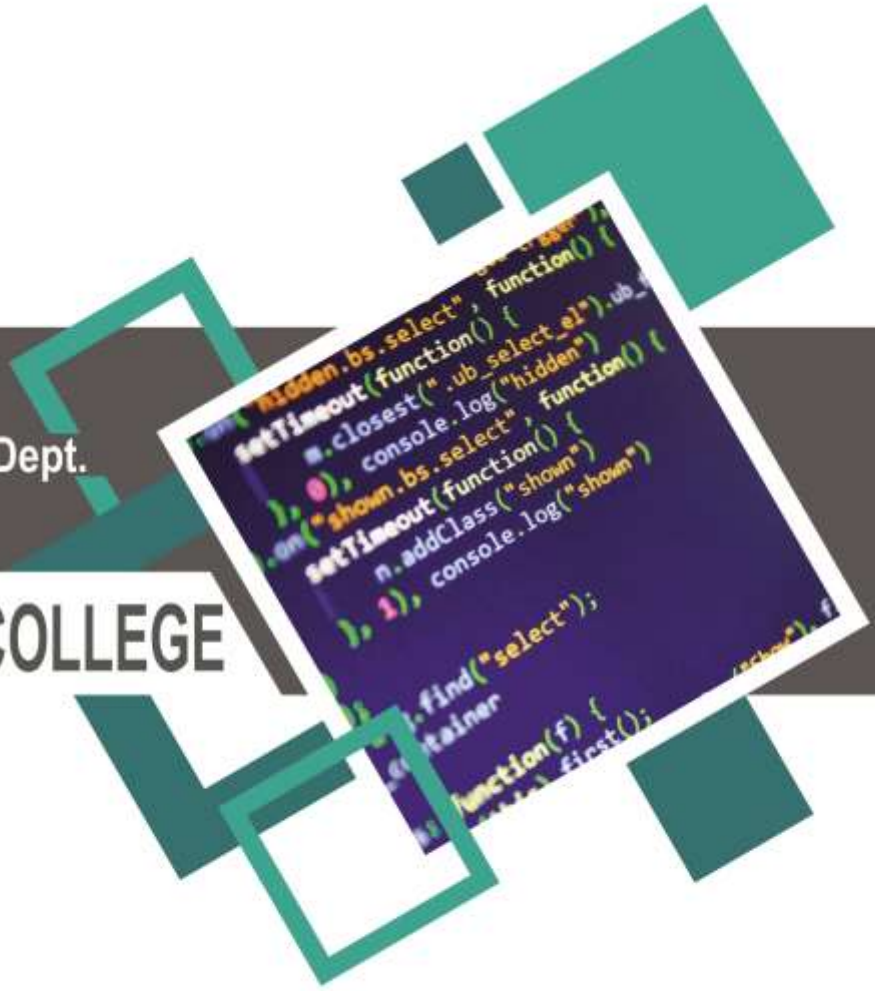
BACHELOR OF COMPUTER APPLICATION LAB MANUAL

4th Semester

Prepared By

Pure and Applied Science Dept.

Computer Application



MIDNAPORE CITY COLLEGE

C++ LABORATORY MANUAL
(Course Code: BCA-2296)

INSTRUCTIONS TO STUDENTS

- Before entering the lab, the student should carry the following things (MANDATORY)

    1. Identity card issued by the college.
    2. Class notes
    3. Lab observation book
    4. Lab Manual
    5. Lab Record

- Student must sign in and sign out in the register provided when attending the lab session without fail.
- Come to the laboratory in time. Students, who are late more than 10 min., will not be allowed to attend the lab.
- Students need to maintain 80% attendance in lab if not a strict action will be taken.
- All students must follow a Dress Code while in the laboratory.
- Foods, drinks are NOT allowed.
- All bags must be left at the indicated place.
- Refer to the lab staff if you need any help in using the lab.
- Respect the laboratory and its other users.
- Workspace must be kept clean and tidy after experiment is completed.
- Read the Manual carefully before coming to the laboratory and be sure about what you are supposed to do.
- Do the experiments as per the instructions given in the manual.
- Copy all the programs to observation which are taught in class before attending the lab session.
- Students are not supposed to use floppy disks, pen drives without permission of lab- in charge.
- Lab records need to be submitted on or before the date of submission.

1. ***Write a C++ Program to display Names, Roll No., and grades of 3 students who have appeared in the examination. Declare the class of name, Roll No. and grade. Create an array of class objects. Read and display the contents of the array.***

Program:

```
#include <iostream>
 using namespace std;
 #define MAX 10

class student
{
private:
char name[30]; int rollNo;
int total; float perc;
public:
void getDetails(void);          //member function to get student's details
 void putDetails(void);         //member function to print student's details
};

void student:: getDetails(void)          //member function definition, outside of the class
{
cout << "Enter name: " ; cin >> name;
cout << "Enter roll number: "; cin >> rollNo;
cout << "Enter total marks outof 500: "; cin >> total;
perc=(float)total/500*100;
}

void student:: putDetails(void)          //member function definition, outside of the class
{
cout << "Student details:\n";
cout << "Name:"<< name << ",Roll Number:" << rollNo << ",Total:" << total <<
",Percentage:" << perc;

}

int main()
{
student std[MAX];     //array of objects creation int n,loop;
cout << "Enter total number of students: "; cin >> n;
for (loop=0;loop< n; loop++)
{
cout << "Enter details of student " << loop+1 << ":\n"; std[loop].getDetails();
}
```

```
cout << endl; for(loop=0;loop< n; loop++)
{
cout << "Details of student " << (loop+1) << ":\n"; std[loop].putDetails();
}
return 0;
}
```

Output:

Enter total number of students: 3 Enter details of student 1:
Enter name: Karthik Enter roll number: 1201
Enter total marks out of 500: 456

Enter details of student 2:
Enter name: Mahesh Enter roll number: 1202
Enter total marks out of 500: 398

Enter details of student 3:
Enter name: Kiran Enter roll number: 1203
Enter total marks out of 500: 456

Details of student 1:
Student details:
Name: Karthik, Roll Number: 101, Total: 456, Percentage: 91.2 Details of student 2:

Student details:
Name: Mahesh, Roll Number: 1202, Total: 398, Percentage:79.6

Details of student 3:
Student details:
Name: Kiran, Roll Number: 1203, Total: 398, Percentage:79.6

2. ***Write a C++ program to declare Struct. Initialize and display contents of member variables.***

Program:

```
#include <iostream>
using namespace std;
struct student
{
char name[50];
 int roll;
float marks;
```

```
};
int main()
{
student s;
cout << "Enter information," << endl;
cout << "Enter name: ";
cin >> s.name;
cout << "Enter roll number: "; cin >> s.roll;
cout << "Enter marks: "; cin >> s.marks;
cout << "\nDisplaying Information," << endl; cout << "Name: " << s.name <<
endl;
cout << "Roll: " << s.roll << endl; cout << "Marks: " << s.marks << endl;
 return 0;
}
```

Output:

```
Enter information,
Enter name: Bill
Enter roll number: 4
Enter marks: 55.6

Displaying Information
Name: Bill
Roll: 4
Marks: 55.6
```

3. Write a C++ program to declare a class. Declare pointer to class. Initialize and display the contents of the class member.

```
#include <iostream>
using namespace std;
class Box
{
public:
Box(double l = 2.0, double b = 2.0, double h = 2.0)
{
cout <<"Constructor called." << endl;
```

```
length = l;
breadth = b;
height = h;
}
double Volume()
{
return length * breadth * height;
}
private:
double length;
double breadth;
double height;
};

int main(void)
{
Box Box1(3.3, 1.2, 1.5);
Box Box2(8.5, 6.0, 2.0);
Box *ptrBox;
ptrBox = &Box1;
cout << "Volume of Box1: " << ptrBox->Volume() << endl; ptrBox = &Box2;
cout << "Volume of Box2: " << ptrBox->Volume() << endl; return 0;
}
```

Output:

```
Constructor called.
Constructor called.
Volume of Box1: 5.94
Volume of Box2: 102
```

4. ***Given that an EMPLOYEE class contains following members: data members: Employee number, Employee name, Basic, DA, IT, Net Salary and print data members.***

Program:

```
#include<iostream.h>
#include<conio.h>
class employee
{
int emp_num;
char emp_name[20]; float emp_basic; float sal;
float emp_da; float net_sal; float emp_it; public:
void get_details(); void find_net_sal();
void show_emp_details();
};
void employee :: get_details()
{
cout<<"\n Enter employee number:\n";
cin>>emp_num;
cout<<"\n Enter employee name:\n";
cin>>emp_name;
cout<<"\n Enter employee basic:\n";
cin>>emp_basic;
}
void employee :: find_net_sal()
{
emp_da=0.52*emp_basic;
emp_it=0.30*(emp_basic+emp_da);
net_sal=(emp_basic+emp_da)-emp_it;
}

void employee :: show_emp_details()
{
cout<<"\n\n\n Details of : "<<emp_name;

cout<<"\n\n Employee number: "<<emp_num; cout<<"\n Basic salary :
"<<emp_basic;
cout<<"\n Employee DA : "<<emp_da; cout<<"\n Income Tax  :"<<emp_it;
cout<<"\n Net Salary       :  "<<net_sal;
}
```

```
int main()
{
employee emp[10]; int i,num;
clrscr();
cout<<"\n Enter number of employee details\n";
 cin>>num;
for(i=0;i<num;i++)
emp[i].get_details();
for(i=0;i<num;i++)
emp[i].find_net_sal();
for(i=0;i<num;i++)
emp[i].show_emp_details();
 getch();
return 0;
}
```

Output:

```
Enter number of employee details
Enter employee number: 5123
Enter employee name: Madhav
Enter employee basic: 10000

Details of : Madhav
Employee number: 5123
Basic salary :                  10000
Employee DA                     : 5200
Income Tax          : 4560
Net Salary    : 10640
```

5. ***Write a C++ program to read the data of N employee and compute Net salary of each employee (DA=52% of Basic and Income Tax (IT) =30% of the gross salary).***

Program:

```
#include<iostream.h>
#include<conio.h>
 #define SIZE 5
class emp
{
float basic,da,it,netsal;
char name[20],num[10];
 public:
void getdata();
void net_sal();
 void dispdata();
};
void emp::getdata()
{
cout<<"\n Enter employee number: " ;
 cin>>name;
cout<<"\n Enter employee name: " ;
 cin>>num;
cout<<"Enter employee basic salary in Rs: " ;
cin>>basic;
}
void emp::net_sal()
{
da=((0.52)*basic );
float gsal=da+basic;
it=((0.3)*gsal);
netsal=gsal-it;
}

void emp::dispdata()
{
cout <<"\n Employee number: "<<name cout <<"\n Employee name: "<<num
cout <<"\n Employee netsalary: "<<netsal<<" Rs.";
}
void main()
{
clrscr();

emp ob[SIZE]; int n;
```

```
cout<<"\n\n**********************************"<<"\n Calculation of
Employee Net Salary"<<"\n**********************************"
<<"\n Enter the number of employees";
cin>>n;
for(int i=0;i<n;i++)
{
ob[i].getdata();
ob[i].net_sal();
}
clrscr();
cout<<"\n     "<<"\n Employee Detail::"<<"\n          ";
 for( i=0;i<n;i++)
{
cout<<"\n\n Employee:"<<i+1<<"\n    ";
ob[i].dispdata();
}
getch();
}
```

Output:

********** Calculation of Employee Net Salary
********** Enter the number of employees: 1

Enter employee number: 22
Enter employee name: Sanath
Enter employee basic salary in Rs: 10000

Employee Detail::

Employee:1 Employee number: 22
Employee name: Sanath
Employee netsalary: 10000 RS.

