Laboratory Manual

FILE STRUCTURES LABORATORY WITH MINI PROJECT
18ISL67

Prepared by
SUSHEELAMMA K H
Assistant Professor
Department of ISE, SJCIT

S J C INSTITUTE OF TECHNOLOGY, CHICKBALLAPUR
DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
2018-2019

SJC Institute of Technology
Department of Information Science & Engineering

VISION

Educating students to engineer Information Science and Technology for advancing the Knowledge as to best serve the real world.

MISSION

1. **Focusing on Fundamentals and Applied aspects in both Information Science Theory and Programming practices.**
2. **Training comprehensively and encouraging R&D culture in trending areas of Information Technology.**
3. **Collaborating with premier Institutes and Industries to nurture Innovation and learning, in cutting edge Information Technology.**
4. **Educating and preparing the students who are much Sought-after, Productive and Well-respected for their work culture having Lifelong Learning practice.**
5. **Promoting ethical and moral values among the students so as to enable them emerge as responsible professionals.**

Program Educational Objectives

ISE Graduates should

PEO-1: Engage in Successful professional career in Information Science and Technology.
PEO-2: Pursue higher studies and research to advance the knowledge for Solving Contemporary Problems in IT industry.
PEO-3: Adapt to a constantly changing world through Professional Development and Sustained Learning.
PEO-4: Exhibit professionalism and team work with social concern.
PEO-5: Develop Leadership and Entrepreneurship Skills by incorporating organizational goals.

Program Specific Outcomes

PSO-1: Apply the knowledge of data structures, database systems, system programming, networking, web development and AI & ML techniques in engineering the software.
PSO-2: Exhibit solid foundations and advancements in developing software / hardware systems for solving contemporary problems.

FILE STRUCTURES LABORATORY WITH MINI PROJECT
[As per Choice Based Credit System (CBCS) scheme]
(Effective from the academic year 2017)
SEMESTER – VI

Subject Code	18ISL67	IA Marks	40
Number of Lecture Hours/Week	02I + 02P	Exam Marks	60
Total Number of Lecture Hours	40	Exam Hours	03
CREDITS – 02			

Lab Experiments:
PART A

1. Write a 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.

2. Write a program to read and write student objects with fixed-length records and thefields delimited by "|". Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.

3. Write a program to read and write student objects with Variable - Length records using any suitable record structure. Implement pack ( ), unpack ( ), modify ( ) and search ( ) methods.

4. Write a 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.

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

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.

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.

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

Part B --- Mini project:
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.

Beyond the syllabus experiments

1. Write a C++ program to implement B-Tree for a given set of integers and its operations insert ( ) and search ( ). Display the tree.

2. Write a C++ program to implement B+ tree for a given set of integers and its operations insert ( ), and search ( ). Display the tree.

Course Learning Objectives: This course (18CISL67) will enable students to:

1. Apply the concepts of Unix IPC to implement a given function.
2. Measure the performance of different file structures
3. Write a program to manage operations on given file system.
4. Demonstrate hashing and indexing techniques

Course Outcomes:

CO	Description
CO 1:	Implement operations related to files
CO 2:	Apply the concepts of file system to produce the given application
CO 3:	Evaluate performance of various file systems on given parameters.
CO 4:	Develop mini Project including Document processing, transaction management, indexing and hashing, buffer management, configuration management.
CO 5:	Demonstrate the mini project with results
CO 6:	Repost the developed project in a team

CO/PO/PSO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2
CO1	2	1	2		2						3	2	3	
CO2	2	2	3	2	2						2	2	3	
CO3	2	2	3	2	2	1	1		3	3	2	2	2	
CO4	2	3	2	1	2						3	2	3	
CO5	2	2	2	1	2						2	2	2	
CO6	3	3	2	1	2				2		2	2	3	
Avg	2.2	2.2	2.3	1.4	2	1	1		2.5	3	2.3	2	2.7	

Conduction of Practical Examination:

1. All laboratory experiments from part A are to be included for practical examination.
2. Mini project has to be evaluated for 40 Marks as per 6(b).
3. Report should be prepared in a standard format prescribed for project work.
4. Students are allowed to pick one experiment from the lot.
5. Strictly follow the instructions as printed on the cover page of answer script.
6. Marks distribution:

a) Part A: Procedure + Conduction + Viva: **09 + 42 +09 =60 Marks**

b) Part B: Demonstration + Report + Viva voce = **20+14+06 = 40 Marks**

7. Change of experiment is allowed only once and marks allotted to the procedure part to be made zero.

A. Basic Notes.

1. Where to type the programs??

- the program can be typed on any editor such as vi or any other linux editors. but the extension of the file should be ".cpp" ---> C++ files.

FOR vi editor

- in terminal type vi <filename>.cpp to open the file.
- after typing the program Press "Esc" foll by ":wq" to save the file and quit to terminal.

2. Where to see the output??

-The saved “source” file is executed in two stages:

1. COMPILATION. (The g++ compiler is used)

type "g++ <filename>.cpp" in the terminal.

3. EXECUTION.

the executable file is stored as a.out in the current directory so file can be executed as follows:

./a.out

(Note:Relative Path Name . -> (Current Directory))

B. Library: string

#include<string> -> C++ Header file

#include<string.h> -> C Header file

string is a C++ library.

string.h is a C library. differences between the two are listed below:

The C++ library has a string class (predefined data type for strings)

In C we can only use character arrays, there is no exclusive data type for strings.

char a[45], b[45]; -->C

string a, b; -->C++

Initializing is same in both cases.

a = "Sjcit";

b = "Cbpur";

ADVANTAGE??

by using C++ string classes we can use inbuilt functions and overloaded operators which have a much simpler and natural syntax than the corresponding C functions, thus no need of remembering special functions for strings.

a. Concatenation

strcat(a , b); -> C

a = a + b; -> C++ equivalent. + and = have already been defined in the library.

b. Copying

strcpy(a,b); -> Copies b to a

a = b; ->equivalent C++ code

c. Comparing

int flag = strcmp(a,b);

if(flag == 0) printf "both are same"; -> using C strings
if(a == b) printf "both are same"; -> using C++ strings

d. Length
intleng = strlen(a); -> using C String
intleng = a.length(); -> using C++ String class. (recall syntax of how member functions are called)

C++ is a superset of C. Thus both C strings as well as C++ strings can be used. If both are being used in a program remember to include <string.h> as well. In some cases we are forced to use character arrays instead of string objects.

C. Library: fstream
fstream is a C++ library that contains a fstream class which has several functions that handle various operations on files. But Before using functions we must have file name pointer which holds the logical file.

Syntax:
fstream fp;

a. Open a file
The first operation generally performed on an object of one of these classes is to associate it to a real file. This procedure is known as to open a file. An open file is represented within a program by a stream (i.e., an object of one of these classes; in the previous example, this was myfile) and any input or output operation performed on this stream object will be applied to the physical file associated to it.

In order to open a file with a stream object we use its member function open:

open (filename, mode);

Where filename is a string representing the name of the file to be opened, and mode is an optional parameter with a combination of the following flags:

To open a file we use open function

ios::in	Open for input operations.
ios::out	Open for output operations.
ios::binary	Open in binary mode. Set the initial position at the end of the file.
ios::ate	If this flag is not set, the initial position is the beginning of the file.
ios::app	All output operations are performed at the end of the file, appending the content to the current content of the file.
ios::trunc	If the file is opened for output operations and it already existed, its previous content is deleted and replaced by the new one.

