SIR CRR COLLEGE OF ENGINEERING, ELURU

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

FILE STRUCTURES LAB MANUAL(CSE 3.2.7)

PREPARED BY

P.NAGA DEEPTHI (III/IV CSE-A)
K.CHAITANYA DEEPTHI(III/IV CSE-B)
S.MOHAN BABU CHOWDARY(III/IV CSE-C)

FILE STRUCTURES LAB

CSE 3.2.7 **CREDITS:2**

LAB : 3 Periods/week. **Sessional marks : 50**

Univ-Exam: 3hrs. **Univ-Exam-Marks:50**

1.File Operations:

Opening, reading, writing, closing and creating of files in C++

2. Study of secondary storage devices:

Tracks, sectors, block capacity of disk, tape and CDROM

3. File Structures in C++

Reading a stream of fields, record structures and its length indicators, Mixing of numbers and characters, Use of a hex dump, Retrieving records by keys using sequential search, direct access

4. File performance

Data compression, storage compacting, reclaiming space dynamically

5. Indexing and indexed sequential files

Index file, inverted file operations, usage of B and B++ trees

6. Hashing files

Hashing functions, algorithms, record distribution and collision resolution by progressive over flow, Extendable hashing and hashing performance

File Structures Laboratory

Index

S.no	Name of the program	Page No.
MODULE-I		
1.1	Write a program to display content of a file.	5
1.2	Write a program for reading data into a file.	5
1.3	Write a program for creating and inserting data into a file.	5-6
1.4	Write a program for unformatted read and write operations.	7-8
1.5	Write a program for renaming an existing file.	9
1.6	Write a program for deleting an existing file	9-10
1.7	Write a program to copy data from one file to another file	10-11
1.8	Write a program for converting given number into words.	11
1.9	Write a program to check whether the given word is in the file or not.	12
MODULE-II		
2.1	Write a c++ program for calculating size of file.	13
2.2	Write a c++ program for estimating the size of disk drive	14
2.3	Write a c++ program to calculate the capacity of song in bytes.	15
2.4	Write a c++ program to find the transfer time of the file.	15
2.5	Write a c++ program for counting characters , digits, special characters and non data overhead's in a given file.	16-17
MODULE-III		
3.1	Write a c++ program to read student information, qualification using length indicator.	17
3.2	Write a c++ program to read student information using field structure with assigning a key value.	18-19

3.3	Write a c++ program to read student details using length indicator.	19-20
3.4	Write a c++ program to read student details using field structure with delimiter.	20-21
3.5	Write a c++ program for use of a hexdump.	22-23
3.6	Write a c++ program for sequential search.	23-24
3.7	Write a c++ program to retrieve a record from the file using direct access.	24-26
MODULE-IV		
4.1	Write a sequential program for data compression.	26-27
4.2	Write a c++ program for suppressing repeated sequence.	28
MODULE-V		
5	Write a squential program to write and read the data into the file using indexing.	29-30
MODULE-VI		
6.1	Write a hashing program to find hash values.	31-33
6.2	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.	33-43
6.3	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	43-53
6.4	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.	53-59
6.5	Write a C++ program to implement B-Tree for a given set of integers and its operations insert ( ) and search ( ). Display the tree.	59-72
6.6	Write a C++ program to implement B+ tree for a given set of integers and its operations insert ( ), and search ( ). Display the tree.	72-87

Lab Cycle 1.1:

Aim: Write a program to display content of a file.

Program:

```
#include<iostream>
using namespace std;
#include<fstream>
int main()
{
fstream infile;
char name[20];
cout<<"Enter the name of the file:";
cin>>name;
infile.open(name);
char ch;
while(1)
{
infile>>ch;
if(!infile.fail() )
cout<<ch;
else break;
}
return 0;
}
```

Output:

```
   vi display.cpp
    g++ display.cpp
     vi f1.txt
     ./a.out
    Enter the name of the file f1.txt
     sukanya
```

Lab Cycle 1.2:

Aim: Write a program for reading data into a file.

Program:

```
#include<iostream>

 #include<fstream>

using namespace std;
int main()
{
fstream student;
char first[20],last[20],file[20];
int age;
cout<<"Enter the file name:\t ";
cin>>file;
student.open(file);
student>>first>>last>>age;
cout<<"\nFirst name:"<<first<<"\nLast name:"<<last<<"\nAge:"<<age<<"\n";
student.close();
}
```

Output:

```
Vi read.cpp
g++ read.cpp
vi f2.txt
sukanya
parvataneni
20
g++  read.cpp
./a.out
Enter the file name:f2.txt
first name = sukanya
last name = parvataneni
age = 20
```

Lab Cycle 1.3:
Aim: Write a program for creating and inserting data into a file.
Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
char firstname[30],lastname[30],filename[30];
int age;
cout<<"Enter the first name:\t";
cin>>firstname;
cout<<"Enter the last name:\t";
cin>>lastname;
cout<<"Enter the age:\t";
cin>>age;
cout<<"Enter the name of the file name to insert to the data:\t";
cin>>filename;
fstream student(filename,ios::out);
student<<firstname<<"\n"<<lastname<<"\n"<<age<<"\n";
cout<<"\n";
student.close();
}
```

Output:

```
  Vi insert.cpp
  g++ insert.cpp
  ./a.out
   Enter the first name    sukanya
    Enter the last name    parvataneni
     Enter the age     20
  Enter the name of the file to insert the data:  file.txt
   Vi file.txt
     Sukanya
    Parvataneni
    20
```

Lab Cycle 1.4:
Aim: Write a program for unformatted read and write operations.
Program:

```
#include<iostream>
#include<fstream>
#include<stdlib.h>
using namespace std;
class file
{
char c,a;
public:
void read()
{
char filename[20];
cout<<"Enter the file name to be read";
cin>>filename;
fstream fd(filename,ios::in);
while(a!='$')
{
fd>>a;
cout<<a;
}
fd.close();
}
void write()
{
char filename[20];
cout<<"Enter the filename to insert the data:\t";
cin>>filename;
cout<<"\nPress '$' symbol to specify EOF";
fstream fp(filename,ios::out);
while(c!='$')
{
cin>>c;
fp<<c;
}
fp.close();
}};
main()
{
int ch;
file f;
while(1)
{
cout<<"\n1.Read\n2.Write\n3.Exit\n"<<"Enter your choice:\n";
cin>>ch;
switch(ch)
{
case 1:f.read();
break;
case 2:f.write();
break;
case 3:exit(0);
default: cout<<"Invalid Choice\n";
return 0;
}}}
```

Output:

```
   Vi lab14.cpp
    g++ lab14.cpp
     vi f1.txt
     hi hello$
    ./a.out
     1.read
      2. write
       3. exit
  Enter your choice  1
   Enter the file name to be read   f1.txt
   Hi hello$
   1.read
    2.write
     3.exit
    Enter your choice   2
```

```
 Enter the file name to insert the data   f1.txt
Press $ symbol to specify eof    sukanya$
 1.read
 2.write
  3.exit
 Enter your choice  3
```

Lab Cycle 1.5:
Aim: Write a program for renaming an existing file.
Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
int result;
char oldname[]="oldname.txt";
char newname[]="newname.txt";
result=rename(oldname,newname);
cout<<"File successfully renamed\n";
if(result==0)
return 0;
}
```

Output:

```
Vi lab15.cpp
g++  lab15.cpp
vi f1.txt
hii
vi f2.txt
hello
./a.out
File successfully renamed
 Vi f2.txt
  Hii
```

Lab Cycle 1.6:
Aim: Write a program for deleting an existing file
Program:

```
#include<iostream>

#include<fstream>
using namespace std;
int main()
{
```

```
char file[25];
int result;
cout<<"Enter the file name:\t";
cin>>file;
result=remove(file);
if(result==0)
cout<<"File is deleted\n";
else
cout<<"File is not deleted\n";
}
```

Output:

```
   Vi lab16.cpp
   g++ lab16.cpp
   vi f1.txt
    hii
    ./a.out
    Enter the file name  f1.txt
    The file was deleted
    Vi f1.txt
```

Lab Cycle 1.7:
Aim: Write a program to copy data from one file to another file
Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
char c,s[20],d[20];
cout<<"Enter the source file to copy:\t";
cin>>s;
cout<<"\nEnter the destination file to copy:\t";
cin>>d;
fstream fs,fd;
fs.open(s,ios::in);
fd.open(d,ios::out);
while(c!='$')
{
fs>>c;
fd<<c;
cout<<c;
}
cout<<"\nFile is copied";
fs.close();
fd.close();
}
```

Output:

```
 Vi lab17.cpp
  g++ lab17.cpp
  vi f1.txt
  sukanya
  vi f2.txt
  parvataneni
  ./a.out
   Enter the source file to copy
   F1.txt
   Enter the destination file to copy  f2.txt
    Sukanya  file copied
```

Lab Cycle 1.8:
Aim: Write a program for converting given number into words.
Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
char
f[30],a[10][8]={"zero","one","two","three","four","five","six","seven","eight","nine
"};
int no,r,s=0,i=0,g=0,m=0;
cout<<"Enter file name:\t";
cin>>f;
fstream fr(f,ios::out);
cout<<"Enter the number:\t";
cin>>no;
while(no!=0)
{
r=no%10;
s=(s*10)+r;
if(r==0&&s==0)
m=m+1;
no/=10;
}
while(s!=0)
{
r=s%10;
fr<<a[r]<<" ";
cout<<a[r];
s/=10;
}
for(i=0;i<m;i++)
{
fr<<a[g]<<"";
cout<<a[g]<<"";
}
fr<<endl;
fr.close();
return 0;
}
```

Output:

```
Enter the file name  s.txt
Enter the  number  112
One one two
```

Lab Cycle 1.9:

Aim: Write a program to check whether the given word is in the file or not.

Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
char filename[20],w[20],c[20];
cout<<"Enter the name of file:\t";
cin>>filename;
cout<<"Enter the the word to check whether it is present in a given file or not:\t";
cin>>w;
fstream fr(filename,ios::in);
while(!fr.eof())
{
fr>>c;
if(strcmp(c,w)==0)
{
cout<<"Word found\n";
break;
}
else
cout<<"Word not found\n";
break;
}
fr.close();
return 0;
}
```

Output:

```
Enter the name of the file
s.txt
enter the name of the word to check whether it is present or not in given file
 one one two
word found
```

Lab Cycle 2.1:

Aim: Write a c++ program size of file.

Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
char filename[30];
FILE*fp;
cout<<"\nEnter the filename to calculate the size of file:\t";
cin>>filename;
fp=fopen(filename,"r");
fseek(fp,01,ios::end);
cout<<"\nThe size of file is "<<ftell(fp);
}
```

Output:

```
Vi lab21.cpp
g++ lab21.cpp
```

```
vi f1.txt
 sukanya
 enter the file name to calculate the size of the file
  f1.txt
  the size of the file is 9
```

Lab Cycle 2.2:

Aim: Write a c++ program for estimating the size of disk drive.

Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
int bps,spt,tpc,cy;
unsigned long int tc,cc,dc;
cout<<"\nEnter the number of bytes per sector:\t";
cin>>bps;
cout<<"\nEnter the number of sectors per track:\t";
cin>>spt;
cout<<"\nEnter the number of tracks per cylinder:\t";
cin>>tpc;
cout<<"\nEnter the number of cylinders:\t";
cin>>cy;
tc=bps*spt;
cc=tc*tpc;
dc=cc*cy;
cout<<"\nDisk capacity in bytes:"<<dc<<" bytes"<<" and in Kilo
Bytes:"<<dc/1024<<" KB"<<"\n";
}
```

Output:

```
 Enter number of bytes per sector 1000
 Enter the number of sectors per track  50
 Enter the number of tracks per cylinder  25
Enter the number of cylinders   15
 Disk capacity is 18750000 bytes
Disk capacity is 18310 kilobytes
```

Lab Cycle 2.3:
Aim: Write a c++ program to calculate the capacity of song in bytes.
Program:
```
#include<iostream>
using namespace std;
int main()
{
long int l=44100,t,time;
cout<<"\nEnter the time taken for a song table played in minutes:";
cin>>t;
t=3*t*60;
l=(t*l)/1024;
cout<<"\n Bytes occupied by the song in mega bytes is:"<<l/1024<<"\n";
}
```

Output:
```
Enter the time taken for a song to be played in minutes  5
Bytes occupied for the song in megabytes  37
```

Lab Cycle 2.4:
Aim: Write a c++ program to find the transfer time of the file.
Program:
```
#include<iostream>
using namespace std;
int main()
{
int rt=12;
long int nbt;
float tt,tc;
cout<<"\nEnter the number of bytes you want to transfer:";
cin>>nbt;
tc=63*512;
tt=nbt/tc*rt;
cout<<"\nTransfer time for "<<nbt<<"bytes to hold is :"<<tt<<"msec"<<"\n";
}
```

Output:
```
Enter the number of bytes you want to transfer  10000
Transfer time for 10000 bytes you hold is 3.72024 msec
```

Lab Cycle 2.5:
Aim: Write a c++ program for counting characters , digits, special characters and non data overhead’s in a given file.

Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
int uc=0,lc=0,d=0,sc=0,count=0;
char filename[20],ch;
cout<<"\nEnter the filename:";
cin>>filename;
ifstream fr1(filename,ios::in);
while(!fr1.eof())
{
fr1>>ch;
count++;
}
fr1.close();
ifstream fr(filename,ios::in);
while(!fr.eof())
{
if(count==1)
break;
fr>>ch;
if(ch>=65&&ch<=90)
{
uc++;
count--;
}
else if(ch>=97&&ch<=122)
{
lc++;
count--;
}
else if(ch>=48&&ch<=57)
{
d++;
count--;
}
else
{
sc++;
count--;
}}
fr.close();
cout<<"\nThe number of upperclass letters:\t"<<uc;
cout<<"\nThe number of lowerclass letters:\t"<<lc;
cout<<"\nThe number of digits are:\t"<<d;
cout<<"\nThe number of special characters are:\t"<<sc<<"\n";
}
```

Output:

```
Vi lab2.5.cpp
g++ lab2.5.cpp
vi p.txt
abcd ABCD 1234$@&%
./a.out
Enter the file name p.txt
The number of upper case letters 4
The number of lower case letters 4
The number of digits are 4
The number of special characters 4
```

Lab Cycle 3.1:

Aim: Write a c++ program to read student information, qualification using length indicator.

Program:

```
#include<iostream>
#include<fstream>

using namespace std;
int main()
{
struct student
{
char name[20],qual[30],city[20];
}s;
FILE *fp;
char filename[20],ch='y';
cout<<"Enter file name:\t";
cin>>filename;
fp=fopen(filename,"w");
while(ch=='y')
{
cout<<"Enter details:\n";
cin>>s.name>>s.qual>>s.city;
fprintf(fp,"%10s%10s%10s",s.name,s.qual,s.city);
cout<<"Press 'y' to continue";
cin>>ch;
}
fclose(fp);
}
```

Output:

```
Vi lab3.1.cpp
g++ lab3.1.cpp
./a.out
Enter file name s.txt
Enter details  sukanya
B tech
Eluru
Press y to continue n
Vi s.txt
Sukanya  btech eluru
```

Lab Cycle 3.2:

Aim: Write a c++ program to read student information using field structure with assigning a key value.

Program:

```
#include<iostream>
#include<fstream>

using namespace std;
int main()
{
struct student
{
char name[30],qual[30],city[20];
}s;
FILE *fp;
char f[30],ch='y';
cout<<"Enter file name:\t";
cin>>f;
fp=fopen(f,"w");
while(ch=='y')
{
cout<<"Enter details:\t";
cin>>s.name>>s.qual>>s.city;
fprintf (fp,"Name=%s\nQualification=%s\nCity=%s\n",s.name,s.qual,s.city);
cout<<"Press y to continue";
cin>>ch;
}
fclose(fp);
}
```

Output:

```
Vi lab3.2.cpp
g++  lab3.2.cpp
./a.out
Enter the file name s.txt
Enter details   sukanya
Btech
Eluru
Press y to continue n
Vi s.txt
Name= sukanya  qual= btech  city= eluru
```

Lab Cycle 3.3:

Aim: Write a c++ program to read student details using length indicator.