6. Write a C++ to illustrate the concepts of console I/O operations.

Program:

```
#include <iostream>
```

```
#include <fstream>
#include <cstdlib>
#include <string>
using namespace std;

int main()
{
string filename = "test.txt";
ofstream fout(filename.c_str()); // default mode is ios::out | ios::trunc

if (!fout)
{
cerr << "error: open file for output failed!" << endl;
abort();
// in <cstdlib> header
}

fout << "apple" << endl; fout << "orange" << endl; fout << "banana" << endl;
fout.close();
ifstream fin(filename.c_str()); // default mode ios::in

if (!fin)
{
cerr << "error: open file for input failed!" << endl;
abort();
}
char ch;

while (fin.get(ch))
{ // till end-of-file cout << ch;
}
fin.close();
return 0;
}
```

Output:

```
apple orange banana
```

7. Write a C++ program to use scope resolution operator. Display the various values of the same variables declared at different scope levels.

```
#include <iostream>
 using namespace std;

class programming
{
public: void output(); //function declaration
};
void programming::output()
{
cout << "Function defined outside the class.\n";
}
int main()
{
programming x; x.output(); return 0;
}
```

Ouput:

Function defined outside class

8. Write a C++ program to allocate memory using new operator.

Program:

```
#include <iostream>
using namespace std;
int main ()
{
int* p = NULL;
p = new(nothrow) int; if (!p)
cout << "allocation of memory failed\n";
else
{
*p = 29;
cout << "Value of p: " << *p << endl;
}
float *r = new float(75.25);
cout << "Value of r: " << *r << endl;
```

```
int n = 5;
int *q = new(nothrow) int[n];

if (!q)
cout << "allocation of memory failed\n"; else
{
for (int i = 0; i < n; i++) q[i] = i+1;
cout << "Value store in block of memory: ";
for (int i = 0; i < n; i++)
cout << q[i] << " ";
}
delete p; delete r;
delete[] q;
return 0;
}
```

Output:

```
Value of p: 29
Value of r: 75.25
Value store in block of memory: 1 2 3 4 5
```

9. ***Write a Program to design a class having static member function named showcount() which has the property of displaying the number of objects created of the class.***

Program:

```
#include<iostream.h>
 #include<conio.h>
class test
{
int code;
static int count;

public:
void setcode(void)
```

```
{
code = ++count;
}
void showcode(void)
{
cout<<"object number:"<<code<<"\n";
}
static void showcount(void)
{
cout<<"count:"<<count<<"\n";
}
};

int test :: count; int main()
{
test t1,t2;

t1.setcode();
t2.setcode();

test :: showcount();

test t3; t3.setcode();

test :: showcount(); t1.showcode(); t2.showcode(); t3.showcode(); return 0;
}
```

10. Write a Program using class to process Shopping List for a Departmental Store. The list include details such as the Code No and Price of each item and perform the operations like Adding, Deleting Items to the list and Printing the Total value of a Order.

Program:

```
#include<iostream.h>
 const m=50;
class ITEMS
{
int itemCode[m];
 float itemPrice[m];
```

```
int count;
public:
void CNT(void){count=0;}
 void getitem(void);
void displaySum(void);
void remove(void);
void displayItems(void);
};
void ITEMS :: getitem(void)
{
cout<<"Enter item code"; cin>>itemCode[count]; cout<<"Enter Item cost";
cin>>itemPrice[count]; count++;
}
void ITEMS :: displaySum(void)
{
float sum=0;
for(int i=0;i<count;i++) sum=sum+itemPrice[i];
cout<<"\n Total Value:"<<sum<<"\n";
}
void ITEMS :: remove(void)
{
int a;
cout<<"Enter Item Code"; cin>>a;
for(int i=0;i<count;i++) if(itemCode[i] == a)
itemPrice[i]=0;
}
void ITEMS :: displayItems(void)
{
cout<<"\n Code     Price\n";

for(int i=0;i<count;i++)
{
cout<<"\n"<<itemCode[i]; cout<<"     "<<itemPrice[i];
}
cout<<"\n";
}



int main()
```

```
{
ITEMS order; order.CNT(); int x;
do
{
cout<<"\n You can do the following;"<<"Enter appropriate number\n";
cout<<"\n1 : Add an Item";
cout<<"\n2 : Display Total Value";
cout<<"\n3 : Delete an Item"; cout<<"\n4 : Display all items";
 cout<<"\n5 : Quit";
cout<<"\n\n What is your option?";
cin>>x;
switch(x)
{
case 1 : order.getitem(); break;
case 2 : order.displaySum(); break;
case 3 : order.remove(); break;
case 4 : order.displayItems(); break;
default : cout<<"Error in input";
}
}while(x!=5); return 0;
}
```

11. Write a Program which creates & uses array of object of a class.(for eg. implementing the list of Managers of a Company having details such as Name, Age, etc..).

Program:

```
#include<iostream.h>
#include<conio.h>
class employee
{
char name [30]; float age;
public:
void getdata(void); void putdata(void);
};
void employee :: getdata(void)
{
cout<<"Enter Name"; cin>>name; cout<<"Enter Age"; cin>>age;
```

```
}
void employee :: putdata(void)
{
cout<<"Name:"<<name<<"\n"; cout<<"Age:          "<<age<<"\n";
}
const int size=3; int main()
{

employee manager[size]; for(int i=0; i<size; i++)
{
cout<<"\n Details of manager"<<i+1<<"\n"; manager[i].getdata();
}
cout<<"\n";
for(i=0; i<size; i++)
{
cout<<"\n Manager"<<i+1<<"\n"; manager[i].putdata();
}

return 0;
}
```

12. Write a Program to find Maximum out of Two Numbers using friend function.

Note: Here one number is a member of one class and the other number is member of some other class.

Program:

```
#include<iostream.h>
 #include<conio.h>
 class ABC;
class XYZ
{
int x; public:
void setvalue(int i)
{
x=i;
}
friend void max(XYZ, ABC);
};
```

```
class ABC
{
int a; public:
void setvalue(int i)
{
a=i;
}
friend void max(XYZ, ABC);
};
void max (XYZ m, ABC n)
{
if(m.x>=n.a)
cout<<m.x;

else

}


cout<<n.a;

int main()
{
ABC abc; abc.setvalue(10); XYZ xyz; xyz.setvalue(20); max(xyz,abc);

return 0;
}
```

13. Write a Program to swap private data members of classes named as class_1, class_2 using friend function.

Program:

```
#include<iostream.h>
 #include<conio.h>
class class_2;
class class_1
{
int value1; public:
void indata(int a)
{
```

```
value1=a;
}
void display(void)
{
cout<<value1<<"\n";
}
friend void exchange(class_1 &, class_2 &);
};
class class_2
{
int value2; public:
void indata(int a)
{
value2=a;
}
void display(void)
{
cout<<value2<<"\n";
}
friend void exchange(class_1 &, class_2 &);
};
void exchange(class_1 &x, class_2 &y)
{
int temp = x.value1; x.value1 = y.value2; y.value2 = temp;
}
int main()
{
class_1 C1;
class_2 C2;

C1.indata(100);
C2.indata(200);
cout<<"Values before exchange"<<"\n";

C1.display();
C2.display();
exchange(C1, C2);
cout<<"Values after exchange"<<"\n";
 C1.display();
C2.display();
```

```
 return 0;
}
```

14. Write a Program to design a class complex to represent complex numbers. The complex class shuold use an external function (use it as a friend function) to add two complex numbers.The function should return an object of type complex representing the sum of two complex numbers.

Program:

```
#include<iostream.h>
#include<conio.h>
class complex
{
float x;
 float y;
public:
void input(float real, float img)
{
x=real; y=img;
}
friend complex sum(complex, complex); void show(complex);
};
complex sum(complex c1, complex c2)
{
complex c3;
c3.x = c1.x + c2.x; c3.y = c1.y + c2.y; return (c3);
}
void complex :: show(complex c)
{
cout<<c.x<<"+j"<<c.y<<"\n";
}
int main()
{
complex A,B,C; A.input(3.1, 5.65);
B.input(2.75, 1.2);

C=sum(A,B);
cout<<"A="; A.show(A); cout<<"B="; B.show(B); cout<<"C="; C.show(C);

return 0;
```

```
}
```

15. Write a Program using copy constructor to copy data of an object to another object.

```
#include<iostream.h>
 #include<conio.h>
class code
{
int id;
public:
code(){}
code(int a)
{
id = a;
}
code(code & x)
{
id = x.id;
}
void display(void)
{
cout<<id;
}
};
int main()
{

code A(100);
code B(A);
code C = A;
code D;
D = A;
cout<<"\n id of A:";
 A.display();
cout<<"\n id of B:";
B.display();
cout<<"\n id of C:";
C.display();
 cout<<"\n id of D:";
```

```
D.display();

return 0;
}
```

16. Write a Program to allocate memory dynamically for an objects of a given class using class's constructor.

Program:

```
#include<iostream.h>
#include<string.h>
#include<conio.h>
 class String
{
char *name; int length;
public:
String()
{
length = 0;
name = new char[length +1];
}
String (char *s)
{
length = strlen(s);
name= new char[length + 1]; strcpy(name, s);
}
void display(void)
{
cout<<name<<"\n";
}
void join(String &a, String &b);
};

void String :: join (String &a, String &b)
{
length = a.length + b.length; delete name;
name = new char [length + 1];

strcpy(name,a.name); strcat(name, b.name);
};
```

```
int main()
{

char *first = "Joseph";
String name1(first), name2("Louis "), name3("Lagrange"),s1,s2; s1.join(name1,
name2);
s2.join(s1, name3); name1.display(); name2.display(); name3.display();
s1.display();
s2.display();

return 0;
}
```

