Sapthagiri College of Engineering

(Affiliated to VTU, Belgaum & Approved by AICTE- New Delhi)

#14/5 Chikkasandra, Hesaraghatta Main Road, Bengaluru-560057

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

FILE STRUCTURES LABORATORY WITH MINI PROJECT

18ISL67

VI SEMESTER

Prepared by

Prof. CHAITHRA B M

Assistant Professor

Dr. ASHA P N

Associate Professor

ACADEMIC YEAR 2020-2021

INDEX

SL. NO	PROGRAMS PART A	PAGE NO.
1	Write a C++ program to read series of names, one per line, from standard input and write these names spelled in reverse order to the standard output using I/O redirection and pipes. Repeat the exercise using an input file specified by the user instead of the standard input and using an output file specified by the user instead of the standard output.	3-5
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.	6-11
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.	12-17
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.	18-21
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.	22-28
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.	29-36
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.	37-39
8	Write a C++ program to read k Lists of names and merge them using kway merge algorithm with k = 8.	40-43
	PART B Student should develop mini project on the topics mentioned below or similar applications Document processing, transaction management, indexing and hashing, buffer management, configuration management. Not limited to these.	

1. Write a C++ program to read series of names, one per line, from standard input and write these names spelled in reverse order to the standard output using I/O redirection and pipes. Repeat the exercise using an input file specified by the user instead of the standard input and using an output file specified by the user instead of the standard output.

```
#include<iostream>
#include<fstream>
#include<string>
using namespace std;
int main()
{
 char buf[20][20],infile[20],outfile[20],buf1[20];
 char line[200];
  int i,j,indx,x,y,ch;
  fstream file,file1;

  cout<<"Enter your choice\n1->file I/O\n2->standard I/O";
  cin>>ch;
  switch(ch)
  {
    case 1:{ cout<<"Enter the input file:"<<flush;
         cin>>infile;
         cout<<"Enter the output file:"<<flush;
         cin>>outfile;
         file.open(infile,ios::in);
         file.unsetf(ios::skipws);
         file1.open(outfile,ios::out);
         while(1)
         {
               file.getline(line,200,'\n');
               if(file.fail())
                        break;
               else
               {
                        for(i=strlen(line);i>=0;i--)
                                 file1<<line[i];
                        file1<<"\n";
               } }
         file.close();
         file1.close();
          }
       break;
    case 2:{ cout<<"Enter the sequence of names or '0' to exit";
          indx=0;
          while(1)
                 {
```

```
            cin>>buf[indx];
            if(strcmp(buf[indx],"0")==0) break;
            indx++;
          }
          for(i=0;i<indx;i++)
          {
            for(j=strlen(buf[i]);j>=0;j--)
            cout<<buf[i][j];
            cout<<endl;
          }
         }
         break;
   }
return 0;
}
```

OUTPUT

```
[root@localhost ~]# gedit program1.cpp
 [root@localhost ~]# gedit program1.cpp
 [root@localhost ~]# g++ program1.cpp
 [root@localhost ~]# ./a.out
 Enter your choice
 1->file I/O
 2->standard I/O
 1
 Enter the input file:inputfile.txt
Enter the output file:outputfile.txt
[root@localhost ~]# cat outputfile.txt
mar
 mahs

 [root@localhost ~]# ./a.outEnter your choice
 1->file I/O
 2->standard I/O
 2
 Enter the sequence of names or '0' to exit
 ram
 sham
 0
 mar
 mahs
```

1. Using Redirection

```
[root@localhost ~]# ./a.out program1>bb.txt
1
inputfile.txt
outputfile.txt
[root@localhost ~]# cat outputfile.txt
mar
```

```
mahs
[root@localhost ~]# ./a.out program1>bb.txt
2
ram
sham
0
[root@localhost ~]# cat bb.txt
Enter your choice
1->file I/O
2->standard I/OEnter the sequence of names or '0' to exitmar
Mahs
```

2. Using Pipes

```
[root@localhost ~]# ./a.out program1.cpp|tee bb.txt
Enter your choice
1->file I/O
2->standard I/O
1
Enter the input file:inputfile.txt
Enter the output file:outputfile.txt
[root@localhost ~]# cat outputfile.txt
mar
mahs
[root@localhost ~]# ./a.out program1.cpp|tee bb.txt
Enter your choice
1->file I/O
2->standard I/O
2
Enter the sequence of names or '0' to exit
ram
sham
0
mar
mahs

[root@localhost ~]#
```

