable length records using any suitable record structure. Implement pack(), unpack(), modify() and search() methods.

```
#include<fstream>

#include<string>

#include<iostream>

#include<cstring>

using namespace std;

class student

{

        public:

                string usn;

                string name;

                string sem;

                string Buf;

                char buf[100];

                void pack();

                void write_f(fstream &);

                void unpack();

                void print(ostream &);

                void read_f(fstream &);

};

void student::pack()

{

Buf=usn+"|"+name+"|"+sem+"\n";

}
```

```
void student::write_f(fstream &fp)

{

    fp<<Buf;

}

void student::print(ostream &stream)

{

    stream<<"student:\n"

    <<"\t usn'"<<usn<<"'\n"

        <<"\t name'"<<name<<"'\n"

        <<"\t sem'"<<sem<<"'\n";

}

void student::unpack()

{

    char stg[100];

    int pos=0,count=0,k;

    while(count<3)

    {

        k=0;

        for(int i=pos;i<strlen(buf);i++,k++)

        {

            stg[k]=buf[i];

            pos++;

            if(buf[i]=='|')

                break;

        }


        stg[k]='\0';

        count++;

        if(count==1) usn=stg;

        if(count==2) name=stg;

        if(count==3) sem=stg;

    }

}
```

```
void student::read_f(fstream &fp)
{
        fp.getline(buf,100,'\n');
}
int main()
{
        int ch;
        fstream fp;
        void search();
        student s;
        system("clear");
        do
        {
                cout<<"enter your choice\n";
                cout<<"1.insert a record\n"
                        <<"2.search and modify a record\n"
                        <<"3.exit\n";
                cin>>ch;
                switch(ch)
                {
                        case 1:fp.open("in.txt",ios::out|ios::app);
                                cout<<"enter usn\n";
                                cin>>s.usn;
                                cout<<"enter name\n";
                                cin>>s.name;
                                cout<<"enter sem\n";
                                cin>>s.sem;
                                s.pack();
                                s.write_f(fp);
                                fp.close();
                                break;
                        case 2:search();
                                break;
```

```
                    case 3:exit(1);
                }
        }
        while(ch<=3);
}
void search()
{
        int c=0,choice;
        string usn;
        student s[100];
        fstream fp1;
        fp1.open("in.txt",ios::in);
        cout<<"enter the usn of the student to be searched and modified\n";
        cin>>usn;
        int cnt=0;
        int i=0;
        while(fp1)
        {
                s[i].read_f(fp1);
                s[i].unpack();
                i++;
        }
        fp1.close();
        cnt=i-1;
        for(i=0;i<cnt;i++)
        {
                if(s[i].usn==usn)
                {
                        c++;
                        break;
                }
        }
        if(c==0)
```

```
{
        cout<<"record not found\n";
        return;
}
else
{
        cout<<"record found\n";
        s[i].print(cout);
        do
        {
                cout<<"\n\t enter your choice of field to  be modified";
                cout<<"\n\n\t usn=>\t"<<s[i].usn
    <<"\n\n\t 1.name=>\t"<<s[i].name
                <<"\n\n\t 2.semester=>\t"<<s[i].sem
            <<"\n\n\t3.exit";
                cout<<"\n\n\t choice=>";
                cin>>choice;
                switch(choice)
                {
                        case 1:cout<<"enter the name=>";
                                cin>>s[i].name;
                                break;
                        case 2:cout<<"enter the semester=>";
                                cin>>s[i].sem;
                                break;
                        case 3:break;
                        default:cout<<"\n\t\t\t invalid entry!"<<endl;
                                break;
                }
        }
        while(choice!=3);
        fp1.open("in.txt",ios::out);
        for(i=0;i<cnt;i++)
```

```
                    {
                              s[i].pack();
                              s[i].write_f(fp1);
                    }
                    fp1.close();
          }
}
```