17. Write a Program to design a class to represent a matrix. The class should have the functionality to insert and retrieve the elements of the matrix.

```
#include<iostream.h>
class matrix
{
int **p; int d1,d2;
public:
matrix(int x, int y);
void get_element(int i, int j, int value)
{
p[i][j]=value;
}
int & put_element(int i, int j)
{
return p[i][j];
}
};
matrix ::matrix(int x, int y)
{
d1 = x; d2 = y;
p = new int *[d1];
for(int i = 0; i < d1; i++) p[i] = new int[d2];
}

int main()
```

```
{
int m, n;

cout<<"Enter size of matrix"; cin>>m>>n;
matrix A(m,n);
cout<<"Enter Matrix Element row by row:"; int i,j,value;

for(i=0;i<m;i++)
for(j=0;j<n;j++)
{




}
cout<<"\n";

cin>>value; A.get_element(i,j,value);

cout<<A.put_element(1,2); return 0;
}
```

18. Write a program to design a class representing complex numbers and having the functionality of performing addition & multiplication of two complex numbers using operator overloading.

```
#include<iostream.h>
class complex
{
private:
float real,
imag;
public:
complex( )
{
}
complex( float r, float i )
{
real = r; imag = i;
}
```

```
void getdata( )
{
float r,
i;
cout << endl << "Enter real and imaginary part "; cin >> r >> i;
real = r; imag = i;
}

void setdata( )
{
real = r; imag = i;
}

void displaydata( )
{
cout << endl << "real = " << real; cout<<endl<<Imaginary = "<<imag;
}

complex operator +( complex c )
{
complex t;
t.real = real + c.real; t.imag = imag + c.imag;
}

complex operator *( complex c )
{
complex t;
t.real = real * c.real - imag * c.imag; t.imag = real * c.imag + c.real * imag;
return t;
}
}        ;
void main( )
{

complex c1,c2 ( 1.2, -2.5 ),c3,c4;
c1.setdata( 2.0, 2.0 );
 c3 = c1 + c2;
c3.displaydata( );
c4.getdata( );
complex c5 ( 2.5, 3.0 ),c6;
```

```
 c6 = c4 * c5;
c6.displaydata( );
complex c7;
c7 = c1 + c2 * c3;
 c7.displaydata( );
}
```

19. Write a Program to overload operators like *, <<, >> using friend function. The following overloaded operators should work for a class vector.

```
#include<iostream.h>
 #include<conio.h>
const size = 3;
class vector
{
int v[size];

public:
vector(); vector(int *x);
friend vector operator *(int a, vector b); friend vector operator *(vector b, int a);
friend istream & operator >>(istream &, vector &); friend ostream & operator
<<(ostream &, vector &);
};

vector ::vector()
{
for(int i=0;i<size;i++) v[i]=0;
}

vector :: vector(int *x)
{
for(int i=0; i<size; i++) v[i] = x[i];
}

vector operator *(int a, vector b)
{
vector c;
for(int i=0; i<size; i++) c.v[i] = a * b.v[i];
return c;
```

```
}

vector operator *(vector b, int a)
{
vector c;
for(int i=0; i<size; i++) c.v[i] = b.v[i] * a;
return c;
}

istream & operator >> (istream &din, vector &b)
{
for(int i=0; i<size; i++) din>>b.v[i];
return(din);
}


ostream & operator << (ostream &dout, vector &b)
{

dout<<"("<<b.v [0];

for(int i=1; i<size; i++) dout<<","<<b.v[i];
dout<<")"; return(dout);
}

int x[size] = {2,4,6};

int main()
{

vector m; vector n = x;

cout<<"Enter Elements of vector m"<<"\n"; cin>>m;

cout<<"\n"; cout<<"m="<<m<<"\n";

vector p,q;

p = 2 * m;
q = n * 2;
```

```
cout<<"\n"; cout<<"p="<<p<<"\n"; cout<<"q="<<q<<"\n";

return 0;
}
```

20. Write a program for developing a matrix class which can handle integer matrices of different dimensions. Also overload the operator for addition, multiplication & comparison of matrices.

```
#include<iostream.h>
 #include<iomanip.h>
 class matrix
{
int maxrow, maxcol; int * ptr;
public:
matrix( int r, int c )
{
maxrow = r; maxcol = c;
ptr      = new int [r * c];
}
void getmat( )
{
int i,j, mat_off,temp;
cout << endl << "enter elements matrix:" << endl; for( i = 0; i < maxrow; i++ )
{
for( j = 0; j < maxcol; j++ )
{
mat_off = i * maxcol + j; cin >> ptr[ mat_off ];
}
}
}
void printmat( )
{
int i, j, mat_off;
for( i = 0; i < maxrow; i++ )
{
cout << endl;
for( j = 0; j < maxcol; j++ )
{
```

```
mat_off = i * maxcol + j;
cout << setw( 3 ) << ptr[ mat_off ];
}
}
}
int delmat( )
{
matrix q ( maxrow - 1, maxcol - 1 );
int      sign = 1, sum = 0, i, j,k,count; int        newsize,newpos,pos,order;
order = maxrow; if( order == 1 )
{
return ( ptr[ 0 ] );
}
for( i = 0; i < order; i++, sign *= -1 )
{
for( j = 1; j < order; j++ )
{
for( k = 0, count = 0; k < order; k++ )

{
if( k == i )
continue;
pos = j * order + k;
newpos = ( j - 1 ) * ( order - 1 ) + count; q.ptr[ newpos ] = ptr[ pos ];
count++;
}
}
sum = sum + ptr[ i ] * sign * q.delmat( );
}
return ( sum );
}
matrix operator +( matrix b )
{
matrix c ( maxrow, maxcol ); int         i,j,mat_off;
for( i = 0; i < maxrow; i++ )
{
for( j = 0; j < maxcol; j++ )
{
mat_off = i * maxcol + j;
c.ptr[ mat_off ] = ptr[ mat_off ] + b.ptr[ mat_off ];
```

```
}
}
return ( c );
}
matrix operator *( matrix b )
{
matrix c ( b.maxcol, maxrow );
int      i,j,k,mat_off1, mat_off2, mat_off3; for( i = 0; i < c.maxrow; i++ )
{
for( j = 0; j < c.maxcol; j++ )
{
mat_off3 - i * c.maxcol + j; c.ptr[ mat_off3 ] = 0;
for( k = 0; k < b.maxrow; k++ )
{
mat_off2 = k * b.maxcol + j; mat_off1 = i * maxcol + k;
c.ptr[mat_off3]+=ptr[mat_off1]* b.ptr[mat_off2 ];
}
}
}
return ( c );
}
int operator ==( matrix b )
{
int i,j, mat_off;
if( maxrow != b.maxrow
|| maxcol != b.maxcol ) return ( 0 );
for( i = 0; i < maxrow; i++ )
{
for( j = 0; j < maxcol; j++ )
{
mat_off = i * maxcol + j; if( ptr[ mat_off ]

!= b.ptr[ mat_off ] ) return ( 0 );
}
}
return ( 1 );
}
}        ;

void main( )
```

```
{
int rowa, cola, rowb, colb;
cout << endl << "Enter dimensions of matrix A "; cin >> rowa >> cola;
matrix a ( rowa, cola ); a.getmat( );
cout << endl << "Enter dimensions of matrix B"; cin >> rowb >> colb;
matrix b ( rowb, colb ); b.getmat( );
matrix c ( rowa, cola ); c = a + b;
cout << endl << "The sum of two matrics = "; c.printmat( );
matrix d ( rowa, colb ); d = a * b;
cout << endl << "The product of two matrics = "; d.printmat( );
cout << endl << "Determinant of matrix a =" << a.delmat( ); if( a == b )
cout << endl << "a & b are equal"; else
cout << endl << "a & b are not equal";
}
```

21. Write a program to overload new/delete operators in a class.

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

const int MAX = 5;
 const int FREE = 0;
const int OCCUPIED = 1;

void memwarning( )
{
cout << endl << "Free store has now gone empty"; exit( 1 );
}

class employee
{
private:
char name[ 20 ]; int          age;
float sal;

public:
void *operator new(size_t bytes) void operator delete( void * q );
void setdata( char * n, int a, float s ); void showdata( );
```

```
~employee( );
}       ;

struct pool
{
employee obj; int   status;
}       ;

int      flag = 0; struct pool * p = NULL;

void * employee::operator new( size_t sz )
{
int i;
if( flag == 0 )
{
p = ( pool * )malloc( sz * MAX ); if( p == NULL )
memwarning( );
for( i = 0; i < MAX; i++ ) p[ i ].status = FREE;
flag    = 1;
p[ 0 ].status = OCCUPIED; return &p[ 0 ].obj;
}

else
{
for( i = 0; i < MAX; i++ )

{
if( p[ i ].status = FREE )
{
p[ i ].status = OCCUPIED; return &p[ i ].obj;
}
}
memwarning( );
}
}

void employee::operator delete( void * q )
{
if( q == NULL )
return;
```

```
for( int i = 0; i < MAX; i++)
{
if( q == &p[ i ].obj )
{
p[ i ].status = FREE;
strcpy( p[ i ].obj.name, "" ); p[ i ].obj.age = 0;
p[ i ].obj.sal = 0.0;
}
}
}

void employee::setdata( char * n, int a, float s )
{
strcpy( name, n ); age = a;
sal = s;
}

void employee::showdata( )
{
cout << endl << name << "\t" << age << "\t" << sal;
}

employee::~employee( )
{
cout << endl << "reached destructor"; free( p );
}

void main( )
{
void memwarning( ); set_new_handler( memwarning ); employee *
e1,*e2,*e3,*e4,*e5,*e6; e1 = new employee;
e1->setdata( "ajay",          23,     4500.50        );
e2 = new employee; e2->setdata( "amol",      25,     5500.50        );
e3 = new employee; e3->setdata( "anil",
26,
3500.50
);

e4 = new employee;
e4->setdata( "anuj", 30, 6500.50 );
```

```
e5 = new employee;
e5->setdata( "atul", 23, 4200.50 );

e1->showdata( ); e2->showdata( ); e3->showdata( ); e4->showdata( ); e5-
>showdata( );

delete e4; delete e5;

e4->showdata( ); e5->showdata( );

e4 = new employee; e5 = new employee; e6 = new employee;

cout << endl << "Done!!";
}
```

22.Write a program in C++ to highlight the difference between overloaded assignment operator and copy constructor.