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<iostream>
#include<fstream>
#include<string>
#include<sstream>
#include<stdio.h>
#include<stdlib.h>
using namespace std;
class student
{
      public:
            string usn;
            string name;
            string branch;
            string sem;
            string buffer;
            void read_data();
            void pack();
            void write_to_file();
            void unpack();
            int search(string);
            int delete_from_file(string);
            void modify(string);
};
void student::read_data()
{
    cout<<"usn:";
    cin>>usn;
    cout<<"name:";
    cin>>name;
    cout<<"branch:";
    cin>>branch;
    cout<<"semester:";
    cin>>sem;
}
void student::pack()
{
      string temp;
      buffer.erase();
      buffer+=usn+"|"+name+"|"+branch+"|"+sem+"$";
      for(;buffer.size()<100;)
      buffer+='$';
      buffer+='\n';
```

```
}
void student::write_to_file()
{
	fstream file;
	file.open("2.txt",ios::out|ios::app);
	file<<buffer;
	file.close();
}
void student:: unpack()
{
	int ch=1,i=0;
	usn.erase();
	while(buffer[i]!='|')
	usn+=buffer[i++];
	name.erase();
	i++;
	while(buffer[i]!='|')
	name+=buffer[i++];
	branch.erase();
	i++;
	while(buffer[i]!='|')
	branch+=buffer[i++];
	sem.erase();
	i++;
	while(buffer[i]!='$')
	sem+=buffer[i++];
}
int student::search(string key)
{
	ifstream file;
	int flag=0,pos=0;
	file.open("2.txt",ios::in);
	while(!file.eof())
	{
		buffer.erase();
		pos=file.tellg();
		getline(file,buffer);
		unpack();
		if(key==usn)
		{
			cout<<"\nfound the key .the record is...\n"<<buffer;
			flag=1;
			return pos;
}	  }
	file.close();
	if(flag==0)
	{
		cout<<"\nnot found..\n";
```

```
            return -1;
        }
}
int student::delete_from_file(string key)
{
        fstream  file;
        int pos, flag=0;
        pos=search(key);
        if(pos>=0)
        {
                file.open("2.txt");
                file.seekp(pos,ios::beg);
                file.put('*');
                flag=1;
                file.close();
        }
        if(flag==1) return 1;
        else
        return 0;
}
void student::modify(string key)
{
        int c;
        if(delete_from_file(key))
        {
                cout<<"\nwhat to modify\n1:usn 2:name 3:branch 4:semester\n";
                cin>>c;
                switch(c)
                {
                        case 1:cout<<"usn:\n";
                                cin>>usn;
                                break;
                        case 2:cout<<"name:\n";
                                cin>>name;
                                break;
                        case 3:cout<<"branch:\n";
                                cin>>branch;
                                break;
                        case 4:cout<<"semester:\n";
                                cin>>sem;
                               break;
                      default:cout<<"wrong choice\n";
                }
                pack();
                write_to_file();
       }
}
int main()
```

```
{
    int count,choice,i;
    student s1;
    string key;
    system("clear");
    while(1)
   {
        cout<<"\nmain menu\n1.add\n2.delete\n3.modify\n4.search\n5.exit ";
        cin>>choice;
        switch(choice)
        {
            case 1:cout<<"\nhow many records to insert\n";
                cin>>count;
                for(i=0;i<count;i++)
                {
                    cout<<"data\n";
                    s1.read_data();
                    s1.pack();
                    s1.write_to_file();
                }
                break;
            case 2: cout<<"\nenter the key...\n";
                cin>>key;
                i=s1.delete_from_file(key);
        if(i==1)
                    cout<<"record deleted\n";
        else
                    cout<<"record not deleted\n";
        break;
            case 3:cout<<"enter the key\n";
                cin>>key;
                s1.modify(key);
                break;
            case 4:cout<<"enter the key\n";
                cin>>key;
                i=s1.search(key);
                break;
            case 5:return 0;
            default: cout<<"wrong choice ...";
        }
  }
  return 0;
}
```

OUTPUT

```
[root@localhost ~]# g++ 2.cpp
[root@localhost ~]# ./a.out
main menu
1.add
2.delete
3.modify
4.search
5.exit 1
how many records to insert
3
data
usn:is001
name:Abdul
branch:ise
semester:6
data
usn:is002
name:Afroz
branch:cse
semester:7
data
usn:is005
name:Akshatha
branch:cv
semester:5
main menu
1.add
2.delete
3.modify
4.search
5.exit 1
how many records to insert
1
data
usn:is008
name:Ankit
branch:ec
semester:8
main menu
1.add
2.delete
3.modify
4.search
5.exit 2
enter the key...
```