All these flags can be combined using the bitwise operator OR (|). For example, if we want to open the fileexample.bin in binary mode to add data we could do it by the following call to member function open:

```
1. ofstream myfile;

2. myfile.open ("example.bin", ios::out | ios::app | ios::binary);
```

Each of the open member functions of classes ofstream, ifstream and fstream has a default mode that is used if the file is opened without a second argument:

class	default mode parameter
ofstream	ios::out
ifstream	ios::in
fstream	ios::in | ios::out

For ifstream and ofstream classes, ios::in and ios::out are automatically and respectively assumed, even if a mode that does not include them is passed as second argument to the open member function (the flags are combined).

For fstream, the default value is only applied if the function is called without specifying any value for the mode parameter. If the function is called with any value in that parameter the default mode is overridden, not combined.

File streams opened in binary mode perform input and output operations independently of any format considerations. Non-binary files are known as text files, and some translations may occur due to formatting of some special characters (like newline and carriage return characters).

Since the first task that is performed on a file stream is generally to open a file, these three classes include a constructor that automatically calls the open member function and has the exact same parameters as this member. Therefore, we could also have declared the previous myfile object and conduct the same opening operation in our previous example by writing:
ofstream myfile ("example.bin", ios::out | ios::app | ios::binary);

Combining object construction and stream opening in a single statement. Both forms to open a file are valid and equivalent.

b. Closing a file

When we are finished with our input and output operations on a file we shall close it so that the operating system is notified and its resources become available again. For that, we call the stream's member function close. This member function takes flushes the associated buffers and closes the file:

```
myfile.close();
```

Once this member function is called, the stream object can be re-used to open another file, and the file is available again to be opened by other processes.

In case that an object is destroyed while still associated with an open file, the destructor automatically calls the member function close.

c. Reading a line from the file and storing it in a c++ string

getline(<filepointer>,<string>,[<delim>]);

eg.
getline(fp, buf, ';');

Extracts line from fp and stores in buf until delimiter character(;) is found or end of file is found. Delimeter is optional. Default ('\n') is used if notspecified.

d. Writing a line to the file
simply use "<<" operator
stringbuf = "sjcit";
fp<<buf;

Note: make sure put pointer is at correct position.

e. Moving the read and write pointers

a logical file has to pointers
-get pointer -put pointer for reading and writing
intpos = fp.tellg();-> used to find out current position of get pointer and store in "pos"
intpos = fp.tellp();-> used to find out current position of put pointer and store in "pos"
Moving:
fp.seekp(27, ios::beg); move the put pointer to 27 bytes from beginning of file
fp.seekg(0, ios::end); mov the get pointer to 0 bytes from end of file

f. Other modes:
ios::cur move --- pointer -- bytes from current position

g. Other functions
system("any_unix_command"); -->executes any unix command.
eg
1. Unix command date can be executed withing program using system("date");
2. TO clear the terminal screen the clear command is used. Thus within the program we have to use system("clear");

NOTE: WE CANNNOT USE clrscr(); SINCE THERE IS NO CONIO LIBRARY IN GCC.

1. Write a 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 <cstring>
#include <fstream>
using namespace std;
int main()
{
char s1[25];
fstream f1,f2;
int i = 0, j = 0, x = 0, c = 0, kb = 0;
char fname1[25],fname2[25];
cout << "1:For std i/o\n2.For file i/o\n";
cout << "Enter your choice :\n";
cin >> kb;
switch(kb)
{
case 1:
cout << "Enter number of names :";
cin >> c;
for(j = 1;j <= c; j++)
{
cout << "Enter name :" << j << ":";
cin >> s1;
x = strlen(s1);
cout << "Reversed name :" << j << ":";
// Code to reverse the input
for(i = x-1; i >= 0; i--)
cout << s1[i];
```

```
// Display on standard output
cout << endl;
}
break;
case 2:
cout << "Enter datafile name :";
cin >> fname1;

cout << "Enter reverse datafile name :";
cin >> fname2;
// Opening requested files
f1.open(fname1, ios::in);
f2.open(fname2, ios::out);
if(!f1)
{
cerr << "File doesnot exist: " << fname1;
return -1;
}
if(!f2)
{
cerr << "File doesnot exist: " << fname2;
return -1;
}
while(1)
{
f1.getline(s1,25);
if(f1.fail())
break;
x = strlen(s1);
for(i = x-1; i >= 0; i--)
f2 << s1[i];
f2 << endl;
```

```
}
f1.close();
f2.close();
break;
}
return 0;
}
```

OUTPUT:

```
[root@localhost~] gedit 1.cpp
[root@localhost~] g++ 1.cpp
[root@localhost~]./a.out
```

Using Standard I/O

```
1:for standard I/O
2:for file I/O
Enter ur choice:1
Enter number of names:3
Enter name 1: CHANDU
Reversed name 1:UDNAHC
Enter name 2: DIVYA
Reversed name 2:AYVID
Enter name 2: RAMYA
Reversed name 2:AYMAR
```

Using File I/O

```
vi input.txt
CHANDU
DIVYA
RAMYA
vi output.txt
```

1:for standard I/O

2:for file I/O

Enter ur choice:2

Enter input file name:input.txt

Enter output file name:output.txt

Cat output.txt

UDNAHC

AYVID

AYMAR

I/O redirection and pipes

Using Pipes

[root@localhost~]cat output.txt|sort

AYMAR

AYVID

UDNAHC

Redirection of standard input

[root@localhost~]cat output.txt|sort>sortedrev.txt

AYMAR

AYVID

UDNAHC

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

```
#include <iostream>
#include <cstring>
#include <fstream>
#include<stdlib.h>
#include<string>
#define SIZE 50
int count=0;
using namespace std;
class fixedlength
{
struct student
{
char usn[11];
char name[20];
char sem[6];
char dept[10];
};
public: void pack();
void unpack();
void search();
void modify();
};
void fixedlength::pack()
{
char buf[SIZE];
student s;
cout<<"\n enter usn,name,sem,dept:";
cin>>s.usn>>s.name>>s.sem>>s.dept;
```

```
ofstream ofile;
ofile.open("student1.txt",ios::app);
sprintf(buf,"%s|%s|%s|%s|",s.usn,s.name,s.sem,s.dept);
int len=strlen(buf);
while(len<(SIZE-1))
{
strcat(buf,"_");
len++;
}
count++;
strcat(buf,"$");
ofile<<buf;
ofile.close();
}
void fixedlength::unpack()
{
char buf[SIZE],temp[100];
student s;
ifstream ifile;
ifile.open("student1.txt",ios::out);
int n=0;
cout<<"\n database: \n";
while(n<count)
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]| %[^|]|%[^|]|%[^$]",s.usn,s.name,s.sem,s.dept,temp);
cout<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept<<endl;
n++;
}
ifile.close();
}
```

```
void fixedlength::search()
{
char buf[SIZE],temp[100],usn[11];
student s;
ifstream ifile;
ifile.open("student1.txt",ios::out);
int n=0;
cout<<"\n enter the usn to be searched:";
cin>>usn;
cout<<"\n database:\n";
while(n<count)
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]| %[^|]|%[^|]|%[^$]",s.usn,s.name,s.sem,s.dept,temp);
if (strcmp(s.usn,usn)==0)
{
cout<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept<<endl;
}
n++;
}
ifile.close();
}
void fixedlength::modify()
{
char buf[SIZE],temp[100],usn[11];
student s;
ifstream ifile;
ifile.open("student1.txt",ios::out);
int n=0,ch;
cout<<"\n enter the usn to be modified:";
cin>>usn;
```