```
#include<iostream.h>
class circle
{
private:
int      radius; float x, y;

public:
circle( )
{
}
circle( int rr, float xx, float yy )
{
radius = rr; x          = xx;
y        = yy;
}
circle operator =( circle & c )
{
cout << endl << "Assignment operator invoked"; radiius = c.radius;
x        = c.x;
y        = c.y;
return circle( radius, x, y );
}
```

```
circle( circle & c )
{
cout << endl << "copy constructor invoked"; radius = c.radius;
x       = c.x;
y       = c.y;
}
void showdata( )
{
cout << endl << "Radius = " << radius; cout << endl << "X-Coordinate=" << x;
cout << endl << "Y-Coordinate=" << y;
}
}       ;

void main( )
{
circle c1 ( 10, 2.5, 2.5 ); circle c2,c4;
c4 = c2 = c1; circle c3 = c1; c1.showdata( ); c2.showdata( ); c3.showdata( );
c4.showdata( );
}
```

23. Write a program to implement the exception handling with the functionality of testing the throw restrictions.

```
#include<iostream.h>
void test(int x) throw(int, double)
{
if(x==0)
throw 'x';

else


if(x == 1)
throw x;




}

int main()
```

```
{

else


if(x == -1)
throw 1.0;
cout<<"End of Function Block\n";

try
{









}



cout<<"Testting Throw Restrictions\n"; cout<<"x == 0\n";
test(0); cout<<"x == 1\n"; test(1);
cout<<"x == -1\n"; test(-1); cout<<"x == 2\n"; test(2);

catch(char c)
{
cout<<"Caught a Character\n";
}

catch(int m)
{
cout<<"Caught an Integer\n";
}

catch(double d)
{
cout<<"Caught a Double\n";
```

```
}

cout<<"End of Try-catch system\n"; return 0;
}
```

24. Write a function template that will sort an array of implicit types like int,float,char etc. it can also sort user-defined objects like strings & date. The necessary classes contains overloading of operators.

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

class mystring
{
private:
enum
{
sz = 100       //       < >
}       ;
char str[ sz ];

public:
mystring( char * s = "" )
{
strcpy( str, s );
}

int operator <( mystring ss )
{
if( strcmp( str, ss.str ) <= 0 ) return 1;
else
return 0;
}

int operator <=( mystring ss )
{
if( strcmp( str, ss.str ) <= 0 ) return 1;
else
return 0;
}
```

```
int operator >( mystring ss )
{
if( strcmp( str, ss.str ) > 0 ) return 1;
else
return 0;
}

friend ostream & operator <<( ostream & o,mystring & dd );

}       ;

ostream operator <<( ostream & o, mystring & ss )
{
o << ss.str; return o;
}


class date
{

private:
int day,
mth, yr;
public:
date( int d = 0, int m = 0, int y = 0 )
{
day = d; mth = m; yr = y;
}

int operator <( date dt )
{
if( yr < dt.yr ) return 1;
if( yr == dt.yr && mth < dt.mth ) return 1;
if( yr == dt.yr && mth == dt.mth && day = dt.day ) return 1;
return 0;
}

class date
{
```

```
private:
int day, mth, yr; public:
date( int d = 0, int m = 0, int y = 0 )
{
day = d; mth = m; yr = y;
}

int operator <( date dt )
{
if( yr < dt.yr ) return 1;
if( yr == dt.yr && mth < dt.mth ) return 1;
if( yr == dt.yr && mth == dt.mth && day < dt.day )
return 1;
return 0;
}
int operator <=( date dt )
{
if( yr <= dt.yr ) return 1;
if( yr == dt.yr && mth <= dt.mth ) return 1;
if( yr == dt.yr && mth == dt.mth && day <= dt.yr )
return 1;
return 0;
}
int operator >( date dt )
{

if( yr > dt.yr ) return 1;
if( yr == dt.yr && mth > dt.mth ) return 1;
if( yr == dt.yr && mth == dt.mth && day > dt.day )
return 1;
return 0;
}

friend ostream & operator <<( ostream & o, date & dd );
}       ;

ostream & operator <<( ostream & o, date & dd )
{
o << dd.day << "\t" << dd.mth << "\t" << dd.yr; return 0;
}
```

```
template<class T> void quick( T * n, int low, int high )
{
int pos;
if( low < high )
{
pos = split( n, low, high ); quick( n, low, pos - 1 ); quick( n, pos + 1, high );
}
}

template<class T> int split( T * n, int low, int high )
{
int pos,
left, right;
T       item, t;

item = n[ low ]; left = low; right = high;

while( left < right )
{
while( n[ right ] > item ) right = right - 1;

while( ( left < right )
&& ( n[ left ] <= item ) ) left = left + 1;

if( left < right )
{
t       = n[ left ]; n[ left ] = n[ right ]; n[ right ] = t;
}
}
pos     = right;
t       = n[ low ];

n[ low ] = n[ pos ]; n[ pos ] = t; return pos;
}



void main( )
{
float num[]={5.4f,3.23f,2.15f,1.09f,34.66f,23.3452f};
```

```
int arr[]={-12,23,14,0,245,78,66,-9};

date dtarr[]={date(17,11,62),date(23,12,65),date(12,12,78)
,date(23,1,69)};

mystring strarr[]={mystring("Kamal"),mystring("Anuj"),
mystring("Sachin"),mystring("Anil")};

int      i;
cout << endl << endl;

quick( num, 0, 5 );
for( i = 0; i <= 5; i++ ) cout << num[ i ] << endl;

cout << endl << endl; quick( arr, 0, 7 );
for( i = 0; i <= 7; i++ ) cout << arr[ i ] << endl;

cout << endl << endl; quick( dtarr, 0, 3 ); for( i = 0; i <= 3; i++ )
cout << dtarr[ i ] << endl;

cout << endl << endl; quick( strarr, 0, 3 ); for( i = 0; i <= 3; i++ )
cout << strarr[ i ] << endl;

}
```

25. Write a program implementing stack and it's operations using template class.

```
#include<iostream.h>
const int MAX = 10;
template<class T>
class stack
{
private:
T       stk[ MAX ]; int top;

public:
stack( )
{
top = -1;
```

```
}
void push( T data )
{
if( top == MAX - 1 )
cout << endl << "Stack is full"; else
{
top++;
stk[ top ] = data;
}
}

T pop( )
{
if( top == -1 )
{
cout << endl << "Stack is empty"; return NULL;
}
else
{
T data = stk[ top ]; top--;
return data;
}
}
}       ;

class complex
{
private:
float real,
imag;

public:
complex( float r = 0.0, float i = 0.0 )
{
real = r; imag = i;
}
friend ostream & operator <<( ostream & o,
complex & c );

}       ;
```

```
ostream & operator <<( ostream & o, complex & c )
{
o << c.real << "\t" << c.imag; return o;
}

void main( )
{
stack< int > s1; s1.push( 10 );
s1.push( 20 );
s1.push( 30 );

cout << endl << s1.pop( ); cout << endl << s1.pop( ); cout << endl << s1.pop( );
stack< float >        s2; s2.push( 3.14 );
s2.push( 6.28 );
s2.push( 8.98 );

cout << endl << s2.pop( ); cout << endl << s2.pop( ); cout << endl << s2.pop( );

complex       c1 ( 1.5, 2.5 ),
c2 ( 3.5, 4.5 ),
c3 ( -1.5, -0.6 );
stack< complex >  s3; s3.push( c1 );
s3.push( c2 );
s3.push( c3 );

cout << endl << s3.pop( ); cout << endl << s3.pop( ); cout << endl << s3.pop( );

}
```

26. Write a program implementing linked list & some required operations on it using class template.

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

class emp
{
private:
char name[ 20 ];
```

```
 int     age;
float sal;

public:
emp( char * n = "", int a = 0, float s = 0.0 )
{
strcpy( name, n );
age = a;
sal = s;
}

friend ostream & operator <<( ostream & s, emp & e );

}       ;
ostream operator <<( ostream & s, emp & e )
{
cout << e.name << "\t" << e.age << "\t" << e.sal;
return s;
}

template<class T>class linklist
{
private:
struct node
{
T       data; node * link;
}       * p;

public:
linklist( );
~linklist( ); void append( T );
void addatbeg( T );
void addafter( int, T ); void del( int );
void display( ); int count( );
}       ;

template<class T> linklist< T >::linklist( )
{
p = NULL;
}
```

```
template<class T> linklist< T >::~linklist( )
{
node * t;

while( p != NULL )
{
t = p;
p = p->link; delete t;
}
}

template<class T> void linklist< T >::append( T num )
{
node * q,
*       t;
if( p == NULL )
{
p       = new node; p->data = num;
p->link = NULL;
}
else
{
q = p;
while( q->link != NULL ) q = q->link;
t       = new node; t->data = num;
t->link = NULL; q->link = t;
}
}

template<class T> void linklist< T >::addatbeg( T num )
{
node * q;
q       = new node; q->data = num;
q->link = p; p        = q;
}

template<class T> void linklist< T >::addafter( int c,
T num )
```

```
{
node * q,
*       t;
int     i;

for( i = q, q = p; i <= c; i++ )
{
q = q->link; if( q == NULL )
{
cout << endl << "There are less than" << c << "element"; return;
}
}

t       = new node; t->data = num;
t->link = q->link;

q->link = t;
}

template<class T> void linklist< T >::del( int n )
{
node * q,
*       r;
int     i = 1; q = p;
if( n == 1 )
{
p = q->link; delete q; return;
}
r = q;
while( q != NULL )
{
if( i == n )
{
r->link = q->link; delete q;
return;
}
r = q;
q = q->link; i++;
}
cout << endl << "Element" << n << "not found";
```