Program:

```
#include<iostream>
#include<fstream>
#include<string.h>
using namespace std;
int main()
{
struct student
{
char name[30],qual[30],city[30];
}s;
FILE *fp;
char f[30],ch='y';
cout<<"Enter file name:\t";
cin>>f;
fp=fopen(f,"w+");
while(ch=='y')
{
cout<<"Enter details:\n";
cin>>s.name>>s.qual>>s.city;
fprintf(fp,"%d%s\t%d%s\t%d%s\t", strlen(s.name),s.name,strlen(s.qual),
s.qual,strlen(s.city),s.city);
cout<<"Press 'Y' to continue";
cin>>ch;
}
fclose(fp);
}
```

Output:

```
Vi lab3.3.cpp
g++ lab3.3.cpp
./a.out
Enter the file name  s.txt
Enter details  sukanya
Btech
Eluru
Press y to continue  n
Vi s.txt
7Sukanya 5btech 5eluru
```

Lab Cycle 3.4:

Aim: Write a c++ program to read student details using field structure with delimiter.

Program:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{
```

```
struct student
{
char name[30],qual[15],city[20];
}s;
FILE *fp;
char filename[30],ch='y';
cout<<"Enter filename:\t";
cin>>filename;
fp=fopen(filename,"w");
while(ch=='y')
{
cout<<"Enter the student details:\t";
cin>>s.name>>s.qual>>s.city;
fprintf(fp,"%s|%s|%s",s.name,s.qual,s.city);
cout<<"Press 'y' to continue";
cin>>ch;
}
fclose(fp);
}
```

Output:

```
Vi lab3.4.cpp
g++ lab3.3.cpp
./a.out
enter file name s.txt
enter details sukanya
btech
eluru
press y to continue  n
vi s.txt
sukanyabtecheluru
```

Lab Cycle 3.5:
Aim: Write a c++ program for use of a hexdump.
Program:

```
#include<iostream>
#include<fstream>
#include<iomanip>
using namespace std;
int main()
{
unsigned long address=0;
char c;
cout<<hex<<setfill('0');
while(cin.good())
{
int nread;
char buf[16];
for(nread=0;nread<16&&cin.get(buf[nread]);nread++)
if(nread==0)
cout<<setw(8)<<address;
for(int i=0;i<16;i++)
{
if(i%8==0)
cout<<' ';
if(i<nread)
cout<<' '<<setw(2)<<(unsigned)buf[i];
else
cout<<" ";
}
cout<<"";
for(int i=0;i<nread;i++)
{
if(buf[i]<32) cout<<':';
else cout<<buf[i];
}
cout<<"\n";
address+=16;
}
}
```

Output:

```
Vi lab3.5.cpp
g++ lab3.5.cpp
./a.out
Abcdefgh
0000 61 62 63 64 65 66 67 68 abcdefgh
0008 abcdefgh
Oa 61 62 63 64 65 66 67: abcdefg
0010 abcdefgh
68 0a 61 62 63 64 65 66h: abcdef
0018 abcdefgh
67 68 0a 61 62 63 64 65 gh: abcde
```

Lab Cycle 3.6:

Aim: Write a c++ program for sequential search.

Program:

```
#include<iostream>
using namespace std;
int main()
{
int arr1[10],i,req,location=5;
cout<<"Enter 5 numbers to store in array:\t"<<endl;
for(i=0;i<5;i++)
{
cin>>arr1[i];
}
cout<<endl;
cout<<"Enter number you want to find:\t";
cin>>req;
cout<<endl;
for(int w=0;w<5;w++)
{
if(arr1[w]==req)
location=w;
}
if(location!=5)
{
cout<<"Required number is found at location: "<<location+1<<"\n";
}
else
cout<<"Number is not found\n";
}
```

Output:

```
Vi lab3.6.cpp
g++ lab3.6.cpp
./a.out
Enter five numbers to sort in array
2 3 4 6 8
Enter number you want to found  4
Required number is found at location 3
```

Lab Cycle 3.7:

Aim: Write a c++ program to retrieve a record from the file using direct access.

Program:

```
#include<iostream>
#include<stdio.h>
#include<fstream>
#include<stdlib.h>
using namespace std;
class Emp
{
public:
int Eno;
char Ename[30];
};
int main()
{
FILE *fp;
Emp e;
int ch,count,no;
long int pos;
cout<<"\n1.Addrecord\n2.Direct Access\n3.Exit\n";
do
{
cout<<"\nEnter your choice:\n";
cin>>ch;
switch(ch)
{
case 1:fp=fopen("d1.txt","rb+");
if(!fp)
{
fp=fopen("d1.txt","wb+");
if(!fp)
{
cout<<"\nSorry Insufficient Space";
}}
count=0;
while(fread(&e,sizeof(e),1,fp))
count++;
e.Eno=++count;
cout<<"\nEmployee number:"<<e.Eno;
cout<<"\nEnter name:";
cin>>e.Ename;
fwrite(&e,sizeof(e),1,fp);
fclose(fp);
break;
exit (1);
case 2:fp=fopen("d1.txt","rb+");
cout<<"\nEnter a EMP number for direct access:";
cin>>no;
count=0;
while(fread(&e,sizeof(e),1,fp))count++;
if(no>count)
{
cout<<"\nInvalid EMP number";
break;
}
pos=(no-1)*sizeof(e);
fseek(fp,pos,0);
fread(&e,sizeof(e),1,fp);
cout<<e.Eno<<e.Ename;
break;
default:
cout<<"\nAborting from the execution\n";
}}
while(ch!=3);
return (0);
}
```

Output:

```
1.addrecords
2.direct access
3.exit
Enter the choice 1
Employee number:1.enter name  sukanya
Enter your choice 1
Employee number : 2  enter the name ramya
Enter the choice 2
 Enter the employee number for direct access: 2
2 ramya
Enter the choice 3
Aborting from the execution
```

Lab Cycle 4.1:
Aim: Write a sequential program for data compression.
Program:

```
#include<iostream>
#include<stdio.h>
#include<string.h>
using namespace std;
class stud
{
public:
int no,g;
char name[20];
};
int main()
{
FILE *fp;
stud s;
int i=0;
char ch,ge[8];
fp=fopen("stu.txt","wt");
cout<<"\nInput";
do
{
cout<<"\nEnter details of the student:";
s.no=++i;
cout<<"\nEnter name:\t";
cin>>s.name;
cout<<"\nEnter gender:\t";
cin>>ge;
if(strcmp(ge,"male")==0)
s.g=0;
else
s.g=1;
fwrite(&s,sizeof(s),1,fp);
cout<<"\nDo you want to continue:\t";
fflush(stdin);
cin>>ch;
} while(ch=='1'||ch=='y');
fclose(fp);
fp=fopen("stu.txt","rt");
fseek(fp,0,0);
cout<<"\nNo\tName\tGender\n";
while(fread(&s,sizeof(s),1,fp))
cout<<s.no<<"\t"<<s.name<<"\t"<<s.g<<"\n";
fclose(fp);
return 0;
}
```

Output:

```
Input:
Enter details of the student
Enter name sukanya
Enter gender
Female
Do you want to continue n
No.    name       gender
1      sukanya      1
```

Lab Cycle 4.2:
Aim: Write a c++ program for suppressing repeated sequence.
Program:

```
#include<iostream>
using namespace std;
int main()
{

int a[14],n,i,count;
cout<<"Enter no. of values:\t";
cin>>n;
cout<<"\nEnter average values:\t";
for(i=0;i<n;i++)
cin>>a[i];
for(i=0;i<n;i++)
{
if(a[i]==a[i+1])
{
count=1;
while(a[i]==a[i+1]&&i<n)
{
count++;
i++;
}
cout<<" ff "<<a[i-1]<<" "<<count;
}
else
cout<<" "<<a[i];
}
cout<<"\n";
}
```

Output:

```
Enter number of values  6
Enter average values   1
1 2 2 3 3
Ff 1 2ff 2  2ff  3 2
```

Lab Cycle 5:

Aim: Write a squential program to write and read the data into the file using indexing.

Program:

```
#include<iostream>
#include<stdio.h>
using namespace std;
class emp
{
public:
int eno;
char ename[20];
float salary;
};
class ind
{
public:
int key;
int no;
};
int main()
{
emp e1;
ind i1;
int k;
char ch;
FILE *fp,*fs;
fp=fopen("data1.doc","w");
fs=fopen("data2.doc","w");
do
{
cout <<"\nEnter Eno, Ename, Esalary:\t";
cin>>e1.eno>>e1.ename>>e1.salary;
fwrite(&e1,sizeof(e1),1,fp);
i1.key=e1.eno;
i1.no=(int)ftell(fp)/sizeof(e1);
fwrite(&i1,sizeof(i1),1,fs);
fflush(stdin);
cout<<"\nAnother record ? press Y :\t ";
cin>>ch;
}
while(ch=='y'||ch=='Y');
fcloseall();
fp=fopen("data1.doc","r");
fs=fopen("data2.doc","r");
while(1)
{
cout<<"\nEnter Eno record to retrieve : \t";
cout<<"\nEnter 0 to quit :\t ";
fflush(stdin);
cin>>k;
if(k==0)
break;
rewind(fp);
rewind(fs);
while(fread(&i1,sizeof(i1),1,fs))
if(i1.key==k)
{
fseek(fp,(i1.no-1)*sizeof(e1),0);
fread(&e1,sizeof(e1),1,fp);
cout<<"\nEno,Ename,Esalary:\n";
cout<<e1.eno<<"\n"<<e1.ename<<"\n"<<e1.salary<<"\n";
break;
}
if(i1.key!=k)
cout<<"Record not found\n";
}
fcloseall();
return 0;
}
```

Output:

```
Enter eno., Ename,salary  1 sukanya  50000
Another record ?  press y y
Enter eno,ename,salary 2 ramya  50000
Another record?  Press y N

Enter eno  record  to retrieve
Enter 0 to quit  0
```

Lab Cycle 6.1:
Aim: Write a hashing program to find hash values.
Program:
```
#include<stdio.h>
#include<iostream>
#include<string.h>
using namespace std;
class emp
{
public:
int no,sal;
char name[20];
};
class hash
{
public:
char no;
int key;
};
FILE *fp,*fs;
int os;
emp e;
hash h;
int main()
{
int found;
char ch='y';
int search(int);
int hash(char a[]);
do
{
cout<<"\nInput:";
fp=fopen("emp.txt","ab+");
fs=fopen("hash.txt","ab+");
cout<<"\nEnter no,salary,name:\t";
cin>>e.no>>e.sal>>e.name;
os=hash(e.name);
found=0;
while(fread(&h,sizeof(h),1,fs))
if(h.no==os)
{
```

```
found=1;
break;
}
if(found==1)
os=search(os);
h.no=os;
h.key=e.no;
fwrite(&h,sizeof(h),1,fs);
fseek(fp,(os-1)*24,0);
fwrite(&e,sizeof(e),1,fp);
cout<<"\nDo you want to continue (Y/N) :\t";
fflush(stdin);
cin>>ch;
}while(ch=='y'||ch=='Y');
fcloseall();
return 0;
}
int hash(char name[])
{
int i=0,sum,n;
sum=(name[i]*name[i+1]);
cout<<"\nSum: "<<sum;
n=sum%19;
cout<<"\nHash Values is :\t"<<n;
return(n);
}
int search(int cur)
{
int i,j;
i=cur+1;
fseek(fs,(cur-1)*20,0);
while(fread(&h,sizeof(h),1,fs))
if(h.no==i)
i++;
return(i);
}
```

Output:

```
Input:
Enter no,salary,name    1 5000 sukanya
Sum    3455
Hash value is : 3
Continue(y/n)  n
```

Labcycle 6.2:

Aim: 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.

Program:

```
#include<iostream.h>

#include<fstream.h>

#include<process.h>

#include<string.h>

#include<conio.h
> class student

{

        private:

                char buf[45],name[10],sem[10],branch[10];
        public:

                void read()

                {
```

```
          cout<<"Name: "<<endl;
          cin>>name;
          cout<<"Semester:
          "<<endl; cin>>sem;

          cout<<"Branch:
          "<<endl; cin>>branch;

}


void pack(fstream &ofile)

{

          read();

          strcpy(buf,"");

          strcat(buf,name);

          strcat(buf,"|");

          strcat(buf,sem);

          strcat(buf,"|");

          strcat(buf,branch);

          strcat(buf,"|");

          while(strlen(buf)<45)

                    strcat(buf,"!");

          strcat(buf,"\n");
```

```
        ofile.write(buf,strlen(buf));

}


void  unpack(fstream &ifile)

{

        char extra[45];
        while(!ifile.eof())

        {

                ifile.getline(name,10,'|');

                ifile.getline(sem,10,'|');

                        ifile.getline(branch,10,'|');


                        ifile.getline(extra,45,'\n');

                cout<<name<<"\t"<<sem<<"\t"<<branch<<"\n";

        }

}


int search(fstream &ifile,char key[])

{

        char extra[45];
        while(!ifile.eof())
        {

            ifile.getline(name,10,'|');

            ifile.getline(sem,10,'|');

            ifile.getline(branch,10,'|');

            ifile.getline(extra,45,'\n');

            if(strcmp(name,key)==0)

            {

                cout<<"Record found and details are:"<<endl;
                cout<<"Name: "<<name<<endl;
                cout<<"Semester: "<<sem<<endl;
                cout<<"Branch: "<<branch<<endl;

                return 1;

            }

        }

        return 0;

}


void modify(fstream &iofile,char key[])

{

        if(search(iofile,key))

        {
```

```
                    cout<<"Record found,enter modification details:"<<endl;
                    iofile.seekp(-47,ios::cur);

                    pack(iofile);

               }

               else

                    cout<<"Sorry!No such record\n";

          }

};


void main()

{

     int n,i,ch;
     student stu;
     fstream ofile;

     ofile.open("student.txt",ios::trunc|ios::app);
     ofile.close();

     clrscr();

     for(;;)

     {

          clrscr();
```

```
cout<<"1. Insert\n2. Display  all\n3. Search\n4. Modify\n5.Exit\n";

cout<<"Enter your choice"<<endl;
cin>>ch;

switch(ch)

{

        case 1: fstream ofile;
                ofile.open("student.txt",ios::out|ios::app);
                cout<<"Enter the no. of students"<<endl;
                cin>>n;

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

                {

                        stu.pack(ofile);

                }

                ofile.close();

                break;


        case 2: fstream infile;
                infile.open("student.txt",ios::in);
                stu.unpack(infile);

                getch();

                infile.close();

                break;
}

        case 3: cout<<"Enter the record name to be searched"<<endl;
                char key[10];


                cin>>key;
                fstream ifile;


                ifile.open("student.txt",ios::in);


                if(stu.search(ifile,key)==0)
                        cout<<"record not found\n";


                getch();


                ifile.close();


                break;



        case 4: fstream iofile;
                iofile.open("student.txt",ios::in|ios::out); cout<<"Enter
                the record name to be modified"<<endl; cin>>key;


                stu.modify(iofile,key);

                getch();


                iofile.close();


                break;
        default: exit(0);


}


}


}
```

Output :

```
1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the choice:1

Enter the number of students:2


Enter the student name = ajay

Enter the sem = 6

Enter the branch = ise


Enter the student name = rahul

Enter the sem = 6

Enter the branch = cse



1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the choice:2
Name            Sem         Branch
ajay            6           ise|||||||||||
rahul           6           cse|||||||||||
1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the choice:4

Enter the record name you want to search =
rahul Record found

rahul         6               cse||||||||||||



1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the choice:3

Enter the record name you want to
modify:rahul record found and details are:


rahul          6              cse|||||||||||

enter  modification details


                      name
Enter the student  =navya

Enter the sem = 6

Enter the branch = ise

1:write to file 2:display the  file 3:modify the file 4:search 5.exit

Enter the choice:2

Name                Sem           Branch

ajay                6             ise|||||||||||

Navya               6             ise|||||||||||
1:write to file 2:display the file 3:modify the file 4:search 5.exit
Enter the choice:4

Enter the record name you want to search:keerthi

Record not found
```

Labcycle 6.3:

Aim: 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

PROGRAM:

```
#include<iostream.h>

#include<fstream.h>

#include<process.h>

#include<string.h>

#include<conio.
h> class student

{

        private: char
        buf[45],name[10],sem[10],branch[10]; int pos;
```

```
public:
void
read()

{

        cout<<"name:"<<endl;

        cin>>name;

        cout<<"semester:"<<endl;

        cin>>sem;

        cout<<"branch:"<<endl;

        cin>>branch;

}


void pack(fstream &ofile)

 {

        read();

        strcpy(buf,"");

        strcat(buf,name);

        strcat(buf,"|");

        strcat(buf,sem);

        strcat(buf,"|");
        strcat(buf,branch);

        strcat(buf,"|");

        strcat(buf,"\n");

        ofile.write(buf,strlen(buf));

}


void unpack(fstream &ifile)

{

        char extra[45];
        while(!ifile.eof())