```
cout<<"\n database:\n";
while(n<count)
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]| %[^|]|%[^|]|%[^$]",s.usn,s.name,s.sem,s.dept,temp);
if (strcmp(s.usn,usn)==0)
{
cout<<"\n key found \n 1:modify name 2:modify sem 3:modify dept\n enter ur choice:";
cin>>ch;
switch(ch)
{
case 1: cout<<"\n enter name:";
cin>>s.name;
break;
case 2: cout<<"\n enter sem:";
cin>>s.sem;
break;
case 3: cout<<"\n enter dept:";
cin>>s.dept;
break;
default:break;
}
ofstream ofile;
ofile.open("student1.txt",ios::in);
sprintf(buf,"%s|%s|%s|%s|",s.usn,s.name,s.sem,s.dept);
int len=strlen(buf);
while(len<(SIZE-1))
{
strcat(buf,"_");
len++;
}
```

```
strcat(buf,"$");
ofile.seekp((n*SIZE),ios::beg);
ofile<<buf;
ofile.close();
}
n++;
}
ifile.close();
}
int main()
{
fixedlength f;
char buffer[SIZE];
ifstream ifile;
ifile.open("student1.txt",ios::out);
while(!ifile.eof())
{
ifile.getline(buffer,100,'$');
count++;
}
count--;
ifile.close();
int ch;
for(;;)
{
cout<<"\n 1:pack\t 2:unpack\t 3:search\t 4:modify\t 5:exit\n enter ur choice\n";
cin>>ch;
switch(ch)
{
case 1:f.pack();
break;
```

```
case 2:f.unpack();
break;
case 3:f.search();
break;
case 4:f.modify();
break;
default:exit(0);
}
}
return 0;
}
```

OUTPUT:

```
[root@localhost ~]# gedit prog2.cpp
[root@localhost ~]# g++ prog2.cpp
[root@localhost ~]# gedit student1.txt
[root@localhost ~]# ./a.out

1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
1
enter usn,name,sem,dept:
1sj11is001
chandu
3
ise

1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
1
enter usn,name,sem,dept:
```

1sj11is002
chethan
4
cse

1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
1
enter usn,name,sem,dept:
1sj11is004
bindu
5
Cse

1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
2
database:
1sj11is001 chandu 3 ise
1sj11is002 chethan 4 cse
1sj11is004 bindu 5 cse

1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
3
enter the usn to be searched:1sj11is002
database:
1sj11is002 chethan 4 cse

enter ur choice
4

enter the usn to be modified:1sj11is002

database:

key found

1:modify name 2:modify sem 3:modify dept

enter ur choice:2

enter sem:5

1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice

2

database:

1sj11is001 chandu 3 ise

1sj11is002 chethan 5 cse

1sj11is004 bindu 5 cse

1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice

5

[root@localhost ~]# vi student1.txt

1sj11is001|chandu|3|ise|_____$1sj11is002|chethan|5|cse|

_____$1sj11is004|bindu|5|cse|_____$

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

```
#include <iostream>
#include <cstring>
#include <fstream>
#include<stdlib.h>
#include<string>
using namespace std;
int count=0;
class variablelength
{
struct student
{
char usn[11];
char name[20];
char sem[6];
char dept[10];
};
public: void pack();
void unpack();
void search();
void modify();
};
void variablelength::pack()
{
char buf[100];
student s;
cout<<"\n enter usn,name,sem,dept:\n";
cin>>s.usn>>s.name>>s.sem>>s.dept;
```

```
ofstream ofile;
ofile.open("student2.txt",ios::app);
sprintf(buf,"%s|%s|%s|%s|$",s.usn,s.name,s.sem,s.dept);
count++;
ofile<<buf;
ofile.close();
}
void variablelength::unpack()
{
char buf[100],temp[100];
student s;
ifstream ifile;
ifile.open("student2.txt",ios::out);
int n=0;
cout<<"\n database: \n";
while(n<count)
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]|%[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept,temp);
cout<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept<<endl;
n++;
}
ifile.close();
}
void variablelength::search()
{
char buf[100],temp[100],usn[11];
int flag=0;
student s;
ifstream ifile;
ifile.open("student2.txt",ios::out);
```

```
int n=0;
cout<<"\n enter the usn to be searched:";
cin>>usn;
while(n<count)
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]| %[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept,temp);
if (strcmp(s.usn,usn)==0)
{
flag=1;
cout<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept<<endl;
}
n++;
}
if(flag==0)
cout<<"\n key not found";
ifile.close();
}
void variablelength::modify()
{
char buf[100],temp[100],usn[11];
int flag=0;
student s;
ifstream ifile;
ifile.open("student2.txt",ios::out);
int n=0,ch;
cout<<"\n enter the usn to be modified:";
cin>>usn;
ofstream ofile;
ofile.open("student2.txt",ios::in);
while(n<count)
```

```
{
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]| %[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept,temp);
if (strcmp(s.usn,usn)==0)
{
flag=1;
cout<<"\n key found \n 1:modify name 2:modify sem 3:modify dept\n enter ur choice:";
cin>>ch;
switch(ch)
{
case 1: cout<<"\n enter name:";
cin>>s.name;
break;
case 2: cout<<"\n enter sem:";
cin>>s.sem;
break;
case 3: cout<<"\n enter dept:";
cin>>s.dept;
break;
default:break;
}
}
sprintf(buf,"%s|%s|%s|%s|$",s.usn,s.name,s.sem,s.dept);
ofile<<buf;
n++;
}
if(flag==0)
cout<<"\n key not found";
ofile.close();
ifile.close();
}
```

```
int main()
{
variablelength v;
char buffer[100];
ifstream ifile;
ifile.open("student2.txt",ios::out);
while(!ifile.eof())
{
ifile.getline(buffer,100,'$');
count++;
}
count--;
ifile.close();
int ch;
for(;;)
{
cout<<"\n 1:pack\t 2:unpack\t 3:search\t 4:modify\t 5:exit\n enter ur choice\n";
cin>>ch;
switch(ch)
{
case 1:v.pack();
break;
case 2:v.unpack();
break;
case 3:v.search();
break;
case 4:v.modify();
break;
default:exit(0);
}
}
```

```
return 0;
}
```

OUTPUT:

```
[root@localhost ~]# gedit prog3.cpp
 [root@localhost ~]# g++ prog3.cpp
[root@localhost ~]# gedit student2.txt
[root@localhost ~]# ./a.out
1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice
1
enter usn,name,sem,dept:
1sj11is001
amith
3
ise
1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice
1
enter usn,name,sem,dept:
1sj11is002
ankur
4
cse
1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice
1
enter usn,name,sem,dept:
```

1sj11is003
mahesh
4
cse
1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
2
database:
1sj11is001 amith 3 ise
1sj11is002 ankur 4 cse
1sj11is003 mahesh 4 cse
1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
3
enter the usn to be searched:1sj11is002
1sj11is002 ankur 4 cse
1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
4
enter the usn to be modified:1sj11is002
key found
1:modify name 2:modify sem 3:modify dept
enter ur choice:1
enter name:ankurjm
key not found
1:pack 2:unpack 3:search 4:modify 5:exit
enter ur choice
2
database:
1sj11is001 amith 3 ise
1sj11is002 ankurjm 4 cse

1sj11is003 mahesh 4 cse

1:pack 2:unpack 3:search 4:modify 5:exit

enter ur choice

5

[root@localhost ~]# vi student2.txt

1sj11is001|amith|3|ise|$1sj11is002|ankurjm|4|cse|$1sj11is003|mahesh|4|cse|$

4. Write a 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 <cstring>
#include <fstream>
#include<stdlib.h>
#include<string>
using namespace std;
int count=0;
class variablerrn
{
struct student
{
char usn[11];
char name[20];
char sem[6];
char dept[10];
};
public: void pack();
void unpack();
};
void variablerrn::pack()
{
char buf[100],c[20];
student s;
cout<<"\n enter usn,name,sem,dept:\n";
cin>>s.usn>>s.name>>s.sem>>s.dept;
ofstream ofile;
ofile.open("student3.txt",ios::app);
ofile.seekp(0,ios::end);
```

```
long p=ofile.tellp();
sprintf(c,"%d|%ld|$",count+1,p);
sprintf(buf,"%s|%s|%s|%s|$",s.usn,s.name,s.sem,s.dept);
count++;
ofile<<buf;
ofstream rrn;
rrn.open("rrn.txt",ios::app);
rrn<<c;
rrn.close();
ofile.close();
}
void variablerrn::unpack()
{
char buf[100],temp[100],c[20],t[50];
student s;
int rrn1,frrn,n=0,flag=0;
long pos;
cout<<"\n Enter record number:\n";
cin>>rrn1;
ifstream rrn;
rrn.open("rrn.txt",ios::out);
rrn.seekg(0,ios::beg);
ifstream ifile;
ifile.open("student3.txt",ios::out);
ifile.seekg(0,ios::beg);
while(n<=count)
{
rrn.getline(c,20,'$');
sscanf(c,"%d|%ld|$",&frrn,&pos,t);
if(rrn1==frrn)
{
```

```
flag=1;
ifile.seekg(pos,ios::beg);
cout<<"\n database: \n";
ifile.getline(buf,100,'$');
sscanf(buf,"%[^|]|%[^|]|%[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept,temp);
cout<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept<<endl;
break;
}
n++;
}
if(flag==0)
cout<<"\n key not found";
rrn.close();
ifile.close();
}
int main()
{
variablerrn r;
char buffer[100];
ifstream ifile;
ifile.open("student3.txt",ios::out);
ifile.seekg(0,ios::beg);
while(!ifile.eof())
{
ifile.getline(buffer,100,'$');
count++;
}
count--;
ifile.close();
int ch;
for(;;)
```

```
{
cout<<"\n 1:pack\t 2:unpack\t 3:exit\n Enter ur choice\n";
cin>>ch;
switch(ch)
{
case 1:r.pack();
break;
case 2:r.unpack();
break;
default:exit(0);
}
}
return 0;}
```

OUTPUT:

```
[root@localhost ~]# gedit prog4.cpp
[root@localhost ~]# g++ prog4.cpp
 [root@localhost ~]# gedit student3.txt
[root@localhost ~]# gedit rrn.txt
[root@localhost ~]# ./a.out
1:pack 2:unpack 3:exit
Enter ur choice
1
enter usn,name,sem,dept:
1sj11is001
chaturvi
3
ise
1:pack 2:unpack  3:exit
```

Enter ur choice
1
enter usn,name,sem,dept:
1sj11is002
pallavi
4
cse
1:pack 2:unpack 3:exit

Enter ur choice
1
enter usn,name,sem,dept:
1sj11is003
rekha
6
ise
1:pack 2:unpack 3:exit
Enter ur choice
2
Enter record number:
3
database:
1sj11is003 rekha 6 ise
1:pack 2:unpack
Enter ur choice
3
3:exit
[root@localhost ~]# vi student3.txt
1sj11is001|chaturvi|3|ise|$1sj11is002|pallavi|4|cse|$1sj11is003|rekha|6|ise|$
[root@localhost ~]# vi rrn.txt
1|0|$2|27|$3|53|$

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 <cstdio>
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <cstring>
using namespace std;
class primaryindex
{
struct student
{
char usn[11],name[15],sem[10],dept[15];
};
public: void add();
void search();
void delete1();
void setup();
};
void primaryindex::setup()
{
ofstream out1,out2;
out1.open("index.txt",ios::in|ios::trunc);
out2.open("student4.txt",ios::in|ios::trunc);
out1.close();
out2.close();
}
void primaryindex::add()
{
char buffer[100],temp[50],usn[11],temp1[50];
```

```
int pos1,flag=0;
student s;
ifstream out2;
out2.open("index.txt",ios::in);
cout<<"\nEnter usn,name,sem,dept :";
cin>>s.usn>>s.name>>s.sem>>s.dept;
while(!out2.eof())
{
out2.getline(temp,50,'$');
sscanf(temp,"%[^|]|%d|$",usn,&pos1);
if(strcmp(s.usn,usn)==0)
{
flag=1;
break;
}
}
out2.close();
if(flag==1)
cout<<"\nPrimary key constraint violation,record not inserted";
else
{
ofstream out1,out2;
out1.open("student4.txt",ios::app);
out2.open("index.txt",ios::app);
out1.seekp(0,ios::end);
long pos=out1.tellp();
sprintf(buffer,"%s|%s|%s|%s|$",s.usn,s.name,s.sem,s.dept);
out1<<buffer;
sprintf(temp1,"%s|%ld|$",s.usn,pos);
out2<<temp1;
out1.close();
```

```
out2.close();
}
}
void primaryindex::search()
{
char buffer[100],temp[50],usn[11],usn1[11];
int pos,flag=0;
student s;
cout<<"\nEnter usn to be searched";
cin>>usn;
ifstream out1,out2;
out2.open("index.txt",ios::in);
while(!out2.eof())
{
out2.getline(temp,50,'$');
sscanf(temp,"%[^|]|%d|$",usn1,&pos);
if(strcmp(usn1,usn)==0)
{
out1.open("student4.txt",ios::out);
out1.seekg(pos,ios::beg);
out1.getline(buffer,100,'$');
sscanf(buffer,"%[^|]|%[^|]|%[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept);
cout<<"\nRecord found";
flag=1;
cout<<endl<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept;
break;
}
}
if(flag==0)
cout<<"\nRecord doesn't exist";
out1.close();
```

```
out2.close();
}
void primaryindex::delete1()
{
char buffer[100],temp[50],usn[11],usn1[11];
int pos,flag=0;
student s;
cout<<"\nEnter usn to be deleted:";
cin>>usn;
ifstream in1,in2;
ofstream of1,of2;
in1.open("index.txt",ios::in);
in2.open("student4.txt",ios::in);
of1.open("index1.txt",ios::out);
of2.open("student41.txt",ios::out);
while(1)
{
in1.getline(temp,50,'$');
if(in1.eof())
break;
in2.getline(buffer,100,'$');
sscanf(temp,"%[^|]|%d|$",usn1,&pos);
strcat(buffer,"$");
strcat(temp,"$");
int len=strlen(buffer);
if(strcmp(usn,usn1)!=0)
{
of1<<temp;
of2<<buffer;
}
else
```

```
{
flag=1;
for(int i=0;i<len;i++)
buffer[i]='*';
of2<<buffer;
}
}
if(flag)
cout<<"\nRecord deleted";
else
cout<<"\nRecord doesn't exists";
in1.close();
in2.close();
of1.close();
of2.close();
remove("index.txt");
remove("student4.txt");
rename("index1.txt","index.txt");
rename("student41.txt","student4.txt");
}
int main()
{
int ch;
primaryindex p;
p.setup();
for( ; ; )
{
cout<<"\n1:add 2:Search 3:delete 4:exit";
cout<<"\n enter the choice :";
cin>>ch;
switch(ch)
```

```
{
case 1:p.add();
break;
case 2:p.search();
break;
case 3:p.delete1();
break;
default:exit(0);
}
}
return 0;
}
```

OUTPUT:

```
[root@localhost ~]# vi student4.txt
[root@localhost ~]# vi index.txt
[root@localhost ~]# g++ prog5.cpp
[root@localhost ~]# ./a.out
1:add 2:Search 3:delete 4:exit
enter the choice :1
Enter usn,name,sem,dept :
1sj11is001
amrutha
4
cse
1:add 2:Search 3:delete 4:exit
enter the choice :1
Enter usn,name,sem,dept :
1sj11is002
Pallavi
4
```

cse

1:add 2:Search 3:delete 4:exit

enter the choice :1

Enter usn,name,sem,dept :

1sj11is003

chaturvi

4

cse

1:add 2:Search 3:delete 4:exit

enter the choice :2

Enter usn to be searched1sj11is003

Record found

1sj11is003 chaturvi 4 cse

1:add 2:Search 3:delete 4:exit

enter the choice :3

Enter usn to be deleted:1sj11is002

Record deleted

1:add 2:Search 3:delete 4:exit

enter the choice :4

[root@localhost ~]# vi student4.txt

1sj11is001|amrutha|4|cse|$**********1sj11is003|chaturvi|4|cse|$

[root@localhost ~]# vi index.txt

1sj11is001|0|$1sj11is003|52|$

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 <cstdio>
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <cstring>
using namespace std;
class secondaryindex
{
struct student
{
char usn[11],name[15],sem[10],dept[15],n;
};
public: void add();
void search();
void delete1();
void setup();
};
void secondaryindex::setup()
{
ofstream out1,out2;
out1.open("index2.txt",ios::in|ios::trunc);
out2.open("student5.txt", ios::in|ios::trunc);
out1.close();
out2.close();
}
void secondaryindex::add()
{
char buffer[100],temp[50],usn[11],temp1[50],name[15];
```

```
int pos1,flag=0;
student s;
ifstream out2;
out2.open("index2.txt",ios::in);
cout<<"\nEnter usn,name,sem,dept:";
cin>>s.usn>>s.name>>s.sem>>s.dept;
while(!out2.eof())
{
out2.getline(temp,50,'$');
sscanf(temp,"%[^|]|%[^|]|%d|$",usn,name,&pos1);
if(strcmp(s.usn,usn)==0)
{
flag=1;
break;
}
}
out2.close();
if(flag==1)
cout<<"\nPrimary key constraints violation,record not inserted";
else
{
ofstream out1,out2;
out1.open("student5.txt",ios::app);
out2.open("index2.txt",ios::app);
out1.seekp(0,ios::end);
long pos=out1.tellp();
sprintf(buffer,"%s|%s|%s|%s|$",s.usn,s.name,s.sem,s.dept);
out1<<buffer;
sprintf(temp1,"%s|%s|%ld|$",s.usn,s.name,pos);
out2<<temp1;
out1.close();
```

```
out2.close();
}
}
void secondaryindex::search()
{
char buffer[100],temp[50],usn[11],name[15],name1[15];
int pos,flag=0;
student s;
cout<<"\nEnter name to be searched";
cin>>name;
ifstream out1,out2;
out2.open("index2.txt",ios::in);
out1.open("student5.txt",ios::out);
while(1)
{
out2.getline(temp,50,'$');
if(out2.eof())
break;
sscanf(temp,"%[^|]|%[^|]|%d|$",usn,name1,&pos);
out1.getline(buffer,100,'$');
sscanf(buffer,"%[^|]|%[^|]|%[^|]|%[^|]|$",s.usn,s.name,s.sem,s.dept);
if(strcmp(name1,name)==0)
{
cout<<"\nRecord found";
flag=1;
cout<<endl<<s.usn<<" "<<s.name<<" "<<s.sem<<" "<<s.dept;
}
}
if(flag==0)
cout<<"\nRecord doesn't exist";
out1.close();
```

```
out2.close();
}
void secondaryindex::delete1()
{
char buffer[100],temp[50],usn[11],name[15],name1[15];
int pos,flag=0;
student s;
cout<<"\nEnter name to be deleted";
cin>>name;
ifstream in1,in2;
ofstream of1,of2;
in1.open("index2.txt",ios::in);
in2.open("student5.txt",ios::in);
of1.open("index21.txt",ios::out);
of2.open("student51.txt",ios::out);
while(1)
{
in1.getline(temp,50,'$');
if(in1.eof())
break;
in2.getline(buffer,100,'$');
sscanf(temp,"%[^|]|%[^|]|%d|$",usn,name1,&pos);
strcat(temp,"$");
int len=strlen(buffer);
if(strcmp(name,name1)!=0)
{
of1<<temp;
of2.write(buffer,len);
}
else
flag=1;
```

```
}
if(flag)
cout<<"\nRecord deleted";
else
cout<<"\nRecord doesn't exists";
in1.close();
in2.close();
of1.close();
of2.close();
remove("index2.txt");
remove("student5.txt");
rename("index21.txt","index2.txt");
rename("student51.txt","student5.txt");
}
int main()
{
int ch;
secondaryindex si;
si.setup();
for(;;)
{
cout<<"\n1:add\t2:Search\t3:delete\t4:exit";
cout<<"\nEnter the choice:";
cin>>ch;
switch(ch)
{
case 1:si.add();
break;
case 2:si.search();
break;
case 3:si.delete1();
```

```
break;
default:exit(0);
}
}
return 0;
}
```

OUTPUT:

```
[root@localhost ~]# vi student5.txt
[root@localhost ~]# g++ prog6.cpp
[root@localhost ~]# ./a.out
1:add  2:Search  3:delete  4:exit
Enter the choice:1
Enter usn,name,sem,dept:
1sj11is001
amrutha
5
cse
1:add 2:Search  3:delete 4:exit
Enter the choice:1
Enter usn,name,sem,dept:
1sj11is002
chandu
5
cse
1:add 2:Search 3:delete 4:exit
Enter the choice:1
Enter usn,name,sem,dept:
1sj11is003
megha
5
```

cse

1:add 2:Search 3:delete 4:exit

Enter the choice:2

Enter name to be searched chandu

Record found

1sj11is002 chandu 5 cse

1:add 2:Search 3:delete 4:exit

Enter the choice:3

Enter name to be deleted chandu

Record deleted

1:add 2:Search 3:delete 4:exit

Enter the choice:4

[root@localhost ~]# vi student5.txt

1sj11is001|amrutha|5|cse|$1sj11is003|megha|5|cse|$

[root@localhost ~]# vi index2.txt

1sj11is001|amrutha|0|$1sj11is003|megha|51|$

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 <cstring>
#include <fstream>
using namespace std;
int m,n;
void writelist()
{
fstream out1,out2;
int i;
char name[20];
out1.open("1.txt",ios::out);
out2.open("2.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 the names in ascending order:\n";
for(i=0;i<n;i++)
{
cin>>name;
out2<<name<<'\n';
```

```
}
}
void match()
{
char list1[50][20],list2[50][20];
int i,j;
fstream out1,out2,out3;
out1.open("1.txt",ios::in);
out2.open("2.txt",ios::in);
out3.open("out.txt",ios::out);
i=0;
out1.getline(list1[i],30,'\n');
cout<<"The names in file1 are:\n";
while(!out1.eof())
{
cout<<list1[i]<<endl;
i++;
out1.getline(list1[i],30,'\n');
}
i=0;
cout<<"The 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];
out3<<list1[i]<<'\n';
i++;j++;
}
else if(strcmp(list1[i],list2[j])<0)
i++;
else
j++;
}
}
int main()
{
writelist();
match();
return 0;
}
```

OUTPUT:

```
[root@localhost ~]# vi 1.txt
[root@localhost ~]# vi 2.txt
[root@localhost ~]# vi out.txt
[root@localhost ~]# g++ prog7.cpp
[root@localhost ~]# ./a.out
Enter no of names in file1:5
Enter the names in ascending order:
amrutha
anvitha
chandu
```

deemanth
pallavi
Enter no of names in file2:5
Enter the names in ascending order:
anusha
arun
anvitha
chaturvi
pallavi
The names in file1 are:
amrutha
anvitha
chandu
deemanth
pallavi
The names in file2 are:
anusha
arun
anvitha
chaturvi
pallavi
Common names are:
pallavi
[root@localhost ~]# vi 1.txt/Cat 1.txt
amrutha
anvitha
chandu
deemanth
pallavi
[root@localhost ~]# vi 2.txt/cat 2.txt
anusha

arun
anvitha
chaturvi
pallavi
[root@localhost ~]# vi out.txt/cat 2.txt
pallavi

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