```
}

template<class T> void linklist< T >::display( )
{
node * q; cout << endl;
for( q = p; q != NULL; q = q->link ) cout << q->data << endl;
}

template<class T> int linklist< T >::count( )
{
node  * q; int         c = 0;
for( q = p; q != NULL; q = q->link ) c++;
return ( c );
}

void main( )
{
linklist< int >        l1;
cout << endl << "No. of elements in linked list = " << l1.count( );

l1.append( 11 );
l1.append( 22 );
l1.append( 33 );
l1.append( 44 );
l1.append( 55 );
l1.append( 66 );

l1.addatbeg( 100 );
l1.addatbeg( 200 );

l1.addafter( 3, 333 );
l1.addafter( 4, 444 ); l1.display( );
cout << endl << "No. of elements in linked list=" << l1.count( );

l1.del( 200 );
l1.del( 66 );
l1.del( 0 );
l1.del( 333 );

l1.display( );
```

```
cout << endl << "no. of elements in linked list = " << l1.count( ); linklist< emp
>        l2;
cout << endl << "No. of elements in linked list = " << l2.count( ); emp e1 (
"Sanjay", 23, 1100.00 );
emp e2 ( "Rahul", 33, 3500.00 ); emp e3 ( "Rakesh", 24, 2400.00 ); emp e4 (
"Sanket", 25, 2500.00 ); emp e5 ( "Sandeep", 26, 2600.00 );

l2.append( e1 ); l2.append( e2 ); l2.append( e3 ); l2.append( e4 ); l2.append( e5
);

l2.display( );

l2.del( 3 );
l2.display( );

cout << endl << "No. of elements in linked list = " << l2.count( );
l2.addatbeg( e5 );
l2.display( );

l2.addafter( 3, e1 ); l2.display( );

cout << endl << "No. of elements in linked list = " << l2.count( );
}
```

GROUP-A: OPERATING SYSTEM LABORATORY MANUAL

(Course Code: BCA-2297)

List of Experiments:

1. Write a shell program to find the highest of three numbers.
2. Write a shell program to generate first 10 Fibonacci numbers.

3. Write a shell program to print all the prime numbers between the range.
4. Write a shell program to check a year is leap year or not.
5. Write a shell program to find factorial of a number.
6. Write a shell program to calculate GCD and LCM of two numbers.
7. Write a shell program to check weather a number is Armstrong or not.
8. Write a shell program to check weather a number is palindrome or not.
9. Write a shell program to check weather a string is palindrome or not.
10. Write a shell program to sort the number using Bubble sort technique.
11. Write a shell program to find the root of quadratic equation.
12. Write a program to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information).
13. Write a shell program to check a file is existed or not.
14. Write a program to copy files using system calls.
15. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.
16. Write a program in C to create a Zombie Process.
17. How to execute zombie and orphan process in a single C program?
18. Write a program in C to print process id of a process and its parent process id also.
19. Write a program in C to duplicate a program's process using fork ().
20. Write program to calculate sum of n numbers using thread library.
21. Write a program in C to implement FCFS CPU scheduling algorithm.
22. Write program to implement Round Robin scheduling algorithm.
23. Write program to implement SJF scheduling algorithm.
24. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

1. Write a shell program to find the highest of three numbers.

Program:

```
echo "Enter three numbers"
read a
read b
read c
if test $a -gt $b -a $a -gt $c
then
        echo "$a is greater than $b and $c"
elif test $b -gt $c
then
        echo "$b is greater than $a and $c"
else
        echo "$c is greater than $a and $b"
fi
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ bash highest3.sh
Enter three numbers
56
78
90
90 is greater than 56 and 78
sanjoy@SANJUVAI:~$ bash highest3.sh
Enter three numbers
88
66
55
88 is greater than 66 and 55
sanjoy@SANJUVAI:~$ bash highest3.sh
Enter three numbers
99
100
98
100 is greater than 99 and 98
```

2. ***Write a shell program to generate first 10 Fibonacci numbers.***

Program:

```
clear
```

```
echo "How many numbers to be generated?"
read n
x=0
y=1
echo "Fibonacci series upto $n terms"
echo "$x"
echo "$y"
for((i=2;i<n;i++))
do
        z=$((x+y))
        echo "$z"
        x=$y
        y=$z
done
```

Input and Output Section:

```
How many numbers to be generated?
10
Fibonacci series upto 10 terms
0
1
1
2
3
5
8
13
21
34
```

3. Write a shell program to print all the prime numbers between the range.

Program:

```
echo "Enter the range: "
echo "Lower limit range:"
read l
echo "Upper limit range: "
read u
for((i=l;i<=u;i++))
do
	f=0
	n=$i
	for((j=2;j<=n-1;j++))
	do
		if test $((n%j)) -eq 0
		then
			f=1
		fi
	done
	if test $f -eq 0
	then
		echo "All prime number are: $i"
	fi
done
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ bash prime.sh
Enter the range:
Lower limit range:
2
Upper limit range:
100
All prime number are: 2
All prime number are: 3
All prime number are: 5
All prime number are: 7
All prime number are: 11
All prime number are: 13
All prime number are: 17
```

All prime number are: 19
All prime number are: 23
All prime number are: 29
All prime number are: 31
All prime number are: 37
All prime number are: 41
All prime number are: 43
All prime number are: 47
All prime number are: 53
All prime number are: 59
All prime number are: 61
All prime number are: 67
All prime number are: 71
All prime number are: 73
All prime number are: 79
All prime number are: 83
All prime number are: 89
All prime number are: 97

4. Write a shell program to check a year is leap year or not.

Program:

```
echo "Enter the year: "
read year
case1=$((year%4))
case2=$((year%100))
case3=$((year%400))
if test $case1 -eq 0 -a $case2 -ne 0 -o $case3 -eq 0
then
        echo "The year $year is Leap Year"
else
        echo "The year $year is not a Leap Year"
fi
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ bash leap.sh
Enter the year:
2004
The year 2004 is Leap Year
sanjoy@SANJUVAI:~$ bash leap.sh
Enter the year:
1900
The year 1900 is not a Leap Year
sanjoy@SANJUVAI:~$ bash leap.sh
Enter the year:
2032
The year 2032 is Leap Year
```

5. Write a shell program to find factorial of a number.

Program:

```
echo "Enter the number"
read n
fact=1
for((i=1;i<=n;i++))
do
        fact=$((fact*i))
done
echo "The factorial of $n is = $fact"
```

Input and Output Section:

```
Enter the number
5
The factorial of 5 is = 120
sanjoy@SANJUVAI:~$ bash Factorial.sh
Enter the number
6
The factorial of 6 is = 720
sanjoy@SANJUVAI:~$ bash Factorial.sh
Enter the number
10
The factorial of 10 is = 3628800
```

6. Write a shell program to calculate GCD and LCM of two numbers.

Program:

```
echo "Enter two numbers"
read m
read n
a=$m
b=$n
while (($m != $n))
do
        if test $m -gt $n
        then
                m=$((m-n))
        else
                n=$((n-m))
        fi
done
echo "$a and $b GCD is $m"
l=$((((a*b))/m))
echo "$a and $b LCM is $l"
```

Input and Output Section:

```
Enter two numbers
43
67
43 and 67 GCD is 1
43 and 67 LCM is 2881
sanjoy@SANJUVAI:~$ bash gcd.sh
Enter two numbers
11
121
11 and 121 GCD is 11
11 and 121 LCM is 121
sanjoy@SANJUVAI:~$ bash gcd.sh
Enter two numbers
13
91
13 and 91 GCD is 13
```

13 and 91 LCM is 91

7. Write a shell program to check weather a number is Armstrong or not.

Program:

```
echo "Enter the number"
read n
p=$n
d=0
while((n!=0))
do
        n=$((n/10))
        d=$((d+1))
done
echo "the digit is = $d"
sum=0
n=$p
while((n!=0))
do
        r=$((n%10))
        sum=$((sum+r**d))
        n=$((n/10))
done
if test $p -eq $sum
then
        echo "The number $p is Armstrong"
else
        echo "The number $p is not Armstrong"
fi
```

Input and Output Section:

```
Enter the number
371
the digit is = 3
The number 371 is Armstrong
sanjoy@SANJUVAI:~$ bash Armstrong.sh
Enter the number
```

```
407
the digit is = 3
The number 407 is Armstrong
sanjoy@SANJUVAI:~$ bash Armstrong.sh
Enter the number
423
the digit is = 3
The number 423 is not Armstrong
sanjoy@SANJUVAI:~$ bash Armstrong.sh
Enter the number
153
the digit is = 3
The number 153 is Armstrong
```

8. Write a shell program to check weather a number is palindrome or not.

Program:

```
clear
echo "Enter the number"
read n
p=$n
sum=0
while((n!=0))
do
r=$((n%10))
sum=$((sum*10+r))
n=$((n/10))
done
if test $p -eq $sum
then
echo "The number $p is Palindrome"
else
echo "The number $p is not Palindrome"
fi
```

Input and Output Section:

Enter the number
125521
The number 125521 is Palindrome

Enter the number
123312
The number 123312 is not Palindrome

9. Write a shell program to check weather a string is palindrome or not.

Program:

```
echo "Enter the String: "
read str
len=${#str}
f=0
for((i=0,j=len-1;i<len;i++,j--))
do
        if test ${str:i:1} != ${str:j:1}
        then
                f=1
                break
        fi
done
if test $f -eq 0
then
        echo "The String $str is palindrome"
else
        echo "The String $str is not palindrome"
fi
```

Input and Output Section:

Enter the String:
MadaM
The String MadaM is palindrome
Enter the String:
madaM
The String madaM is not palindrome

10. Write a shell program to sort the number using Bubble sort technique.

Program:

```
clear
echo "Enter the how many number "
read n
echo "Enter the elements"
for((i=0;i<n;i++))
do
        read a[$i]
done
echo "Before sorting elements are "
for((i=0;i<n;i++))
do
        echo "${a[$i]}"
done
for((i=0;i<n-1;i++))
do
        for((j=0;j<n-i-1;j++))
        do
                if test ${a[$j]} -gt ${a[$j+1]}
                then
                        t=${a[$j]}
                        a[$j]=${a[$j+1]}
                        a[$j+1]=$t
                fi
        done
done
echo "After the sorting elements are "
for((i=0;i<n;i++))
do
        echo "${a[$i]}"
done
```

Input and Output Section:

```
Enter the how many number
10
Enter the elements
10
23
67
1
34
17
569
56
78
672
Before sorting elements are
10
23
67
1
34
17
569
56
78
672
After the sorting elements are
1
10
17
23
34
56
67
78
569
672
```

11. Write a shell program to find the root of quadratic equation.

Program:

```
clear
echo "Enter the three number "
read a
read b
read c
d=$((((b**2))-((4*a*c))))
if test $d -lt 0
then
        echo "The roots are imaginary "
else
        d1=`echo "sqrt($d) " | bc`
        echo "$d1"
        root1=`echo "scale=2;(-$b+$d1)/(2*$a)" | bc`
        root2=`echo "scale=2;(-$b-$d1)/(2*$a)" | bc`
        echo "Real roots are "$root1, $root2
fi
```

Input and Output Section:

```
Enter the three number
5
6
1
4
Real roots are -.20, -1.00
Enter the three number
1
5
6
1
Real roots are -2.00, -3.00
Enter the three number
2
5
4
The roots are imaginary
```

12. Write a program to report behaviour of Linux kernel including kernel version, CPU type and model. (CPU information).

Program:

```
#include<stdio.h>
#include<stdlib.h>
#include<sys/utsname.h>
using namespace std;
int main()
{
   int m=0;
   struct utsname s1;
   m=uname(&s1);

   if(m==0)
   {
      printf("\n The name of System:%s , system" , s1.sysname);
      printf("\n The version:%s" , s1.version);
      printf("\n The Machine:%s" , s1.machine);
      printf("\n");
      system("cat /proc/cpuinfo | awk 'NR==3, NR==4{print}' \n");
   }

   else
   {
      printf("Error");
   }
   return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ touch question2.cpp
sanjoy@SANJUVAI:~$ g++ question2.cpp
sanjoy@SANJUVAI:~$ ./a.out
 The name of System:Linux , system
 The version:#1 SMP Wed Nov 23 01:01:46 UTC 2022
 The Machine:x86_64
cpu family      : 6
model           : 126
```

13. Write a shell program to check a file is existed or not.