Output:

```
enter your choice
1.insert a record
2.search and modify a record
3.exit

1

enter usn
1234
enter name
chiru
enter sem
6

enter your choice
1.insert a record
2.search and modify a record
3.exit

2

enter the usn of the student to be searched and modified
```

```
1234

record found

student:

            usn '1234'

            name 'chiru'

            sem '6'


            enter your choice of field to  be modified


            usn=>   1234


            1.name=>        chiru


            2.semester=>    6


           3.exit


            choice=>1


enter the name=>chiranthan


            enter your choice of field to  be modified


            usn=>   1234


            1.name=>        chiranthan


            2.semester=>    6


           3.exit
```

```
            choice=>3
enter your choice
1.insert a record
2.search and modify a record
3.exit
```

PROGRAM 4

Write a c++ program to write student objects with variable-length records using any suitable record structure and to read from this file a student record using RRN.

```
#include<iostream>
#include<string>
#include<fstream>
#include<stdlib.h>
#include <cstring>

using namespace std;
char st_no[5];
int no;
class record
{
            public:
                        char usn[20];
                        char name[20];
                        char sem[2];
}
rec[20];
void retrieve_details()
```

```
{
	fstream file2;
	char name[20],usn[20],rrn[5],sem[5];
	file2.open("record.txt",ios::in);
	for(int i=0;i<no;i++)
	{
		file2.getline(rrn,5,'|');
		file2.getline(usn,20,'|');
		file2.getline(name,20,'|');
		file2.getline(sem,5,'\n');
		if(strcmp(rrn,st_no)==0)
		{
			cout<<"\n\n"<<"student details are:";
			cout<<"\n\nusn:"<<usn<<"\nname:"<<name<<"\nsem:"<<sem<<"\n";
		}
	}
	file2.close();
}
int main()
{
	fstream file1,file2;
	int ch;
	char rt_usn[20],st_rrn[20];
	char ind[2],name[20],sem[2];
	int i,flag,flag1;
	file1.open("index.txt",ios::out);
	file2.open("record.txt",ios::out);
	if(!file1||!file2)
	{
		cout<<"file creation error!\n";
		exit(0);
```

```
        }
        for(;;)
        {
                cout<<"\n1:add record"<<"\n2:search record\n";
                cout<<"enter your choice:\n";
                cin>>ch;
                switch(ch)
                {
                        case 1:cout<<"enter the no of students:";
                                cin>>no;
                                cout<<"enter the details:\n";
                                for(i=1;i<=no;i++)
                                {
                                        cout<<"\nname:";
                                        cin>>rec[i].name;
                                        cout<<"usn:";
                                        cin>>rec[i].usn;
                                        cout<<"sem:";
                                        cin>>rec[i].sem;
                                        file1<<rec[i].usn<<"|"<<i<<"\n";

        file2<<i<<"|"<<rec[i].usn<<"|"<<rec[i].name<<"|"<<rec[i].sem<<"\n";
                                }
                                file1.close();
                                file2.close();
                                break;
                        case 2:cout<<"enter rrn whose record is to be displayed:";
                                cin>>st_rrn;
                                file1.open("index.txt",ios::in);
                                if(!file1)
                                {
```

```
                    cout<<"\nerror!\n";
                    exit(0);
                }
                flag1=0;
                for(i=0;i<no;i++)
                {
                    file1.getline(rt_usn,20,'|');
                    file1.getline(st_no,4,'\n');
                    if(strcmp(st_rrn,st_no)==0)
                    {
                        retrieve_details();
                        flag1=1;
                    }
                }
                if(!flag1)
                    cout<<"record search failed!\n";
                    file1.close();
                    break;
            default : cout<<"invalid choice";
                exit(0);
                break;


        }
    }
}
```

Output:

```
1:add record
2:search record
enter your choice:
```

```
1
enter the no of students:2
enter the details:

name:chiru
usn:1234
sem:6

name:afnan
usn:1235
sem:6

1:add record
2:search record
enter your choice:
2
enter rrn whose record is to be displayed:1


student details are:

usn:1234
name:chiru
sem:6

1:add record
2:search record
enter your choice:
```