```
#include<iostream>
#include<fstream>
#include<cstring>
using namespace std;
class record
{
public:
char name[20];
char usn[20];
}rec[20];
int no;
fstream file[8];
char fname[8][8]={"1.txt","2.txt","3.txt","4.txt","5.txt","6.txt","7.txt","8.txt"};
void merge_file(char * file1,char *file2, char *filename)
{
record rcd[20];
int k=0;
fstream f1,f2;
f1.open(file1,ios::in);
f2.open(file2,ios::in);
while(!f1.eof())
{
f1.getline(rcd[k].name,20,'|');
f1.getline(rcd[k++].usn,20,'\n');
}
while(!f2.eof())
{
f2.getline(rcd[k].name,20,'|');
```

```
f2.getline(rcd[k++].usn,20,'\n');
}
record temp;
int t,y;
for(t=0;t<k-2;t++)
for(y=0;y<k-t-2;y++)
if(strcmp(rcd[y].name,rcd[y+1].name)>0)
{
temp=rcd[y];
rcd[y]=rcd[y+1];
rcd[y+1]=temp;
}
fstream temp1;
temp1.open(filename,ios::out);
for(t=1;t<k-1;t++)
temp1<<rcd[t].name<<"|"<<rcd[t].usn<<"\n";
f1.close();
f2.close();
temp1.close();
return;
}
void kwaymerge()
{
char filename[7][20]={"11.txt","22.txt","33.txt","44.txt","111.txt","222.txt","1111.txt"};
int i,k=0;
for(i=0;i<8;i=i+2)
merge_file(fname[i],fname[i+1],filename[k++]);
merge_file(filename[0],filename[1],filename[4]);
merge_file(filename[2],filename[3],filename[5]);
merge_file(filename[4],filename[5],filename[6]);
return;
```

```
}
int main()
{
int i;
cout<<"\nEnter the no:of records\n";
cin>>no;
cout<<"\nEnter the details\n";
for(i=0;i<8;i++)
file[i].open(fname[i],ios::out);
for(i=0;i<no;i++)
{
cout<<"\nName: ";
cin>>rec[i].name;
cout<<"\nUsn: ";
cin>>rec[i].usn;
file[i%8]<<rec[i].name<<"|"<<rec[i].usn<<"\n";
}
for(i=0;i<8;i++)
file[i].close();
kwaymerge();
fstream result;
result.open("1111.txt",ios::in);
cout<<"\nSorted records:\n";
char name[20],usn[20];
for(i=0;i<no;i++)
{
result.getline(name,20,'|');
result.getline(usn,20,'\n');
cout<<"\nName: "<<name<<"\nUsn: "<<usn<<"\n";
}
return 0;
```

}

OUTPUT:

```
[root@localhost ~]# vi prog8.cpp
[root@localhost ~]# g++ prog8.cpp
[root@localhost ~]# ./a.out
Enter the no:of records
10
Enter the details
Name: anusha
Usn: 1sj11is001
Name: bindu
Usn: 1sj11is002
Name: chaturvi
Usn: 1sj11is003
Name: deemanth
Usn: 1sj11is004
Name: eshwar
Usn: 1sj11is005
Name: fathima
Usn: 1sj11is006
Name: geetha
Usn: 1sj11is007
Name: hema
Usn: 1sj11is008
Name: ilayaraj
Usn: 1sj11is009
Name: jamuna
Usn: 1sj11is010
Sorted records:
Name: anusha
```

Usn: 1sj11is001

Name: bindu

Usn: 1sj11is002

Name: chaturvi

Usn: 1sj11is003

Name: deemanth

Usn: 1sj11is004

Name: eshwar

Usn: 1sj11is005

Name: fathima

Usn: 1sj11is006

Name: geetha

Usn: 1sj11is007

Name: hema

Usn: 1sj11is008

Name: ilayaraj

Usn: 1sj11is009

Name: jamuna

Usn: 1sj11is010

[root@localhost ~]# cat 1.txt

anusha|1sj11is001

ilayaraj|1sj11is009

[root@localhost ~]# cat 2.txt

bindu|1sj11is002

jamuna|1sj11is010

[root@localhost ~]# cat 3.txt

chaturvi|1sj11is003

[root@localhost ~]# cat 4.txt

deemanth|1sj11is004

[root@localhost ~]# cat 5.txt

eshwar|1sj11is005

```
[root@localhost ~]# cat 6.txt
fathima|1sj11is006
[root@localhost ~]# cat 7.txt
geetha|1sj11is007
[root@localhost ~]# cat 8.txt
hema|1sj11is008
[root@localhost ~]# cat 11.txt
anusha|1sj11is001
bindu|1sj11is002
ilayaraj|1sj11is009
jamuna|1sj11is010
[root@localhost ~]# cat 22.txt
chaturvi|1sj11is003
deemanth|1sj11is004
[root@localhost ~]# cat 33.txt
eshwar|1sj11is005
fathima|1sj11is006
[root@localhost ~]# cat 44.txt
geetha|1sj11is007
hema|1sj11is008
[root@localhost ~]# cat 111.txt
anusha|1sj11is001
bindu|1sj11is002
chaturvi|1sj11is003
deemanth|1sj11is004
ilayaraj|1sj11is009
jamuna|1sj11is010
[root@localhost ~]# cat 222.txt
eshwar|1sj11is005
fathima|1sj11is006
geetha|1sj11is007
```

hema|1sj11is008

[root@localhost ~]# cat 1111.txt

anusha|1sj11is001

bindu|1sj11is002

chaturvi|1sj11is003

deemanth|1sj11is004

eshwar|1sj11is005

fathima|1sj11is006

geetha|1sj11is007

hema|1sj11is008

ilayaraj|1sj11is009

jamuna|1sj11is010

Beyond the syllabus experiments

3. Write a C++ program to implement B-Tree for a given set of integers and its operations insert ( ) and search ( ). Display the tree.

```
#include<iostream.h>
#include<stdio.h>
#include<fstream.h>
#include<stdlib.h>
#include<st
ring.h>
class node
{
      public:
            int a[4];
            node
            *next[4
            ]; node
            *parent
            ; int
            size;
            node();
};

node::node()
{
      for(int
            i=0;i<4
            ;i++)
            next[i]
            =NULL;
            parent=
            NULL;
            size=0;
}

class btree
{
      node
      *roo
      t;
      publ
      ic:
            node *findLeaf(int key,int &level);
            void updateKey(node *p,node *c,int
            newkey); void search(int key);
            void insert(int key);
            void insertIntoNode(node *n,int key,node
            *addresss); void promote(node *n,int key,node
            *addresss);
            node *split(node
```

```
            *n); void
            traverse(node
            *ptr); btree();
};

void btree::traverse(node *ptr)
{
      if(ptr==NULL)
      return;
      for(int i=0;i<ptr-
            >size;i++)
            cout<<ptr-
            >a[i]<<" ";
      cout<<endl;
      for(i=0;i<ptr-
      >size;i++)
      traverse(ptr-
      >next[i]);
}
btree::btree()
{
      root=NULL;
}

node* btree::findLeaf(int key,int &level)
{
      node
      *ptr=root;
      node
      *prevptr=NUL
      L; level=0;
      in
      t
      i;
      wh
      il
      e(
      pt
      r)
      {
            i=0;
            level++;
            while(i<ptr->size-1 &&
            key>ptr->a[i]) i++;
            prevptr=pt
            r;
            ptr=ptr-
            >next[i];
      }
      return prevptr;
}
```