```
sanjoy@SANJUVAI:~$ cat>college.txt
Well Come to Midnapore City College
Dept. of Computer Science
^Z
```

Program:

```
echo "Enter your filename: "
read file
if [ -f $file ]
then
        echo "File is exist"
        echo "Contents of file"
        cat $file
else
        echo "File is not exist"
fi
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ bash file.sh
Enter your filename:
college.txt
File is exist
Contents of file
Well Come to Midnapore City College
Dept. of Computer Science
sanjoy@SANJUVAI:~$ bash file.sh
Enter your filename:
sanjoy.txt
File is not exist
```

14. Write a program to copy files using system calls.

Program:

```
sanjoy@SANJUVAI:~$ gedit copyfile.c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>
#include <stdlib.h>
int main(int argc, char *argv[])
 {
   int f1, f2;
   char buff[50];
   long int n;

 if(((f1 = open(argv[1], O_RDONLY)) == -1 || ((f2=open(argv[2],
O_CREAT |
 O_WRONLY | O_TRUNC, 0700))== 1)))
    {
      perror("problem in file");
      exit(1);
    }

 while((n=read(f1, buff, 50))>0)

    if(write(f2, buff, n)!=n)
       {
         perror("problem in writing");
         exit(3);
       }

    if(n==-1)
       {
         perror("problem in reading");
         exit(2);
       }

      close(f2);
      exit(0);
 }
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ cc copyfile.c
sanjoy@SANJUVAI:~$ cat>bsc.txt
This is BSc Computer Science File.
BSc Computer Science File is copy into BCA File.
^Z
[7]+  Stopped                 cat > bsc.txt
sanjoy@SANJUVAI:~$ ./a.out bsc.txt bca.txt
sanjoy@SANJUVAI:~$ cat bca.txt
This is BSc Computer Science File.
BSc Computer Science File is copy into BCA File.
```

15. Write a program to print file details including owner access permissions, file access time, where file name is given as argument.

Program:

```
#include<iostream>
using namespace std;
#include<stdlib.h>
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<unistd.h>
int main(int argc, char *argv[])
{
int i;
struct stat s;
if (argc < 2)
{
cout<<"\n enter filename in";
//exit();
}
for(i=1;i<argc; i++)
{
cout<<"File : "<<argv[i]<<"\n";
if(stat(argv[i],&s)<0)
cout<<"error in obtaining stats In";
else
{
cout<<"owner UID : "; cout<<s.st_uid; cout<<"\n";
```

```
cout<<"group ID :"; cout<<s.st_gid; cout<<"\n";
cout<<"Access permissions : "; cout<<s.st_mode; cout<<"\n";
cout<<"Access Time :" ;cout<<s.st_atime; cout<<"\n";
cout<<"File Size : "; cout<<s.st_size; cout<<"\n";
cout<<"File Size(in blocks) : "; cout<<s.st_blksize; cout<<"\n";
}
}
return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ cat>f1
Well, Come to Midnapore City College.
^Z
[3]+  Stopped                 cat > f1
sanjoy@SANJUVAI:~$ g++ test2.cpp
sanjoy@SANJUVAI:~$ ./a.out f1
File : f1
owner UID : 1000
group ID :1000
Access permissions : 33188
Access Time :1674936817
File Size : 37
File Size(in blocks) : 4096
```

16. Write a program in C to create a Zombie Process.

Program:

```
#include <stdlib.h>
#include <sys/types.h>
#include <unistd.h>
int main ()
{
  int pid_t,child_pid;
  child_pid = fork ();
  if (child_pid > 0) {
    sleep (20);
  }
  else {
```

```
        exit (0);
  }
  return 0;
}
```

Input and Output Section:

sanjoy@SANJUVAI:~$ gcc Zombie.c -o a.out

sanjoy@SANJUVAI:~$ ./a.out

Then command prompt will wait for some time (20 sec) and then again command prompt will appear later.

17. How to execute zombie and orphan process in a single C program?

Program:

```
#include<stdio.h>
#include<unistd.h>
int main(){
        int x = fork();
        if(x>0){
                printf("Inside Parent --- PID is : %d\n",getpid());
        }else if(x==0){
                sleep(5);
                x=fork();
                if(x>0){
                        printf("\n Inside child PID: %d & PID of parent: %d\n",getpid(),getppid());
                        while(1)
                                sleep(1);
                                printf("Inside child --- PID of Parent: %d\n",getppid());
                }else if(x==0){
                        printf("Inside grandchild process --- PID of parent: %d\n",getpid());
                }
        }
        return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc ZombieOrphan.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Inside Parent --- PID is : 955
sanjoy@SANJUVAI:~$
Inside grandchild process --- PID of parent: 957
 Inside child PID: 956 & PID of parent: 45
```

18. Write a program in C to print process id of a process and its parent process id also.

Program:

```
#include<stdio.h>
#include<unistd.h>
int main(){
        int p_id,p_pid;
        p_id=getpid();
        p_pid=getppid();
        printf("Process ID is %d\n",p_id);
        printf("Parent process ID %d\n",p_pid);
        return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc processId.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Process ID is 1051
Parent process ID 46
```

19. Write a program in C to duplicate a program's process using fork ().

Program:

```
#include<stdio.h>
        #include<unistd.h>
        int main(){
                int n1=fork();
                int n2=fork();

                if(n1>0 && n2>0){
                        printf("Parent\n");
                        printf("%d %d\n",n1,n2);
                        printf("My ID is %d\n",getpid());
                }else if(n1==0 && n2>0){
                        printf("1st Child\n");
                        printf("%d %d\n",n1,n2);
                        printf("My ID is %d\n",getpid());
                }else if(n1>0 && n2==0){
                        printf("2nd Child\n");
                        printf("%d %d\n",n1,n2);
                        printf("My ID is %d\n",getpid());
                }else{
                        printf("Third Child\n");
                        printf("%d %d\n",n1,n2);
                        printf("My ID is %d\n",getpid());
                }
                return 0;
        }
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc fork.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Parent
2nd Child
1139 1140
My ID is 1138
1139 0
My ID is 1140
Third Child
0 0
```

```
My ID is 1141
1st Child
sanjoy@SANJUVAI:~$ 0 1141
My ID is 1139
```

20. Write program to calculate sum of n numbers using thread library.

Program:

```
sanjoy@SANJUVAI:~$ gedit thread.c
#include <pthread.h>
#include <stdlib.h>
#include <stdio.h>
typedef struct data{
   int* arr;
   int thread_num;
} data;
int arrSize = 10;
void* halfSum(void* p){
   data* ptr = (data*)p;
   int n = ptr->thread_num;
   // Declare sum dynamically to return to join:
   int* thread_sum = (int*) calloc(1, sizeof(int));
   if(n == 0){
      for(int i = 0; i < arrSize/2; i++)
         thread_sum[0] = thread_sum[0] + ptr->arr[i];
   }
   else{
      for(int i = arrSize/2; i < arrSize; i++)
         thread_sum[0] = thread_sum[0] + ptr->arr[i];
   }

   pthread_exit(thread_sum);
}
int main(void){
   // Declare integer array [1,2,3,4,5,6,7,8,9,10]:
   int* int_arr = (int*) calloc(arrSize, sizeof(int));
   for(int i = 0; i < arrSize; i++)
      int_arr[i] = i + 1;
```

```
    // Declare arguments for both threads:
    data thread_data[2];
    thread_data[0].thread_num = 0;
    thread_data[0].arr = int_arr;
    thread_data[1].thread_num = 1;
    thread_data[1].arr = int_arr;
    // Declare thread IDs:
    pthread_t tid[2];
    // Start both threads:
    pthread_create(&tid[0], NULL, halfSum, &thread_data[0]);
    pthread_create(&tid[1], NULL, halfSum, &thread_data[1]);
    // Declare space for sum:
    int* sum0;
    int* sum1;
    // Retrieve sum of threads:
    pthread_join(tid[0], (void**)&sum0);
    pthread_join(tid[1], (void**)&sum1);
    printf("Sum of whole array = %i\n", *sum0 + *sum1);
    return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc thread.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Sum of whole array = 55
```

21. Write a program in C to implement FCFS CPU scheduling algorithm.

Program:

```
// C program for implementation of FCFS scheduling
#include<stdio.h>
// Function to find the waiting time for all processes
void findWaitingTime(int processes[], int n, int bt[], int wt[])
{
        // waiting time for first process is 0
        wt[0] = 0;

        // calculating waiting time
        for (int i = 1; i < n ; i++ )
```

```
                wt[i] = bt[i-1] + wt[i-1] ;
}

// Function to calculate turn around time
void findTurnAroundTime( int processes[], int n, int bt[], int wt[], int tat[])
{
        // calculating turnaround time by adding bt[i] + wt[i]
        for (int i = 0; i < n ; i++)
                tat[i] = bt[i] + wt[i];
}
//Function to calculate average time
void findavgTime( int processes[], int n, int bt[])
{
        int wt[n], tat[n], total_wt = 0, total_tat = 0;
        //Function to find waiting time of all processes
        findWaitingTime(processes, n, bt, wt);
        //Function to find turn around time for all processes
        findTurnAroundTime(processes, n, bt, wt, tat);
        //Display processes along with all details
        printf("Processes Burst time Waiting time Turn around time\n");
        // Calculate total waiting time and total turn around time
        for (int i=0; i<n; i++)
        {
                total_wt = total_wt + wt[i];
                total_tat = total_tat + tat[i];
                printf(" %d ",(i+1));
                printf("\t   %d", bt[i] );
                printf("\t\t %d",wt[i] );
                printf("\t\t %d\n",tat[i] );
        }
        int s=(float)total_wt / (float)n;
        int t=(float)total_tat / (float)n;
        printf("Average waiting time = %d",s);
        printf("\n");
        printf("Average turn around time = %d ",t);
}
int main()
```