PROGRAM 5

Write a C++ program to implement simple index on primary key for a file of student objects. Implement add(), search(), delete() using the index.

```
#include<iostream>
#include<fstream>
#include<string>
using namespace std;
int n;
string usn_list[100];
int addr_list[100];
int cnt;
class student
{
        public:
                string usn,name,sem;
                void add_rec(fstream &);
                void get_data();
};
void student::get_data()
{
        cout<<"\nUSN : ";
        cin>>usn;
        cout<<"\nName : ";
        cin>>name;
        cout<<"\nSem : ";
        cin>>sem;
}
void create_index()
```

```
{
	void sort_index();
	int pos;
	string buf,urn;
	fstream fp("inp.txt",ios::in);
	cnt=-1;
	while(fp)
	{
		pos=fp.tellg();
		buf.erase();
		getline(fp,buf);
		int i=0;
		if(buf[i]=='*')
			continue;
		urn.erase();
		while(buf[i]!='|')
			urn+=buf[i++];
		usn_list[++cnt]=urn;
		addr_list[cnt]=pos;
	}
	fp.close();
	sort_index();
	for(int i=0;i<cnt;i++)
		cout<<usn_list[i]<<'|'<<addr_list[i]<<'\n';
}
void sort_index()
{
	int t_addr;
	string t_usn;
	cout<<cnt<<'\n';
	for(int i=0;i<cnt-1;i++)
```

```
        {
                for(int j=0;j<cnt-1-i;j++)
                {
                        if(usn_list[j]>usn_list[j+1])
                        {
                                t_usn=usn_list[j];
                                usn_list[j]=usn_list[j+1];
                                usn_list[j+1]=t_usn;
                                t_addr=addr_list[j];
                                addr_list[j]=addr_list[j+1];
                                addr_list[j+1]=t_addr;
                        }
                }
        }
}

void student::add_rec(fstream &fp)
{
        fp.seekp(0,ios::end);
        fp<<usn<<'|'<<name<<'|'<<sem<<"\n";
}

int search( string key)
{
  int pos=0,adr,l=0,h=cnt,mid,flag=0;
  string buffer;
  fstream fp("inp.txt",ios::in);
  while(l<=h)
  {
        mid=(l+h)/2;
        if(usn_list[mid]==key)
```

```
          {
                    flag=1;
                    break;
          }
          if(usn_list[mid]>key)
                    h=mid-1;
          if(usn_list[mid]<key)
                    l=mid+1;
 }
          if(flag)
          {
                    adr=addr_list[mid];
                    fp.seekp(adr,ios::beg);
                    getline(fp,buffer);
                    cout<<"\nFond the record "<<buffer;
                    cout<<' ' <<mid<<"mid\n";
                    return mid;
          }
          else
          {
                    cout<<"\nNot found";
                    return -1;
          }
}

void del_rec(string key)
{
          int pos,adr;
          fstream fp;
          pos=search(key);
          adr=addr_list[pos];
```

```
        if(pos !=-1)
        {
                fp.open("inp.txt",ios::out | ios::in);
                fp.seekp(adr,ios::beg);
                fp.put('*');
                cout<<"\nRecord added!";
                fp.close();
                for(int i=pos;i<cnt;i++)
                {
                        usn_list[i]=usn_list[i+1];
                        addr_list[i]=addr_list[i+1];
                }
                cnt--;

        }
        else
                cout<<"\n Record not found!";
}
int main()
{
        student s[100];
        string key;
        fstream fp;
        for(;;)
        {
        int ch;
                cout<<"\nenter ur choice \n1.add rec\n2. show index\n3.search\n4. delete\n5.
Exit\n";
                cin>>ch;
                switch(ch)
                {
                        case 1:
                                fp.open("inp.txt", ios::out);
                                cout<<"enter how many records\n";
                                cin>>n;
                                for(int i=0; i<n; i++)
                                {
                                        s[i].get_data();
                                        s[i].add_rec(fp);
                                }
                                fp.close();
                                break;

                        case 2: create_index();
                                break;

                        case 3: cout<<"enter key of record to searched\n";
                                cin>>key;
                                search(key);
                                break;

                        case 4: cout<<"enter key of record to deleted\n";
                                cin>>key;
                                del_rec(key);
                                break;

                        case 5: exit(0);
                }
        }
        return 0 ;
}
```

Output:

```
enter ur choice
1.add rec
2. show index
3.search
4. delete
5. exit
1
enter how many records
1

USN : 1234

Name : chiru

Sem : 6

enter ur choice
1.add rec
2. show index
3.search
4. delete
5. exit
2
1
1234|0

enter ur choice
1.add rec
2. show index
```

```
3.search

4. delete

5. exit

3

enter key of record to searched

0


Not found

enter ur choice

1.add rec

2. show index

3.search

4. delete

5. exit

5
```