        {

                ifile.getline(name,10,'|');

                ifile.getline(sem,10,'|');

                ifile.getline(branch,10,'|');

                ifile.getline(extra,45,'\n');

                cout<<name<<"\t";

                cout<<sem<<"\t";

                cout<<branch<<"\n";

        }

}
```

```
int search(fstream &ifile,char key[])

{

    char extra[45];
    while(!ifile.eof())

    {

        ifile.getline(name,10,'|');

        ifile.getline(sem,10,'|');

        ifile.getline(branch,10,'|');

        ifile.getline(extra,45,'\n');


         if(strcmp(name,key)==0)

         {

             cout<<" "<<"record found and details
             are:"<<endl; cout<<"
             "<<"name"<<name<<endl;


            cout<<"
             "<<"semester"<<sem<<endl;
             cout<<"
             "<<"branch"<<branch<<endl;
             return 1;


         }
        }

          return 0;

}

void modify(fstream &ifile,char key[])

{

        student s[10];
        char extra[50];
        int i=0;

          while(!ifile.eof())

          {

                ifile.getline(s[i].name,10,'|');

                ifile.getline(s[i].sem,10,'|');

                ifile.getline(s[i].branch,10,'|');

                ifile.getline(extra,45,'\n');

                i++;

          }

        ifile.close();
        int flag=0;

        for(int j=0;j<i;j++)
```

```
{

        if(strcmp(key,s[j].name)==0)

        {

                flag=1;

                cout<<"record found details
                are:"<<endl; cout<<s[j].name<<endl;
                cout<<s[j].sem<<endl;
                cout<<s[j].branch<<endl;

                cout<<"enter the modification
                details"<<endl; cout<<"enetr the
                name"<<endl;

                cin>>s[j].name;

                cout<<"enter the
                sem;"<<endl; cin>>s[j].sem;

                cout<<"enter the
                branch"<<endl;
                cin>>s[j].branch;

        }

}

if(flag==0)

{

        cout<<"Record not
        found\n"; return;
```

```
                }

                ifile.open("student.txt",ios::trunc|ios::app);
                for(int k=0;k<i;k++)

                {

                        strcpy(buf,"");

                        strcat(buf,s[k].name);

                        strcat(buf,"|");

                        strcat(buf,s[k].sem);

                        strcat(buf,"|");

                        strcat(buf,s[k].branch);

                        strcat(buf,"|");

                        strcat(buf,"\n");

                        ifile.write(buf,strlen(buf));

                }

        }

};

void main()

{
```

```
int n,i,ch;
char
key[10];
student stu;

fstream ifile,ofile;
ofile.open("student.txt",ios::trunc|ios::app);
ofile.close();

for(;;)

{

        clrscr();

        cout<<"1.insert\n 2.display\n 3.search\n 4.modify\n 5.exit\n";
        cout<<"enter your choice"<<endl;

        cin>>ch;

        switch(ch)

        {

                case 1: fstream ofile;
                        ofile.open("student.txt",ios::out|ios::app);
                        cout<<"enter the no of students";

                        cin>>n;


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

                        {

                                stu.pack(ofile);
```

```
            }

            ofile.close();

            break;


      case 2: fstream infile;
              infile.open("student.txt",ios::in);
              stu.unpack(infile);

              getch();

              infile.close();

              break;


      case 3:cout<<"enter the record name to be
              searched"<<endl; cin>>key;

              fstream ifile;
              ifile.open("student.txt",ios::in);
              if(stu.search(ifile,key)==0)
              cout<<"record not found\n";
              getch();

              ifile.close();

              break;


      case 4: fstream iofile;
              iofile.open("student.txt",ios::in|ios::out);
              cout<<"enter the record name to be
              modified\n"<<endl; cin>>key;

              stu.modify(iofile,key);
```

```
                    getch();

                    iofile.close();

                    break;

                    default:exit(0);

                }

            }

        }
```

Output:

```
1:write to file 2:display the file 3:modify the file 4:search 5.exit
Enter the choice:1

Enter the number of
students:2 Enter the student
name = ajay Enter the sem =
6

Enter the branch = ise


Enter the student name = rahul

Enter the sem = 6

Enter the branch = cse


1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the choice:2
```

```
Name                 Sem          Branc
                                  h
                                           1:write to file 2:display the
Ajay                 6            ise      file 3:modify the file 4:search
Rahul                6            cse      5.exit Enter the choice:4

Enter the record name you want to search =
rahul Record found

rahul          6              cse


1:write to file 2:display the file 3:modify the file 4:search 5.exit
Enter the choice:3

Enter the record name you want to
modify:rahul record found and details are:

rahul          6              cse


enter modification details
Enter the student name
=navya Enter the sem = 6

Enter the branch = ise

1:write to file 2:display the file 3:modify the file 4:search 5.exit

Enter the
choice:2 Name

ajay
Nav
ya

1:write to file 2:display the file 3:modify the file 4:search 5.exit
```

```
Enter the choice:4

Enter the record name you want to
search:keerthi Record not found
```

Labcycle 6.4:

AIM: 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.

Program:

```
#include<stdio.h>

#include<conio.h>

#include<string.h>

#include<stdlib.h>

#include<iostream.h>

#include<fstream.h>


class student

{

        private:

                char
        buf[40],name[10],sem[10],branch[10],extra[40]; public:

                void read()
```

```
{

        cout<<"Name: "<<endl;
        cin>>name;
        cout<<"Semester:
        "<<endl; cin>>sem;

        cout<<"Branch:
        "<<endl; cin>>branch;

}


void insert(fstream &ofile,char rrn[])