```
{
        //process id's
        int processes[] = { 1, 2, 3};
        int n = sizeof processes / sizeof processes[0];

        //Burst time of all processes
        int burst_time[] = {10, 5, 8};

        findavgTime(processes, n, burst_time);
        return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc fcfs.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out
Processes Burst time Waiting time Turn around time
 1        10          0           10
 2        5           10          15
 3        8           15          23
Average waiting time = 8
Average turn around time = 16
```

22. Write program to implement Round Robin scheduling algorithm.

Program:

```
#include<stdio.h>
//#include<conio.h>
int main()
{
   // initlialize the variable name
   int i, NOP, sum=0,count=0, y, quant, wt=0, tat=0, at[10], bt[10],
temp[10];
   float avg_wt, avg_tat;
   printf(" Total number of process in the system: ");
   scanf("%d", &NOP);
   y = NOP; // Assign the number of process to variable y
```

```
// Use for loop to enter the details of the process like Arrival time and the Burst Time
for(i=0; i<NOP; i++)
{
printf("\n Enter the Arrival and Burst time of the Process[%d]\n", i+1);
printf(" Arrival time is: \t");  // Accept arrival time
scanf("%d", &at[i]);
printf(" \nBurst time is: \t"); // Accept the Burst time
scanf("%d", &bt[i]);
temp[i] = bt[i]; // store the burst time in temp array
}
// Accept the Time qunat
printf("Enter the Time Quantum for the process: \t");
scanf("%d", &quant);
// Display the process No, burst time, Turn Around Time and the waiting time
printf("\n Process No \t\t Burst Time \t\t TAT \t\t Waiting Time ");
for(sum=0, i = 0; y!=0; )
{
if(temp[i] <= quant && temp[i] > 0) // define the conditions
{
   sum = sum + temp[i];
   temp[i] = 0;
   count=1;
   }
   else if(temp[i] > 0)
   {
      temp[i] = temp[i] - quant;
      sum = sum + quant;
   }
   if(temp[i]==0 && count==1)
   {
      y--; //decrement the process no.
      printf("\nProcess No[%d] \t\t %d\t\t\t\t %d\t\t\t %d", i+1, bt[i], sum-at[i], sum-at[i]-bt[i]);
      wt = wt+sum-at[i]-bt[i];
      tat = tat+sum-at[i];
```

```
        count =0;
    }
    if(i==NOP-1)
    {
        i=0;
    }
    else if(at[i+1]<=sum)
    {
        i++;
    }
    else
    {
        i=0;
    }
}
// represents the average waiting time and Turn Around time
avg_wt = wt * 1.0/NOP;
avg_tat = tat * 1.0/NOP;
printf("\n Average Turn Around Time: \t%f", avg_wt);
printf("\n Average Waiting Time: \t%f", avg_tat);
//getch();
return 0;
}
```

Input and Output Section:

Total number of process in the system: 4

```
 Enter the Arrival and Burst time of the Process[1]
 Arrival time is:       0

Burst time is:  8

 Enter the Arrival and Burst time of the Process[2]
 Arrival time is:       1

Burst time is:  5
```

Enter the Arrival and Burst time of the Process[3]
Arrival time is: 2

Burst time is: 10

Enter the Arrival and Burst time of the Process[4]
Arrival time is: 3

Burst time is: 11
Enter the Time Quantum for the process: 6

Process No	Burst Time	TAT	Waiting Time
Process No[2]	5	10	5
Process No[1]	8	25	17
Process No[3]	10	27	17
Process No[4]	11	31	20

Average Turn Around Time: 14.750000
Average Waiting Time: 23.250000

23. Write program to implement SJF scheduling algorithm.

Program:

```
sanjoy@SANJUVAI:~$ gedit sjf.cpp
#include <bits/stdc++.h>
using namespace std;
//structure for every process
struct Process {
  int pid; // Process ID
  int bt; // Burst Time
  int art; // Arrival Time
};
void findTurnAroundTime(Process proc[], int n, int wt[], int tat[]) {
  for (int i = 0; i < n; i++)
  tat[i] = proc[i].bt + wt[i];
}
//waiting time of all process
void findWaitingTime(Process proc[], int n, int wt[]) {
  int rt[n];
```

```
  for (int i = 0; i < n; i++)
  rt[i] = proc[i].bt;
  int complete = 0, t = 0, minm = INT_MAX;
  int shortest = 0, finish_time;
  bool check = false;
  while (complete != n) {
    for (int j = 0; j < n; j++) {
      if ((proc[j].art <= t) && (rt[j] < minm) && rt[j] > 0) {
        minm = rt[j];
        shortest = j;
        check = true;
      }
    }
    if (check == false) {
      t++;
      continue;
    }
    // decrementing the remaining time
    rt[shortest]--;
    minm = rt[shortest];
    if (minm == 0)
      minm = INT_MAX;
      // If a process gets completely
      // executed
      if (rt[shortest] == 0) {
        complete++;
        check = false;
        finish_time = t + 1;
        // Calculate waiting time
        wt[shortest] = finish_time -
        proc[shortest].bt -
        proc[shortest].art;
        if (wt[shortest] < 0)
          wt[shortest] = 0;
      }
      // Increment time
      t++;
  }
}
// Function to calculate average time
```

```
void findavgTime(Process proc[], int n) {
  int wt[n], tat[n], total_wt = 0,
  total_tat = 0;
  // Function to find waiting time of all
  // processes
  findWaitingTime(proc, n, wt);
  // Function to find turn around time for
  // all processes
  findTurnAroundTime(proc, n, wt, tat);
  cout << "Processes " << " Burst time " << " Waiting time " << " Turn
around time\n";
  for (int i = 0; i < n; i++) {
    total_wt = total_wt + wt[i];
    total_tat = total_tat + tat[i];
    cout << " " << proc[i].pid << "\t\t" << proc[i].bt << "\t\t " << wt[i] <<
"\t\t " << tat[i] << endl;
  }
  cout << "\nAverage waiting time = " << (float)total_wt / (float)n;
  cout << "\nAverage turn around time = " << (float)total_tat / (float)n;
}
int main() {
  Process proc[] = { { 1, 5, 1 }, { 2, 3, 1 }, { 3, 6, 2 }, { 4, 5, 3 } };
  int n = sizeof(proc) / sizeof(proc[0]);
  findavgTime(proc, n);
  return 0;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ sjf.cpp
sanjoy@SANJUVAI:~$ ./a.out
Processes  Burst time  Waiting time  Turn around time
 1          5          3          8
 2          3          0          3
 3          6          12          18
 4          5          6          11

Average waiting time = 5.25
Average turn around time = 10
```

24. Write a program to implement first-fit, best-fit and worst-fit allocation strategies.

First-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ gedit firstfit.c
// C implementation of First - Fit algorithm
#include<stdio.h>

// Function to allocate memory to blocks as per First fit algorithm
void firstFit(int blockSize[], int m, int processSize[], int n)
{
    int i, j;
    // Stores block id of the block allocated to a process
    int allocation[n];

    // Initially no block is assigned to any process
    for(i = 0; i < n; i++)
    {
            allocation[i] = -1;
    }
    for (i = 0; i < n; i++)        //here, n -> number of processes
    {
            for (j = 0; j < m; j++)       //here, m -> number of blocks
            {
                    if (blockSize[j] >= processSize[i])
                    {
                            // allocating block j to the ith process
                            allocation[i] = j;

                            // Reduce available memory in this block.
                            blockSize[j] -= processSize[i];

                            break; //go to the next process in the queue
                    }
            }
    }
```

```
    printf("\nProcess No.\tProcess Size\tBlock no.\n");
    for (int i = 0; i < n; i++)
    {
            printf(" %i\t\t", i+1);
            printf("%i\t\t", processSize[i]);
            if (allocation[i] != -1)
                    printf("%i", allocation[i] + 1);
            else
                    printf("Not Allocated");
            printf("\n");
    }
}

int main()
{
    int m; //number of blocks in the memory
    int n; //number of processes in the input queue
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    m = sizeof(blockSize) / sizeof(blockSize[0]);
    n = sizeof(processSize) / sizeof(processSize[0]);

    firstFit(blockSize, m, processSize, n);

    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ gcc firstfit.c -o a.out
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             2
 2              417             5
 3              112             2
 4              426             Not Allocated
```

Best-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ touch bestfit.cpp
// C++ implementation of Best - Fit algorithm
#include<iostream>
using namespace std;

// Method to allocate memory to blocks as per Best fit algorithm
void bestFit(int blockSize[], int m, int processSize[], int n)
{
    // Stores block id of the block allocated to a process
    int allocation[n];

    // Initially no block is assigned to any process
    for (int i = 0; i < n; i++)
            allocation[i] = -1;

    // pick each process and find suitable blocks according to its size ad
assign to it
    for (int i = 0; i < n; i++)
    {
            // Find the best fit block for current process
            int bestIdx = -1;
            for (int j = 0; j < m; j++)
            {
                    if (blockSize[j] >= processSize[i])
                    {
                            if (bestIdx == -1)
                                    bestIdx = j;
                            else if (blockSize[bestIdx] > blockSize[j])
                                    bestIdx = j;
                    }
            }

            // If we could find a block for current process
            if (bestIdx != -1)
            {
                    // allocate block j to p[i] process
                    allocation[i] = bestIdx;

                    // Reduce available memory in this block.
```

```
                blockSize[bestIdx] -= processSize[i];
            }
    }

    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
            cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
            if (allocation[i] != -1)
                cout << allocation[i] + 1;
            else
                cout << "Not Allocated";
            cout << endl;
    }
}
int main()
{
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    int m = sizeof(blockSize) / sizeof(blockSize[0]);
    int n = sizeof(processSize) / sizeof(processSize[0]);

    bestFit(blockSize, m, processSize, n);

    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ bestfit.cpp
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             4
 2              417             2
 3              112             3
 4              426             5
```

Worst-fit allocation strategies:

Program:

```
sanjoy@SANJUVAI:~$ gedit worstfit.cpp
// C++ implementation of worst - Fit algorithm
#include<bits/stdc++.h>
using namespace std;