```
is003
found the key .the record is...
is003|Aishwarya|me|7$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$$record deleted
main menu
1.add
2.delete
3.modify
4.search
5.exit  3
enter the key
is002
found the key .the record is...
is002|Afroz|cse|5$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$
what to modify
1:usn 2:name 3:branch 4:semester
2
name:
xxxx
main menu
1.add
2.delete
3.modify
4.search
5.exit  4
enter the key
is002
found the key .the record is...
is002|xxxx|cse|5$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$$$$$$$$$$
main menu
1.add
2.delete
3.modify
4.search
5.exit 6
wrong choice....
main menu
1.add
2.delete
3.modify
4.search
5.exit 5
[root@localhost ~]
```

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<iostream>
#include<fstream>
#include<string>
#include<sstream>
#include<stdio.h>
#include<stdlib.h>
using namespace std;
class student
{
      public:
            string usn;
            string name;
            string branch;
            string sem;
            string buffer;
            void er(){buffer.erase();}
            void read_data();
            void pack();
            void write_to_file();
            void unpack();
            int search(string);
            int delete_from_file(string);
            void modify(string);
};
void student::read_data()
{
   cout<<"usn:";
   cin>>usn;
   cout<<"name:";
   cin>>name;
   cout<<"branch:";
   cin>>branch;
   cout<<"semester:";
   cin>>sem;
}
void student::pack()
{
      buffer.erase();
      buffer+=usn+"|"+name+"|"+branch+"|"+sem+"$"+"\n";
}
void student::write_to_file()
```

```
{
        fstream file;
        file.open("3.txt",ios::out|ios::app);
        file<<buffer;
        file.close();
}
void student:: unpack()
{
        int ch=1,i=0;
        usn.erase();
        while(buffer[i]!='|')
        usn+=buffer[i++];
        name.erase();
        i++;
        while(buffer[i]!='|')
        name+=buffer[i++];
        branch.erase();
        i++;
        while(buffer[i]!='|')
        branch+=buffer[i++];
        sem.erase();
        i++;
        while(buffer[i]!='$')
        sem+=buffer[i++];
}
int student::search(string key)
{
        ifstream file;
        int flag=0,pos=0;
        file.open("3.txt",ios::in);
        while(!file.eof())
        {
                buffer.erase();
                pos=file.tellg();
                getline(file,buffer);
                unpack();
                if(key==usn)
                {
                        cout<<"\nfound the key .the record is...\n"<<buffer;
                        return pos;
                }
        }
        file.close();
        if(flag==0)
        {
                cout<<"\nnot found..\n";
                return -1;
```

```
}
}
int student::delete_from_file(string key)
{
	fstream  file;
	int pos, flag=0;
	pos=search(key);
	if(pos>=0)
	{
		file.open("3.txt");
		file.seekp(pos,ios::beg);
		file.put('*');
		flag=1;
		file.close();
	}
	if(flag==1) return 1;
	else
	return 0;
}
void student::modify(string key)
{
	int c;
	if(delete_from_file(key))
	{
		cout<<"\nwhat to modify\n1:usn 2:name 3:branch 4:semester\n";
		cin>>c;
		switch(c)
		{
			case 1:cout<<"usn:\n";
				cin>>usn;
				break;
			case 2:cout<<"name:\n";
				cin>>name;
				break;
			case 3:cout<<"branch:\n";
				cin>>branch;
				break;
			case 4:cout<<"semester:\n";
				cin>>sem;
				break;
			default:cout<<"wrong choice\n";
		}
		buffer.erase();
		pack();
		write_to_file();
}
}
int main()
```

```
{
      int count,choice,i;
      student s1;
      string key;
     system("clear");
     while(1)
    {
          cout<<"\nmain menu\n1.add\n2.delete\n3.modify\n4.search\n5.exit ";
          cin>>choice;
          switch(choice)
          {
               case 1:cout<<"\nhow many records to insert\n";
                    cin>>count; s1.er();
                    for(i=0;i<count;i++)
                    {
                         cout<<"data\n";
                         s1.read_data();
                         s1.pack();
                         s1.write_to_file();
                    }
                    break;
               case 2: cout<<"\nenter the key...\n";
                    cin>>key;
                    i=s1.delete_from_file(key);
          if(i==1)
                         cout<<"record deleted\n";
          else
                         cout<<"record not deleted\n";
          break;
               case 3:cout<<"enter the key\n";
                    cin>>key;
                    s1.modify(key);
                    break;
               case 4:cout<<"enter the key\n";
                    cin>>key;
                    i=s1.search(key);
                    break;
               case 5:return 0;
               default:cout<<"wrong choice ...";
          }
     }
     return 0;
}
```