PROGRAM 6

Write a C++ program to implement index on secondary key, the name, for a file of student objects. Implement add(),search(),delete() using the secondary index.

```
#include<string>

#include<cstring>

#include<fstream>

#include<iomanip>

#include<iostream>


using namespace std;


class record

{

   public:
    char sem[5] , usn[20] , name[20];
}rec[20] , found[20];

char st_no[5] , rt_name[20];
int no;
void sort()
{
  int i, j ;
  record temp;
  for(i = 0; i < no-1; i++)
  {
    for( j = 0; j < no-i-1; j++)
    {
      if(strcmp(rec[j].name , rec[j+1].name) > 0)
      {
        temp = rec[j];
        rec[j] = rec[j+1];
        rec[j+1] = temp;
      }
    }

  }

}

void create_index_file()
{
  ofstream index , index1;
  int i;
  index.open("secindex.txt" , ios::out);
  index1.open("record.txt" , ios::out);
  for( i = 0; i < no; i++)
  {
        if(i == no-1)
        {
            index <<rec[i].name<<"|"<<rec[i].usn<<"|"<<i+1;
            index1 <<i+1<<"|"<<rec[i].usn<<"|"<<rec[i].name<<"|"<<rec[i].sem;
        }

        else
        {
            index <<rec[i].name<<"|"<<rec[i].usn<<"|"<<i+1<<endl;
            index1 <<i+1<<"|"<<rec[i].usn<<"|"<<rec[i].name<<"|"<<rec[i].sem<<endl;
        }

    }
    index.close();
    index1.close();
}

void retrieve_record(char *index)
{
    fstream f1;
    int i;
    char buff[80],*p;
    f1.open("record.txt",ios::in);
    while(!f1.eof())
    {
        f1.getline(buff,80,'\n');
        p=strtok(buff,"|");
        if(strcmp(index, p)==0)
        {
            cout<<"\n\nStudent Details\n";
            cout<<"\nUSN\t\tName\tSemester\n";
            while(p!=NULL)
            {
```

```
            p=strtok(NULL,"|");
            if(p!=NULL)
            cout<<p<<"\t";
          }
        }

      }
      f1.close();

    }
    void delete_record(char *idx)
    {
      fstream f1;
      int i;
      char buff[80],*p,index[20][20];
      f1.open("record.txt",ios::in);
      i=0;
      while(!f1.eof())
      {
        f1.getline(buff,80,'\n');
        p=strtok(buff,"|");
        strcpy(index[i],p);
        p=strtok(NULL,"|");
        strcpy(rec[i].usn,p);
        p=strtok(NULL,"|");
        strcpy(rec[i].name,p);
        p=strtok(NULL,"|");
        strcpy(rec[i].sem,p);
        i++;

      }
      no=i;
      f1.close();
    int k=-1;
    for(i=0;i<no;i++)
    {
        if(strcmp(index[i],idx)==0)
        {
            k=i;
            break;
        }
    }
    if(k>-1)
    {
        for(i=k;i<no-1;i++)
        {
            rec[i]=rec[i+1];
        }
        no--;
        sort();
        create_index_file();
        cout<<"\nData Successfully Deleted\n";

    }
    else
    {
        cout<<"\nInvalid Name\n";
    }

}

void display_record()
{
    char buff[80] , *p;
    int flag=1;
    ifstream f1;
    f1.open("record.txt" , ios::in);
    cout<<"\n\nStudent Details\n";
    cout<<"USN\t\tName\tSemester\n";
    while(! f1.eof())
    {
        f1.getline(buff , 80 , '\n');
        p= strtok(buff, "|");
        while(p!= NULL)
        {
            flag =0;
            p= strtok(NULL , "|");
            if(p != NULL)
                cout<<p<<setw(15);
        }
        cout<<endl<<setw(0);
    }

    if(flag == 1)
        cout<<"\nNo record found";
    f1.close();
}

void retrieve_details(int ch)
{
    int k=0, i;
    char buff[80] , *p;
    ifstream f1;
    char chusn[20] , index[20][80];
    f1.open("secindex.txt" , ios::in);
    while(!f1.eof())
    {
        f1.getline(buff , 80 , '\n');
        p = strtok(buff , "|");
        if(strcmp(rt_name , p) == 0)
        {
            strcpy(found[k].name , p);
            p = strtok(NULL , "|");
            strcpy(found[k].usn , p);
            p = strtok(NULL , "|");
            strcpy(index[k] , p);
            k++;
        }
    }

    if(k == 1)
    {
        if(ch == 2)
            retrieve_record(index[0]);
        else
            delete_record(index[0]);
    }
    else if(k > 1)
    {
        cout<<"Please choose the candidate USN\n";
        for( i = 0; i < k; i++)
        {
            cout<<"Name = "<<found[i].name <<"USN = "<<found[i].usn<<endl;
        }
        cin>>chusn;
        for(i=0; i<k ; i++)
        {
            if(strcmp(chusn , found[i].usn) == 0)
            {
                if(ch == 2)
                    retrieve_record(index[i]);
                else
```

```
                    delete_record(index[i]);
            }
        }

    }
    else
        cout<<"Invalid Name\n";

}

int main()
{
    int ch, flag=1;
    while(flag)
    {
        cout<<"\n1. Add New records\n2.Retrieve Record\n3.Delete a Record\n4.Display\n5.Exit\n";
        cout<<"Enter the choice\n";
        cin>>ch;
        switch (ch)
        {
            case 1: cout<<"Enter the Number of record\t";
                    cin>>no;
                    for(int i = 0; i < no; i++)
                    {
                        cout<<"Enter the details of "<<i+1<<"th student";
                        cout<<"\nUSN\t";
                        cin>>rec[i].usn;
                        cout<<"\nName\t";
                        cin>>rec[i].name;
                        cout<<"\nSem\t";
                        cin>>rec[i].sem;
                    }
                    sort();
```

```
                create_index_file();

                break;

            case 2:

            case 3: if(ch ==2)

                  cout<<"Enter the name to search\t";

                else

                  cout<<"Enter the student name to delete\t";

                cin>>rt_name;

                retrieve_details(ch);

                break;

            case 4: display_record();

                break;


            default:

                flag =0;

                break;

        }

    }

    return 0;

}
```