```
node * btree::split(node *n)
{
      int midpoint=(n-
      >size+1)/2; int
      newsize=n->size-
      midpoint; node
      *newptr=new node;
      node *child;
      newptr->parent=n-
      >parent; int i;
      for(i=0;i<midpoin
      t;i++)
      {
            newptr->a[i]=n->a[i];
            newptr->next[i]=n-
            >next[i]; n->a[i]=n-
            >a[i+midpoint]; n-
            >next[i]=n-
            >next[i+midpoint];
      }
      n-
      >size=midpoint;
      newptr-
      >size=newsize;
      for(i=0;i<n-
      >size;i++)
      {
            child=n-
            >next[i];
            if(child!=
            NULL)
                  child->parent=n;
      }
      for(i=0;i<newptr->size;i++)
      {
            child=newptr-
            >next[i];
            if(child!=NULL)
                  child->parent=newptr;
      }
      return newptr;
}

void btree::updateKey(node *parent,node *child,int newkey)
{
      if(parent==NULL)
        return;
       if(parent-
        >size==0)
        return;
```

```
	int oldkey=child->a[child-
	>size-2]; for(int
	i=0;i<parent->size;i++)
		if(parent->a[i]==oldkey)
		{
			parent-
			>a[i]=newkey;
			parent-
			>next[i]=child;
		}
}

void btree::insertIntoNode(node *n,int key,node *address)
{
	int
	i;
	if(n
	==NU
	LL)
		return;
	for(i=0;i<n-
	>size;i++)
		if(n-
			>a[
			i]=
			=ke
			y)
			ret
			urn
			;
	i=n->size-1;
	while(i>=0 && n->a[i]>key)
	{
		n->a[i+1]=n-
		>a[i]; n-
		>next[i+1]=n-
		>next[i]; i--;
	}
	i++;
	n-
	>a[i
	]=ke
	y;
	n-
	>next[i]=add
	ress; n-
	>size++;
	if(i==n-
	>size-1)
		updateKey(n->parent,n,key);
}

void btree::promote(node *n,int key,node *address)
```

```
{
      if(n==NULL)
return;

      if(n->size<4)
      {
            insertIntoNode(n,key,address);
            return;
      }
      if(n==root)
      {
            root=ne
            w node;
            n-
            >parent
            =root;
      }
      node
      *newptr=split(n
      ); node *t;
      if(key<n->a[0])
            t=newptr;
      else
            t=n;
      insertIntoNode(t,key,address); promote(n-
      >parent,n->a[n->size-1],n); promote(newptr-
      >parent,newptr->a[newptr->size-1],newptr);
}

void btree::insert(int key)
{
      if(root==NULL)
      {
            root=new node;
            root->a[root-
            >size]=key;
            root->size++;
            return;
      }
      int level;
      node
      *leaf=findLeaf(key,leve
      l); int i;
      for(i=0;i<leaf-
      >size;i++)
            if(leaf->a[i]==key)
            {
                  cout<<"The key to be inserted already
                  exists"<<endl; return;
            }
      promote(leaf,key,NULL);
```

```
	cout<<"------------------\n";
	traverse(root);
	cout<<"------------------\n";
}

void btree::search(int key)
{
	if(root==NULL)
	{
		cout<<"The tree does not
		exist"<<endl; return;
	}
	int level;
	node
	*leaf=findLeaf(key,level)
	; int flag=0;
	for(int i=0;i<leaf-
		>size;i++)
		if(leaf-
		>a[i]==key)
		{
			flag=1;
			cout<<"The key "<<key<<" exists in the B-tree at the
			level
"<<level<<endl;
		}
	if(!flag)
		cout<<"The key searched for was not found"<<endl;
}

int main()
{
	btree b;
	int
	choice=1,k
	ey;
	while(choi
	ce<=2)
	{
		cout<<"1.Insert a
		key\n";
		cout<<"2.Search a
		key\n";
		cout<<"3.Exit\n";
		cout<<"Enter your
		choice: ";
		cin>>choice;
		switch(choice)
		{
			case 1: cout<<"Enter the key to be inserted in a B-
				tree\n"; cin>>key;
```

```
                        b.insert(key);
                        break;
                  case 2: cout<<"Enter the key to be
                        searched\n"; cin>>key;
                        b.search(key);
                        break;
            }
      }
      return 0;
}
```

Output :

```
1.Inse
rt    a
Key
2.Sear
ch    a
key
3.Exit
Enter u’r choice :1

Enter The Key to be inserted
in B-Tree 100

1.Inse
rt    a
Key
2.Sear
ch    a
key
3.Exit
Enter u’r choice :1

Enter The Key to be inserted
in B-Tree 50
-----------
50 100
-----------
1.Inse
rt    a
Key
2.Sear
ch    a
key
3.Exit
Enter u’r choice :1

Enter The Key to be inserted
in B-Tree 75
-------------
```

```
50 75 100
-------------
1.Inse
rt    a
Key
2.Sear
ch    a
key
3.Exit
Enter u’r choice :1

Enter The Key to be inserted
in B-Tree 200
---------------
50 75 100 200
---------------

1.Inse
rt    a
Key
2.Sear
ch    a
key
3.Exit
Enter u’r choice :2

Enter The key to be
searched 100
Key 100 exists in B-tree at level 1
```

4. Write a C++ program to implement B+ tree for a given set of integers and its operations insert ( ), and search ( ). Display the tree.

```
#include<iostream.h>
#include<process.h>
#include<stdio.h>
#include<conio.h>

struct node
{
      int elements[50];
      node
      *link[5],*first,*next,*parent
      ; int level;
};

int cur_level=0;

node *create_node()
{
      node *cur=new
      node; for(int
      i=0;i<5;i++)
      {
            cur-
            >elements[i]=
            -1; cur-
            >link[i]=NULL
            ;
      }
      cur-
      >next=NULL
      ; cur-
      >parent=NU
      LL; cur-
      >level=0;
      cur-
      >first=NUL
      L;
return cur;
}

node* search(node *root,int key,int depth)
{
      node
      *cur=roo
      t;
      if(depth
      ==0)
            retu
      rn root;
```

```
	int i=0,j;
	while(i<de
	pth)
	{
		for(j=0;j<=4;j++)
		{
			if(cur->elements[j]==-1)
			{
				cur=cur-
				>link[j-1];
				break;
			}
			if(key<cur->elements[j])
			{
				if(j==0)
					cur=cur
					-
					>first;
				else
					cur=cur->link[j-1];
				break;
			}
		}
		i++;
	}
	return cur;
}

int search(node *root,int key,int *index,int &k)
{
	node
	*cur=root;
	int i=0,j;
	while(i<cur
	_level)
	{
		for(j=0;j<4;j++)
		{
			if(cur->elements[j]==-1)
			{
				if(j==0)
					retur
				n 0;
				cur=cur-
				>link[j-1];
				index[k++]=
				j; break;
			}
			if(key<cur->elements[j])
			{
				if(j==0)
```

```
                        {
                              cur=cur-
                              >first;
                              index[k+
                              +]=0;
                        }
                        else
                        {
                              cur=cur-
                              >link[j-1];
                              index[k++]=
                              j;
                        }
                        break;
                  }
            }
            i++;
      }
      for(j=0;j<4;j++)
            if(key==cur-
            >elements[j])
            {
                  index[
                  k++]=j
                  ;
                  return
                  1;
            }
      return 0;
}

void update_level(node *cur)
{
      cur->level++;
      if(cur-
      >first!=NULL)
            update_level(cur-
      >first); for(int
      i=0;i<5;i++)
            if(cur->link[i]!=NULL)
                  update_level(cur-
                  >link[i]);
}

void display(node *root)
{
      node
      *q[100],*cur;
      int r=-
      1,f=0,i=0,j=0;
      q[++r]=root;
      cout<<"\nThe tree:
```