{

        read();

        strcpy(buf,"");

        strcat(buf,rrn);

        strcat(buf,"|");

        strcat(buf,name);

        strcat(buf,"|");

        strcat(buf,sem);

        strcat(buf,"|");

        strcat(buf,branch);

        strcat(buf,"|");
        strcat(buf,"\n");

        ofile.write(buf,strlen(buf));

}


int search(fstream &ifile,char key[])

{

        char rrn[10];
        while(!ifile.eof())

        {

                ifile.getline(rrn,10,'|');

                ifile.getline(name,10,'|');

                ifile.getline(sem,10,'|');

                ifile.getline(branch,10,'|');

                ifile.getline(extra,40,'\n');
                        if(strcmp(key,rrn)==0)

                {

                        cout<<"Record found and details
                        are:"<<endl; cout<<"Name: "<<name;

                        cout<<"Semester:
                        "<<sem;
                        cout<<"Branch:
                        "<<branch; return 1;
```

```
                    }

                }

                return 0;

            }

};


void main()

{

    int n,i,ch,k=0;
    char key[10];
    student stu;
    fstream ofile;

    ofile.open("student2.txt",ios::trunc|ios::app);

    ofile.close();

    clrscr();

    for(;;)

    {

        cout<<"1.Insert\n2.Search\n3.Exit\n
        "; cout<<"Enter your choice: ";

        cin>>ch;

        switch(ch)
```

```
{

        case 1: fstream ofile;
                ofile.open("student2.txt",ios::out|ios::app);
                cout<<"Enter the no. of students: ";

                cin>>n;

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

                {

                        itoa(++k,key,10);

                        stu.insert(ofile,key);

                }

                ofile.close();

                break;

        case 2: cout<<"Enter the RRN to search:
                "; cin>>key;

                fstream ifile;
                ifile.open("student.txt",ios::in);
                if(stu.search(ifile,key)==0)
                cout<<"Record not found\n";
                ifile.close();

                break;

        default:exit(0);

}
```

```
          }

}
```

Output:

```
1.Insert

2.Search

3.Exit

Enetr your choice:1

Enter the no. of students:2
name = ajay

sem = 6
branch =
ise


name =
rahul sem =
6 branch =
cse


1.Insert

2.Search

3.Exit

Enetr your choice:2

Enter the RRN to search:1
```

```
Record found and details
are:"<< rahul 6 cse 1.Insert


2.Search

3.Exit

Enetr your choice:2

Enter the RRN to
search:5 Record not
found
```

Lab cycle 6.5:

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

Program:

```
#include<iostream.h>

#include<stdio.h>

#include<fstream.h>

#include<stdlib.h>

#include<string.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]=NUL
                L;
                parent=NUL
                L; size=0;

}


class btree

{

        node
        *root;
        public:

                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=NULL;
        level=0;

        int i;
        while(ptr)

        {

                i=0;

                level++;

                while(i<ptr->size-1 && key>ptr-
                >a[i]) i++;

                prevptr=ptr;
                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<midpoint;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++)

{
    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) return;


        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]=key;


        n-
        >next[i]=address;
        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=new
                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,level); 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,key;
     while(choice<=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.Insert        a
Key  2.Search
a key 3.Exit

Enter u’r choice :1


Enter The Key to be inserted in B-
Tree 100



1.Insert        a
Key  2.Search
a key 3.Exit

Enter u’r choice :1
Enter The Key to be inserted in B-
Tree 50

-----------

50 100

-----------

1.Insert        a
Key  2.Search
a key 3.Exit

Enter u'r choice :1


Enter The Key to be inserted in B-
Tree 75

-------------

50 75 100

-------------

1.Insert        a
Key  2.Search
a key 3.Exit

Enter u'r choice :1


Enter The Key to be inserted in B-
Tree 200

---------------
```

```
50 75 100 200

---------------


1.Insert        a
Key  2.Search
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
```

Labcycle 6.6:

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

Program:

```
    #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=NULL; cur->level=0; cur->first=NULL;

return cur;

}


node* search(node *root,int key,int depth)

{

    node
    *cur=root;
    if(depth==0)

        return root;
    int i=0,j;
    while(i<depth)

    {

        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)

                return 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)

                flag=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]=key;
            cur->link[j]=child1;
            if(child1!=NULL)

                child1->parent=cur;

            break;

        }

    }

    if(j==0)

    {

        cur-
        >elements[j]=key;
        cur->link[j]=child1;
        cur->first=child2;
        if(child1!=NULL)

            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!=NU
        LL)

        {

                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,NU
                        LL); break;

                case 2: cout<<"Enter key:
                        "; cin>>key;

                        int index[100];
                        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(root);
                              break;

                        case 4: sequential(cur);
                              break;

                        default:exit(0);

                  }

            }

      }
```

Output:

```
1.Insert a Key 2.Search a key 3.Traverse Leaf 4.Exit

enter u'r choice : 1

Enter The Key to be inserted in B-Tree 100

1.Insert a Key
```

```
2.Search a key 3.Traverse Leaf 4.Exit

enter u'r choice : 1

Enter The Key to be inserted in B-Tree 50

--------

50 100

--------

1.Insert a Key 2.Search a key 3.Traverse Leaf 4.Exit

enter u'r choice : 1

Enter The Key to be inserted in B-Tree 200

----------

50 100 200

----------

1.Insert a Key 2.Search a key 3.Traverse Leaf 4.Exit

enter u'r choice : 1

Enter The Key to be inserted in B-Tree 75

---------------

50 75 100 200

---------------

1.Insert a Key 2.Search a key 3.Traverse Leaf 4.Exit

enter u'r choice : 2

Enter The key to be searched 300

The Key Searched for was not found
```