OUTPUT

```
[root@localhost ~]# g++ 3.cpp
[root@localhost ~]# ./a.out
main menu
1.add
2.delete
3.modify
4.search
5.exit 1
how many records to insert
3
data
usn:is009
name:Anusha
branch:is
semester:7
data
usn:cs011
name:Anvith
branch:cs
semester:6
data
usn:me012
name:aqil
branch:me
semester:8
main menu
1.add
2.delete
3.modify
4.search
5.exit 1
how many records to insert
1
data
usn:cv014
name:Arpitha
branch:cv
semester:5
main menu
1.add
2.delete
3.modify4.search
5.exit 2
enter the key...
cv014
found the key .the record is...
```

```
cv014|Arpitha|cv|5$record deleted
main menu
1.add
2.delete
3.modify
4.search
5.exit  3
enter the key
me012
found the key .the record is...
me012|aqil|me|8$
what to modify
1:usn 2:name 3:branch 4:semester
4
semester:
6
main menu
1.add
2.delete
3.modify
4.search
5.exit  4
enter the key
me012
found the key .the record is...
me012|aqil|me|6$
main menu
1.add
2.delete
3.modify
4.search
5.exit 6
wrong choice....
main menu
1.add
2.delete
3.modify
4.search
5.exit 5
[root@localhost ~]#
```

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<fstream>
#include<string>
#include<sstream>
using namespace std;
class student
{
  public:
      string usn;
      string name;
      string branch;
      string semester;
      string buffer;
      int count;
      int  rrn_list[100];
      void read_data();
      void pack();
      void write_to_file();
      void create_rrn();
      void search_by_rrn(int);
};
void student::read_data()
{
      cout<<"usn:";
      cin>>usn;
      cout<<"name:";
      cin>>name;
      cout<<"branch:";
      cin>>branch;
      cout<<"semester:";
      cin>>semester;
}
void student::pack()
{
      buffer.erase();
      buffer=usn+'|'+name+'|'+branch+'|'+semester+'$'+'\n';
}
void student::write_to_file()
{
      fstream file;
      file.open("4.txt",ios::out|ios::app);
      file<<buffer;
      file.close();
}
```

```
void student::create_rrn()
{
	ifstream file;
	int pos;
	count=-1;
	file.open("4.txt",ios::in);
	while(!file.eof())
	{
		pos=file.tellg();
		buffer.erase();
		getline(file,buffer);
		rrn_list[++count]=pos;
	}
	file.close();
}
void student::search_by_rrn(int rrn)
{
	int pos;
	fstream file;
	create_rrn();
	if(rrn>=count)
	cout<<"\nnot found";
	else
	{
		buffer.erase();
		file.open("4.txt");
		pos=rrn_list[rrn];
		file.seekg(pos,ios::beg);
		getline(file,buffer);
		cout<<"\n"<<buffer<<"\n";
		file.close();
	}
}
int main()
{
	int choice,rrn,count,i;
	student s1;
	while(1)
	{
		cout<<"\nmain menu\n1.add 2.search 3.exit\nenter the choice:";
		cin>>choice;
		switch(choice)
		{
			case 1:cout<<"\nhow many records to insert\n";
				cin>>count;
				for(i=0;i<count;i++)
				{
					cout<<"data\n";
```

```
                        s1.read_data();
                        s1.pack();
                s1.write_to_file();
            }
        s1.create_rrn();
            break;
        case 2:cout<<"enter the rrn";
            cin>>rrn;
            s1.search_by_rrn(rrn);
            break;
        case 3:return 0;
        default:cout<<"\nwrong choice";
            break;
      }
    }
}
```

OUTPUT

```
[root@localhost ~]# g++ 444.cpp
[root@localhost ~]#  ./a.out
main menu
1.add 2.search 3.exit
enter the choice:1
how many records to insert
3
data
usn:is016
name:Ashwithc
branch:is
semester:6
data
usn:cv017
name:Akshithk
branch:cv
semester:8
data
usn:me018
name:Chaitra
branch:me
semester:5
main menu
1.add 2.search 3.exit
enter the choice:2
enter the rrn 0
is016|Ashwithc|is|6$
```