Output:

```
1. Add New records

2.Retrieve Record

3.Delete a Record

4.Display

5.Exit

Enter the choice

1

Enter the Number of record      2

Enter the details of 1th student
```

```
USN     1234


Name    chiru


Sem     6

Enter the details of 2th student

USN     1235


Name    afnan


Sem     6


1. Add New records

2.Retrieve Record

3.Delete a Record

4.Display

5.Exit

Enter the choice

2

Enter the name to search          chiru



Student Details


USN               Name    Semester

1234    chiru    6

1. Add New records

2.Retrieve Record

3.Delete a Record

4.Display

5.Exit
```

```
Enter the choice

4



Student Details

USN              Name   Semester

1235       afnan          6

1234       chiru         6


1. Add New records

2.Retrieve Record

3.Delete a Record

4.Display

5.Exit

Enter the choice

5
```

PROGRAM 7

Write a C++ program to read two lists of names and then match the names in the two lists using Consequential Match based on a single loop. Output the names common to both the lists.

```
#include<iostream>

#include<cstring>

#include<fstream>

using namespace std;

int m,n;

void write()

{

fstream out1,out2;

int i;

char name[20];
```

```
out1.open("a.txt",ios::out);

out2.open("b.txt",ios::out);

cout<<"Enter no of names in file1:";

cin >> m;

cout << "Enter the names in ascending order:\n";

for(i=0;i<m;i++)

{

cin >> name;

out1 << name << "\n";

}

cout << "Enter no of names in file2:";

cin >> n;

cout << "Enter names in ascending order\n";

for(i=0;i<n;i++)

{

cin >> name;

out2 << name << "\n";

}

}

void match()

{char list1[50][50],list2[50][50];

int i,j;

fstream out1,out2,out3;

out1.open("a.txt",ios::in);

out2.open("b.txt",ios::in);

out3.open("c.txt",ios::out);

i=0;

out1.getline(list1[i],30,'\n');

cout<<"Names in file1 are:\n";

while(!out1.eof())

{

cout << list1[i] << endl;

i++;
out1.getline(list1[i],30,'\n');
}
i=0;
cout<<"Names in file2 are:\n";
out2.getline(list2[i],30,'\n');
while(!out2.eof())
{
cout << list2[i] << endl;
i++;
out2.getline(list2[i],30,'\n');
}
cout << "\nCommon names are:\n";
i = j = 0;
while(i<m && j<n)
{
if(strcmp(list1[i],list2[j]) == 0)
{
cout << list1[i] << "\n";
out3 << list1[i] << '\n';
i++;
j++;
}
else if(strcmp(list1[i],list2[j]) < 0)
i++;
}
}
int main()
{
write();
match();
return 0;
}
```