```
	\n";
	cout<<"\nLevel:
	"<<i<<endl;
	while(r>=f)
	{
		cur=q[f++]
		; if(cur-
		>level!=i)
			cout<<"\nLevel:
		"<<++i<<endl; if(cur-
		>first!=NULL)
			q[++r]=cur-
		>first;
		for(j=0;j<4;j++)
		{
			if(cur->elements[j]!=-1)
			{
				cout<<" "<<cur-
				>elements[j]; if(cur-
				>link[j]!=NULL)
					q[++r]=cur->link[j];
			}
			else
				break;
		}
		cout<<"\t";
	}
}

node *insert(node *root,int key,int level,int type,node *child1,node *child2)
{
	node
	*cur=search(root,key,level)
	; int i,j,flag=0;
	for(i=0;i<5;i++)
		if(cur-
		>elements[i]==-
		1)
			break;
		if(i==4)
			f
	lag=1;
	for(j=i;j>0;
	j--)
	{
		if(key<cur->elements[j-1])
		{
			cur->elements[j]=cur-
			>elements[j-1]; cur-
			>link[j]=cur->link[j-1];
		}
		else
```

```
	{
		cur-
		>elements[j]=k
		ey; cur-
		>link[j]=child
		1;
		if(child1!=NUL
		L)
			child1->parent=cur;
		break;
	}
}
if(j==0)
{
	cur-
	>elements[j]=k
	ey; cur-
	>link[j]=child
	1; cur-
	>first=child2;
	if(child1!=NUL
	L)
		child1-
	>parent=cur;
	if(child2!=NULL)
		child2->parent=cur;
}
if(flag==1)
{
	if(cur->level==0)
	{
		cur_level++;
		node
		*new_root=create_node(
		); new_root-
		>first=cur; cur-
		>parent=new_root;
		root=new_root;
		update_level(cur);
	}
	node
	*new_cur=create_node()
	; int next_key=cur-
	>elements[2];
	if(type==0)
	{
		new_cur-
		>next=cur->next;
		cur-
		>next=new_cur;
	}
```

```
            for(j=0;j<3;j++)
            {
                  new_cur->elements[j]=cur-
                  >elements[j+2]; new_cur-
                  >link[j]=cur->link[j+2];
            }
            if(type==1)
            {
                  for(j=0;j<2;j++)
                  {
                        new_cur->elements[j]=cur-
                        >elements[j+3]; new_cur-
                        >link[j]=cur->link[j+3];
                  }
                  new_cur-
                  >elements[2]=-1;
                  new_cur-
                  >link[2]=NULL;
                  new_cur->first=cur-
                  >link[2];
            }
            for(j=2;j<5;j++)
            {
                  cur-
                  >elements[j]=
                  -1; cur-
                  >link[j]=NULL
                  ;
            }
            if(new_cur->link[0]!=NULL && new_cur->link[1]!=NULL && new_cur-
>link[2]!=NULL)
            {
                  new_cur->link[0]-
                  >parent=new_cur; new_cur-
                  >link[1]->parent=new_cur;
                  new_cur->first-
                  >parent=new_cur;
                  if(type==0)
                        new_cur->link[2]->parent=new_cur;
            }
            new_cur->level=cur->level;
            root=insert(root,next_key,new_cur->level-
            1,1,new_cur,cur);
      }
      return root;
}

void sequential(node *root)
{
      node
      *cur=root
```

```
        ;
        while(cur
        !=NULL)
        {
                for(int i=0;i<4;i++)
                {
                        if(cur-
                                >elements[
                                i]==-1)
                                break;
                        cout<<"\t"<<cur->elements[i];
                }
                cur=cur->next;
        }
}

Void  main()
{
        int
        ch,key,flag=0
        ,k; clrscr();
        node
        *root=create_node(
        ); node *cur=root;
        for(;;)
        {
                cout<<"1.Insert 2.Search 3.Display 4.Sequential
                access\n"; cout<<"Enter the choice: ";
                cin>>ch;
                switch(ch)
                {
                        case 1: cout<<"Enter key to be inserted: ";
                                cin>>key;
                                root=insert(root,key,cur_level,0,NULL,
                                NULL); break;
                        case 2: cout<<"Enter
                                key: ";
                                cin>>key;
                                int
                                index[10
                                0]; k=0;
                                flag=search(root,key,index,k);
                                if(flag)
                                {
                                        cout<<"Element
                                        found,Path is: ";
                                        for(int i=0;i<k-1;i++)
                                                cout<<" "<<index[i];
                                }
                                else
                                        cout<<"Element not
                                        found\n"; break;
```

```
                    case 3:
                          display(r
                          oot);
                          break;
                    case 4:
                          sequential(
                          cur);
                          break;
                    default:exit(0);
              }
      }
}
```

Output:

```
1.Inser
t a Key
2.Searc
h a key
3.Trave
rse
Leaf
4.Exit
enter u’r choice : 1

Enter The Key to be inserted
in B-Tree 100

1.Insert a Key
2.Searc
h a key
3.Trave
rse
Leaf
4.Exit
enter u’r choice : 1

Enter The Key to be inserted
in B-Tree 50
--------
50 100
--------

1.Inser
t a Key
2.Searc
h a key
3.Trave
rse
Leaf
4.Exit
enter u’r choice : 1
```

```
Enter The Key to be inserted
in B-Tree 200

----------
50 100 200
----------
1.Inser
t a Key
2.Searc
h a key
3.Trave
rse
Leaf
4.Exit
enter u’r choice : 1

Enter The Key to be inserted
in B-Tree 75
---------------
50 75 100 200
---------------

1.Inser
t a Key
2.Searc
h a key
3.Trave
rse
Leaf
4.Exit
enter u’r choice : 2

Enter The key to be
searched 300

The Key Searched for was not found
```

VIVA QUESTIONS

1.What is File Structure?

2. What is a File?

3. What is a field?

4. What is a Record?

5. What is fixed length record?

6. What is RRN?

7. What is Variable length record?

8. What are the different modes of opening a file?

9. What is ifstream()?

10. What is ofstream()?

11. What is the difference between read() and getline()?

12. How to close a file? What happens if a file is not closed?

13. What is Hashing? What is its use?

14. Explain any one collision resolution technique.

15. What is Btree? What is B+tree?

16. Differentiate between Logical and Physical file

17. What is the use of seekg() and seekp()?

18. Explain the different way of write data to a file.

19. Explain the different way of write data to a file.

20. What is meant by Indexing?

21. What is multilevel indexing?

22. What is File descriptor?

23. What is Fragmentation? What is internal fragmentation?

24. What is DMA?

25. What is a delimeter?

26. Define direct access.

27. Define sequential access.

28. What is the need of packing the record before writing into the file?

29. Explain ios::trunk and ios::nocreate

30. What is the use of End-of-file (EOF)?

31. What are stdin, stdout and stderr?

32. What is Fragmentation? 33. What is data compression?

34. What are the properties of B tree?

35. How do we delete fixed length records?

36. How can we reclaim the deleted space dynamically?

37. What are the advantages and disadvantages of indexes that are too large to hold in memory?

38. What is an AVL tree?

39. H M L B Q S T N A Z P E G C V J K D I U Show B tree creation, insertion, splitting, deletion, merging and redistribution.

40. What is memset() ? Explain its parameters.

41. What is sprintf() ? Explain its parameters.

42. What is the use of tellg() and tellp()?

43. What is Boeing tree?