```
main menu
1.add 2.search 3.exit
enter the choice:2
enter the rrn 1
cv017|Akshithk|cv|8$
main menu
1.add 2.search 3.exit
enter the choice:2
enter the rrn 3
not found
main menu
1.add 2.search 3.exit
enter the choice:2
enter the rrn 2
me018|Chaitra|me|5$
main menu
1.add 2.search 3.exit
enter the choice:3
[exam@localhost ~]$
```

5. Write a 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>
#include<sstream>
#include<stdlib.h>
using namespace std;
class student
{
  public:
        string usn;
        string name;
        string branch;
        string sem;
        string buffer;
    string usn_list[100];
    int Address_list[100];
    int count;
    void read_data();
    void pack();
    void  write_to_file();
    void  create_index();
    void remove(string);
    void search(string);
    int search_index(string);
    string extract_usn();
    void sort_index();
};
void student::read_data()
{
    cout<<"usn:";
    cin>>usn;
    cout<<"name:";
    cin>>name;
    cout<<"branch:";
    cin>>branch;
    cout<<"semester:";
    cin>>sem;
}
void student::pack()
{
        string temp;
        buffer.erase();
        buffer+=usn+"|"+name+"|"+branch+"|"+sem+"$"+"\n";
}
```

```
void student::write_to_file()
{
        int pos;
        fstream file;
        file.open("5.txt",ios::out|ios::app);
        pos=file.tellp();
        file<<buffer;
        file.close();
        usn_list[++count]=usn;
        Address_list[count]=pos;
        sort_index();
}
string student::extract_usn()
{
  string usn;
  int i=0;
  usn.erase();
   while(buffer[i]!='|')
   usn+=buffer[i++];
   return usn;
}
void student::create_index()
{
  fstream file;
  int pos;
  string usn;
  count=-1;
  file.open("5.txt",ios::in);
  while(!file.eof())
  {
     pos=file.tellg();
     buffer.erase();
     getline(file,buffer);
     if(buffer[0]!='*')
                {
                  if(buffer.empty())break;
                  usn=extract_usn();
                  usn_list[++count]=usn;
                  Address_list[count]=pos;
                }
   }
   file.close();
   sort_index();
   buffer.erase();
}
void student::sort_index()
{
      int i,j,temp_Address;
```

```
        string temp_usn;
        for(int i=0;i<=count;i++)
        {
                for(int j=i+1;j<=count;j++)
                {
                        if(usn_list[i]>usn_list[j])
                        {
                                temp_usn=usn_list[i];
                                usn_list[i]=usn_list[j];
                                usn_list[j]=temp_usn;
                                temp_Address=Address_list[i];
                                Address_list[i]=Address_list[j];
                                Address_list[j]=temp_Address;
                        }
                }
        }
          for(i=0;i<=count;i++)
        {
                cout<<usn_list[i]<<"\t"<<Address_list[i]<<"\n";
        }
}
int student::search_index(string key)
{
         int low=0,high=count,mid=0,flag=0,pos;
          while(low<=high)
        {
                mid=(low+high)/2;
                if(usn_list[mid]==key){flag=1;break;}
                if(usn_list[mid]>key)high=mid-1;
                if(usn_list[mid]<key)low=mid+1;
         }
        if(flag)
        return mid;
        else
        return -1 ;
}
void student::search(string key)
{
        int pos=0,address;
        fstream file;
        buffer.erase();
        pos=search_index(key);
        if(pos==-1)
        cout << endl << "record not found" << endl;
        else if(pos>=0)
        {
         file.open("5.txt");
        address=Address_list[pos];
```

```
        file.seekp(address,ios::beg);
        getline(file,buffer);
        cout<<"record found... \n"<<buffer;
        file.close();
    }
}
void student::remove(string key)
{
        int pos=0,i,address;
        fstream file;
        pos=search_index(key);
        if(pos>=0)
        {
                file.open("5.txt",ios::out|ios::in);
                address=Address_list[pos];
                file.seekp(address,ios::beg);
                file.put('*');
                file.close();
                cout<<"\nRecord Deleted: ";
                for(i=pos;i<count;i++)
                {
                        usn_list[i]=usn_list[i+1];
                        Address_list[i]=Address_list[i+1];
      }
      count--;
    }
    else
      cout<<"record not found\n";
}
int main()
{
        int choice,count,i;
        string key;
        student s1;
        s1.create_index();
        while(1)
        {
                cout<<"\nMain Menu\n--------\n1.Add \n2.Search \n3.Delete\n4.Exit\n -------- \n";
                cout<<"Enter the choice:";
                cin>>choice;
                switch(choice)
                {
          case 1:
          cout<<"\nhow many records to insert\n";
          cin>>count;
          for(i=0;i<count;i++)
          {
                cout<<"data\n";
```

```
                s1.read_data();
                s1.pack();
                s1.write_to_file();
            }
             break;
             case 2:
            system("clear");
            cout<<"\nEnter the usn\n";
    cin>>key;
    s1.search(key);
    break;
           case 3:cout<<"\n\nEnter the usn\n";
    cin>>key;
    s1.remove(key);
    break;
           case 4:return 0;
           default:cout<<"\n\nWrong choice\n";
           break;
    }
  }
}
```