Output:

```
Enter no of names in file1:2

Enter the names in ascending order:

a b

Enter no of names in file2:2

Enter names in ascending order

a c

Names in file1 are:

a

b

Names in file2 are:

a

c


Common names are:

a
```

PROGRAM 8

Write a C++ program to read k Lists of names and merge them using K-way merge algorithm with k = 8.

```
#include <iostream>

#include <cstring>

#include <fstream>

using namespace std;

class filelist

{

char list[10][20];

int n;

public:
```

```
void merger();
void input(char filename[]);
};
char merge[80][20];
int m=0;
void filelist::merger()
{
int i,j,k;
char output[100][20];
i=0;
j=0;
k=0;
while(i<n && j<m)
{
if(strcmp(list[i],merge[j])<0 || strcmp(list[i],merge[j])==0)
strcpy(output[k++],list[i++]);
else
strcpy(output[k++],merge[j++]);
}
while(i<n)
strcpy(output[k++],list[i++]);
while(j<m)
strcpy(output[k++],merge[j++]);i=0;
while(i<k)
{
strcpy(merge[i],output[i]);
i++;
}
m=k;
}
void filelist::input(char filename[])
```

```
{
int i=0;
fstream out(filename,ios::out);
cout<<"Enter the no of names:";
cin>>n;
cout<<"Enter the names in ascending order:\n";
while(i<n)
{
cin>>list[i];
out<<list[i++];
out<<'\n';
}
out.close();
}
int main()
{
int i=0;
filelist t1;
char filename[30];
fstream file("output.txt",ios::out);
cout<<"Enter name of the first file:";
cin>>filename;
t1.input(filename);
t1.merger();
cout<<"Enter name of the second file:";
cin>>filename;
t1.input(filename);
t1.merger();
cout<<"Enter name of the third file:";
cin>>filename;t1.input(filename);
t1.merger();
cout<<"Enter name of the fourth file:";

cin>>filename;

t1.input(filename);

t1.merger();

cout<<"Enter name of the fifth file:";

cin>>filename;

t1.input(filename);

t1.merger();

cout<<"Enter name of the sixth file:";

cin>>filename;

t1.input(filename);

t1.merger();

cout<<"Enter name of the seventh file:";

cin>>filename;

t1.input(filename);

t1.merger();

cout<<"Enter name of the eigth file:";

cin>>filename;

t1.input(filename);

t1.merger();

cout<<"Merged output:"<<endl;

while(i<m)

{

file<<merge[i];

cout<<merge[i]<<endl;

file<<'\n';

i++;

}

file.close();

}
```

Output:

```
Enter name of the first file:1

Enter the no of names:1

Enter the names in ascending order:

a

Enter name of the second file:2

Enter the no of names:1

Enter the names in ascending order:

2

Enter name of the third file:3

Enter the no of names:1

Enter the names in ascending order:

3

Enter name of the fourth file:4

Enter the no of names:1

Enter the names in ascending order:

d

Enter name of the fifth file:5

Enter the no of names:1

Enter the names in ascending order:

e

Enter name of the sixth file:6

Enter the no of names:1

Enter the names in ascending order:

f

Enter name of the seventh file:7

Enter the no of names:1

Enter the names in ascending order:

g

Enter name of the eigth file:8
Enter the no of names:1

Enter the names in ascending order:

h

Merged output:

2

3

a

d

e

f

g

h
```