OUTPUT

```
[root@localhost ~]# g++ 5.cpp
[root@localhost ~]# ./a.out
Main Menu
- - - - -
1.Add
2.Search
3.Delete
4.Exit
- - - - - -
Enter the choice:1
how many records to insert
4
data
usn:cs021
name:Chandni
branch:cs
semester:8

cs021 0
data
usn:is021
name:Chandni
branch:is
```

```
semester:5
cs021 0
is021 20
data
usn:cv022
name:Charan
branch:cv
semester:4
cs021 0
cv022 40
is021 20
data
usn:me023
name:Chetana
branch:me
semester:7
cs021 0
cv022 40
is021 20
me023 59
Main Menu
------
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:2
Enter the usn
is021
record found....
is021|Chandni|is|5$
Main Menu
------
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:3
Enter the usn
cs021
Record Deleted:
Main Menu
------
1.Add
2.Search
3.Delete
```

4.Exit

- - - - - -

Enter the choice:4

[root@localhost ~]#

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

```
#include<iostream>
#include<fstream>
#include<string>
#include<sstream>
#include<stdlib.h>
using namespace std;
class student
{
  public:
        string usn;
        string name;
        string branch;
        string sem;
        string buffer;
       string Name_list[100];
        int  Address_list[100];
        int count;
         student(){ count=-1;}
         void read_data();
         void pack();
         void write_to_file();
void create_index();
    string extract_name();
          void disp();
         void remove(string);
         void delete_from_file(int);
         void search(string);
         int  search_index(string);
         void read_from_file(int);
         void sort_index();
};
void student::read_data()
{
    cout<<"usn:";
    cin>>usn;
    cout<<"name:";
    cin>>name;
    cout<<"branch:";
    cin>>branch;
    cout<<"semester:";
    cin>>sem;
```

```
}
void student::pack()
{
        buffer.erase();
        buffer+=usn+"|"+name+"|"+branch+"|"+sem+"$"+"\n";
}
void student::write_to_file()
{
       int pos;
       fstream file;
       file.open("6a.txt",ios::out|ios::app);
       pos=file.tellp();
       file<<buffer;
       file.close();
       Name_list[++count]=name;
       Address_list[count]=pos;
       sort_index();
}
string student::extract_name()
{
  string usn;
  int i=0;
  usn.erase();
   while(buffer[i]!='|')
   usn+=buffer[i++];
i++;
name.erase();
   while(buffer[i]!='|')
   name+=buffer[i++];

   return name;
}
void student::create_index()
{
  fstream file;
  int pos;
  string usn;
  count=-1;
  file.open("5.txt",ios::in);
  while(!file.eof())
  {
     pos=file.tellg();
     buffer.erase();
     getline(file,buffer);
     if(buffer[0]!='*')
                {
                  if(buffer.empty())break;
                  name=extract_name();
```

```
                name_list[++count]=name;
                Address_list[count]=pos;
            }
    }
    file.close();
    sort_index();
    buffer.erase();
}


void student::disp()
{
        int i;
        cout << endl << "INDEX FILE " << endl;
        for(i=0;i<=count;i++)
        cout<<endl<<Name_list[i]<<" "<<Address_list[i];
        cout<<"\n";
        system("cat 6a.txt");
}
void student::sort_index()
{
         int i,j,temp_Address;
        string temp_Name;
        for(int i=0;i<=count;i++)
        {
                for(int j=i+1;j<=count;j++)
                {
                        if(Name_list[i]>Name_list[j])
                        {
                                temp_Name=Name_list[i];
                                Name_list[i]=Name_list[j];
                                Name_list[j]=temp_Name;
                                temp_Address=Address_list[i];
                                Address_list[i]=Address_list[j];
                                Address_list[j]=temp_Address;
                        }
                }
        }
}
int student::search_index(string key)
{
         int low=0,high=count,mid=0,flag=0,pos;
        while(low<=high)
        {
                mid=(low+high)/2;
                if(Name_list[mid]==key){flag=1;break;}
                if(Name_list[mid]>key)high=mid-1;
                if(Name_list[mid]<key)low=mid+1;
```

```
        }
         if(flag)
        return mid;
        else
        return -1 ;
}
void student::search(string key)
{
        int pos=0,t;
         string buffer;
         buffer.erase();
        pos=search_index(key);
        if(pos==-1)
        cout << endl << "record not found" << endl;
        else if(pos>=0)
        {
                read_from_file(pos);
                 t=pos;
                while(Name_list[++t]==key && t<=count) read_from_file(t);
                 t=pos;
                while(Name_list[--t]==key && t>=0) read_from_file(t);
        }
}
void student::read_from_file(int pos)
{
        int address,i;
         fstream file;
        file.open("6a.txt",ios::in);
        address=Address_list[pos];
        file.seekp(address,ios::beg);
        buffer.erase();
        getline(file,buffer);
        cout<<"\nFound the record: "<<buffer;
        file.close();
}
void student::remove(string key)
{
        int  pos=0,t,choice;
        string buffer;
        buffer.erase();
        pos=search_index(key);
        if(pos==-1)
        cout << endl << "not possible to remove";
        else if(pos>=0)
        {
                 read_from_file(pos);
                cout<<"\nDelete?(1/0):";
                cin>>choice;
```

```
            if(choice)delete_from_file(pos);
            t=pos;
            while(Name_list[++t]==key)
            {
                  read_from_file(t);
                 cout<<"\nDelete?";
                 cin>>choice;
                 if(choice)delete_from_file(t);
            }
            t=pos;
            while(Name_list[--t]==key )
            {
                 read_from_file(t);
                 cout<<"\nDelete?";
                 cin>>choice;
                 if(choice)
                 delete_from_file(t);
            }
      }
}
void student::delete_from_file(int pos)
{
       int i,address;
      fstream file;
      file.open("6a.txt");
      address=Address_list[pos];
      file.seekp(address,ios::beg);
      file.put('*');
      cout<<"\nRecord Deleted: ";
      for(i=pos;i<count;i++)
      {
            Name_list[i]=Name_list[i+1];
            Address_list[i]=Address_list[i+1];
      }
      count--;
}
int main()
{
      int choice,count,i;
      string key;
      student s1;
s1.create_index()

      while(1)
      {
            cout<<"\nMain Menu\n--------\n1.Add \n2.Search \n3.Delete\n4.Exit\n --------\n";
            cout<<"Enter the choice:";
             cin>>choice;
```

```
            switch(choice)
            {
                case 1: cout<<"\nhow many records to insert\n";
                cin>>count;
                for(i=0;i<count;i++)
                {
                    cout<<"data\n";
                    s1.read_data();
                    s1.pack();
                    s1.write_to_file();
                }
                break;
                case 2: system("clear");
                    s1.disp();
                    cout<<"\nEnter the name\n";
                cin>>key;
                s1.search(key);
                break;
                case 3: cout<<"\n\nEnter the name\n";
                cin>>key;
                s1.remove(key);
                break;
                case 4: return 0;
                default:cout<<"\n\nWrong choice\n";
                break;
            }
        }
    }
```

OUTPUT

```
[root@localhost ~]#  g++ 666.cpp
[root@localhost ~]#  ./a.out
Main Menu
------
1.Add
 2.Search
3.Delete 4.Exit ---------
Enter the choice:1
how many records to insert 3
data
 usn:is060
name:Abhishree
branch:6
semester:is
 data
usn:ec017
name:kajal
branch:ec
semester:6
data usn:is016
name:jeevan
branch:is
semester:5
Main Menu
------
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:2
INDEX FILE
Abhishree 0
jeevan 40
 kajal 22
is060|Abhishree|6|is$
ec017|kajal|ec|6$
is016|jeevan|is|5$
Enter the name
jeevan
Found the record:
is016|jeevan|is|5$
Main Menu
------
```

```
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:3
Enter the name
Kajal
Found the record: ec017|kajal|ec|6$
Delete?(1/0):1
Record Deleted:
Main Menu
------
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:3
Enter the name
kajal
not possible to remove
Main Menu
------
1.Add
2.Search
3.Delete
4.Exit
------
Enter the choice:4
[root@localhost ~]#
```

7. Write a 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<fstream>
#include<string>
using namespace std;
class coseq
{
 private:
    string list1[100],list2[100];
    int count1,count2;
 public:
   void load_list();
    void sort_list();
    void match();
};
void coseq::load_list()
{
        fstream file;
        string name;
        count1=-1;count2=-1;
        file.open("name1.txt");
        while(!file.eof())
        {
          name.erase();
          getline(file,name);
          list1[++count1]=name;
       }
       file.close();
       file.open("name2.txt");
       while(!file.eof())
       {
              name.erase();
              getline(file,name);
              list2[++count2]=name;
       }
       file.close();
}
void coseq::sort_list()
{
      int  i,j;
      string temp;
      for(i=0;i<count1;i++)
      {
              for(j=i+1;j<count1;j++)
```

```
            {
                if(list1[i]>list1[j])
                {
                    temp=list1[i];
                list1[i]=list1[j];
                list1[j]=temp;
                }
            }
        }
         for(i=0;i<count2;i++)
         {
            for(j=i+1;j<count2;j++)
            {
                if(list2[i]>list2[j])
                {
                temp=list2[i];
                    list2[i]=list2[j];
                    list2[j]=temp;
                }
            }
        }
}
void coseq::match()
{
         int i=0,j=0,flag=0;
         while(i<count1 && j<count2)
        {
        if(list1[i]==list2[j])
        {
                    cout<<list1[i]<<"\n";
                     i++;
                    j++;
                    flag=1;
             }
             if(list1[i]<list2[j])
        i++;
                if(list1[i]>list2[j])
                j++;
    }
    if(flag==0) cout<<"no match found\n";
}
int main()
{
        coseq c1;
       c1.load_list();
       c1.sort_list();
       c1.match();
       return 0;
```

```
}
```

OUTPUT

```
[root@localhost ~]#  gedit name1.txt
Gousekhan
Hithashree
Divya
[root@localhost ~]# gedit name2.txt
Jayashree
Aishwarysr
Divya
[root@localhost ~]# g++ 7.cpp
[root@localhost ~]# ./a.out
Divya
[root@localhost ~]# ./a.out
no match found
[root@localhost ~]#
```

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

```
#include<string.h>
#include<fstream>
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
using namespace std;
class filelist
{
        char list[10][20];
        int n;
        public:
        void merger();
        void input(char filename[]);
};
char merge1[80][20];
int m=0;
void filelist::merger()
{
       cout<<m;
       int i,j,k;
        char output[100][20];
        i=0;j=0;k=0;
        while(i<n && j<m)
        {
                if(strcmp(list[i],merge1[j])<0 || strcmp(list[i],merge1[j])==0)
               strcpy(output[k++],list[i++]);
                else
                    strcpy(output[k++],merge1[j++]);
        }
        while(i<n)
        strcpy(output[k++],list[i++]);
        while(j<m)
        strcpy(output[k++],merge1[j++]);
        i=0;
        while(i<k)
        {
                strcpy(merge1[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 \n";
        while(i<n)
        {
                cin>>list[i];
                out<<list[i++];
                out<<'\n';
        }
        out.close();
        int j,k; char temp[20];
        for(j=0;j<n;j++)
        {
           for(k=j+1;k<n;k++)
             {
                if ( strcmp( list[j] , list[k]) >0 )
                {
                    strcpy(temp,list[k]);
                    strcpy(list[k],list[j]);
                    strcpy(list[j],temp);
                }
             }
         }
}

int main()
{
        int i=0,files;
        filelist t1;
        char filename[30];
        fstream file("output.txt",ios::out);
        cout << endl << "enter how many files" << endl;
        cin >> files;
        int j;
       for(j=0;j<files;j++)
        {
                cout<<"Enter name of "<<j+1<<" file: ";
                cin>>filename;
                t1.input(filename);
                t1.merger();
        }
       cout << endl << "cosequential merging is" << endl;
        while(i<m)
        {
                 file<<merge1[i];
                 file<<'\n';
```

```
                i++;
        }
    file.close();
    system("cat output.txt");
return 0;
}
```

OUTPUT

```
[root@localhost ~]# g++ 8.cpp
[root@localhost ~]# ./a.out
enter how many files
2
Enter name of 1 file: abc.txt
Enter the no of names: 8
Enter the names
Karthik
Lavanya
Lenson
Likhith
Lipitha
Vrinda
Medhini
Megha
0Enter name of 2 file: xyz.txt
Enter the no of names: 8
Enter the names
Meghana
Minolita
Mona
Naval
Navya
Neha
Nikhitha
Nirosha
8
cosequential merging is
Karthik
Lavanya
Lenson
Likhith'
Lipitha
Medhini
Megha
Meghana
Minolita
```

```
Mona
Naval
Navya
Neha
Nikhitha
Nirosha
Vrinda
[root@localhost ~]#
```