// Function to allocate memory to blocks as per worst fit
// algorithm
void worstFit(int blockSize[], int m, int processSize[],

    int n)
{
    // Stores block id of the block allocated to a
    // process
    int allocation[n];

    // Initially no block is assigned to any process
    memset(allocation, -1, sizeof(allocation));

    // pick each process and find suitable blocks
    // according to its size ad assign to it
    for (int i=0; i<n; i++)
    {
            // Find the best fit block for current process
            int wstIdx = -1;
            for (int j=0; j<m; j++)
            {
                    if (blockSize[j] >= processSize[i])
                    {
                            if (wstIdx == -1)
                                    wstIdx = j;
                            else if (blockSize[wstIdx] < blockSize[j])
                                    wstIdx = j;
                    }
            }

            // If we could find a block for current process
            if (wstIdx != -1)
            {
                    // allocate block j to p[i] process
                    allocation[i] = wstIdx;
```

```
                // Reduce available memory in this block.
                blockSize[wstIdx] -= processSize[i];
            }
    }

    cout << "\nProcess No.\tProcess Size\tBlock no.\n";
    for (int i = 0; i < n; i++)
    {
            cout << " " << i+1 << "\t\t" << processSize[i] << "\t\t";
            if (allocation[i] != -1)
                cout << allocation[i] + 1;
            else
                cout << "Not Allocated";
            cout << endl;
    }
}

// Driver code
int main()
{
    int blockSize[] = {100, 500, 200, 300, 600};
    int processSize[] = {212, 417, 112, 426};
    int m = sizeof(blockSize)/sizeof(blockSize[0]);
    int n = sizeof(processSize)/sizeof(processSize[0]);

    worstFit(blockSize, m, processSize, n);

    return 0 ;
}
```

Input and Output Section:

```
sanjoy@SANJUVAI:~$ g++ worstfit.cpp
sanjoy@SANJUVAI:~$ ./a.out

Process No.     Process Size    Block no.
 1              212             5
 2              417             2
 3              112             5
 4              426             Not Allocated
```

GROUP-A: COMPUTER NETWORK LABORATORY MANUAL
(Course: BCA 2297)

Assignments

1. Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.
2. Simulate and implement stop and wait protocol for noiseless channel.
3. Simulate and implement stop and wait protocol for noisy channel.
4. Simulate and implement go back n sliding window protocol.
5. Simulate and implement selective repeat sliding window protocol.
6. Simulate and implement Hamming Code error correction.
7. Simulate and implement distance vector routing algorithm.
8. Simulate and implement Dijkstra algorithm for shortest path routing.

1. **Simulate Cyclic Redundancy Check (CRC) error detection algorithm for noisy channel.**

Program:

```
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
   int i,j,k,l;

   //Get Frame
   int fs;
   cout<<"\n Enter Frame size: ";
   cin>>fs;

   int f[20];

   cout<<"\n Enter Frame:";
   for(i=0;i<fs;i++)
   {
      cin>>f[i];
   }

   //Get Generator
   int gs;
   cout<<"\n Enter Generator size: ";
   cin>>gs;

   int g[20];

   cout<<"\n Enter Generator:";
   for(i=0;i<gs;i++)
   {
      cin>>g[i];
   }

   cout<<"\n Sender Side:";
   cout<<"\n Frame: ";
   for(i=0;i<fs;i++)
   {
      cout<<f[i];
   }
```

```
cout<<"\n Generator :";
for(i=0;i<gs;i++)
{
   cout<<g[i];
}

//Append 0's
int rs=gs-1;
cout<<"\n Number of 0's to be appended: "<<rs;
for (i=fs;i<fs+rs;i++)
{
   f[i]=0;
}

int temp[20];
for(i=0;i<20;i++)
{
   temp[i]=f[i];
}

cout<<"\n Message after appending 0's :";
for(i=0; i<fs+rs;i++)
{
   cout<<temp[i];
}

//Division
for(i=0;i<fs;i++)
{
   j=0;
   k=i;
   //check whether it is divisible or not
   if (temp[k]>=g[j])
   {
      for(j=0,k=i;j<gs;j++,k++)
      {
         if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
         {
            temp[k]=0;
         }
         else
         {
            temp[k]=1;
```

```
            }
        }
    }
}

//CRC
int crc[15];
for(i=0,j=fs;i<rs;i++,j++)
{
    crc[i]=temp[j];
}

cout<<"\n CRC bits: ";
for(i=0;i<rs;i++)
{
    cout<<crc[i];
}

cout<<"\n Transmitted Frame: ";
int tf[15];
for(i=0;i<fs;i++)
{
    tf[i]=f[i];
}
for(i=fs,j=0;i<fs+rs;i++,j++)
{
    tf[i]=crc[j];
}
for(i=0;i<fs+rs;i++)
{
    cout<<tf[i];
}

cout<<"\n Receiver side : ";
cout<<"\n Received Frame: ";
for(i=0;i<fs+rs;i++)
{
    cout<<tf[i];
}

for(i=0;i<fs+rs;i++)
{
    temp[i]=tf[i];
```

```
}

//Division
for(i=0;i<fs+rs;i++)
{
   j=0;
   k=i;
   if (temp[k]>=g[j])
   {
      for(j=0,k=i;j<gs;j++,k++)
      {
         if((temp[k]==1 && g[j]==1) || (temp[k]==0 && g[j]==0))
         {
            temp[k]=0;
         }
         else
         {
            temp[k]=1;
         }
      }
   }
}

cout<<"\n Reaminder: ";
int rrem[15];
for (i=fs,j=0;i<fs+rs;i++,j++)
{
   rrem[j]= temp[i];
}
for(i=0;i<rs;i++)
{
   cout<<rrem[i];
}

int flag=0;
for(i=0;i<rs;i++)
{
   if(rrem[i]!=0)
   {
      flag=1;
   }
}
```

```
    if(flag==0)
    {
       cout<<"\n Since Remainder Is 0 Hence Message Transmitted From
Sender To Receiver Is Correct";
    }
    else
    {
       cout<<"\n Since Remainder Is Not 0 Hence Message Transmitted From
Sender To Receiver Contains Error";
    }
    getch();
}
```

Input and Output Section:

Enter Frame size: 8

Enter Frame:1 0 0 1 1 0 0 1

Enter Generator size: 4

Enter Generator:1 0 0 1

Sender Side:

Frame: 10011001

Generator :1001

Number of 0's to be appended: 3

Message after appending 0's :10011001000

CRC bits: 000

Transmitted Frame: 10011001000

Receiver side :

Received Frame: 10011001000

Remainder: 000

Since Remainder Is 0 Hence Message Transmitted From Sender To Receiver Is Correct

2. **Simulate and implement stop and wait protocol for noiseless channel.**

Program:

```
#include<iostream>
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<dos.h>
using namespace std;
#define time 5
#define max_seq 1
#define tot_pack 5
int randn(int n)
{
   return rand()%n + 1;
}
typedef struct
{
      int data;
}packet;
typedef struct
{
       int kind;
       int seq;
       int ack;
       packet info;
}frame;
typedef enum{ frame_arrival,error,time_out}event_type;
frame data1;
```

```
//creating prototype
void from_network_layer(packet *);
void to_physical_layer(frame *);
void to_network_layer(packet *);
void from_physical_layer(frame*);
void sender();
void receiver();
void wait_for_event_sender(event_type *);
void wait_for_event_receiver(event_type *);
//end
#define inc(k) if(k<max_seq)k++;else k=0;
int i=1;
char turn;
int disc=0;
int main()
{
  while(!disc)
  {  sender();
    // delay(400);
     receiver();
  }
   getchar();
}
void sender()
{
    static int frame_to_send=0;
    static frame s;
    packet buffer;
    event_type event;
```

```
    static int flag=0;       //first place
    if (flag==0)
    {
  from_network_layer(&buffer);
  s.info=buffer;
  s.seq=frame_to_send;
  cout<<"\nsender information \t"<<s.info.data<<"\n";
  cout<<"\nsequence no. \t"<<s.seq;
  turn='r';
  to_physical_layer(&s);
  flag=1;
    }
 wait_for_event_sender(&event);
 if(turn=='s')
 {
    if(event==frame_arrival)
     {
     from_network_layer(&buffer);
     inc(frame_to_send);
     s.info=buffer;
     s.seq=frame_to_send;
     cout<<"\nsender information \t"<<s.info.data<<"\n";
     cout<<"\nsequence no. \t"<<s.seq<<"\n";
     getch();
     turn='r';
     to_physical_layer(&s);
     }
 }
}                //end of sender function
```

```
void from_network_layer(packet *buffer)
{
    (*buffer).data=i;
    i++;
}                       //end of from network layer function
void to_physical_layer(frame *s)
{
    data1=*s;
}           //end of to physical layer function
void wait_for_event_sender(event_type *e)
{
     static int timer=0;
     if(turn=='s')
     {   timer++;
   //timer=0;
   return ;
  }
  else                      //event is frame arrival
    {
      timer=0;
      *e=frame_arrival;
    }
}           //end of wait for event function
void receiver()
{
    static int frame_expected=0;
    frame s,r;
    event_type event;
    wait_for_event_receiver(&event);
```

```
    if(turn=='r')
    {  if(event==frame_arrival)
   {
        from_physical_layer(&r);
        if(r.seq==frame_expected)
   {
     to_network_layer(&r.info);
     inc (frame_expected);
   }
   else
   cout<<"\nReceiver :Acknowledgement resent \n";
   getch();
   turn='s';
   to_physical_layer(&s);
     }
    }
}                    //end of receiver function
void wait_for_event_receiver(event_type *e)
{
    if(turn=='r')
    {
  *e=frame_arrival;
    }
}
void from_physical_layer(frame *buffer)
{
   *buffer=data1;
}
void to_network_layer(packet *buffer)
```

```
{
    cout<<"\nReceiver : packet received \t"<< i-1;
    cout<<"\n Acknowledgement  sent \t";
    getch();
    if(i>tot_pack)
     { disc=1;
 cout<<"\ndiscontinue\n";
     }
}       //end of network layer function
```

Input and Output Section:

sender information 1

sequence no. 0

Receiver : packet received 1

Acknowledgement sent

sender information 2

sequence no. 1

Receiver : packet received 2

Acknowledgement sent

sender information 3

sequence no. 0

Receiver : packet received 3

Acknowledgement sent

sender information 4

sequence no. 1

Receiver : packet received 4

Acknowledgement sent

sender information 5

sequence no. 0

Receiver : packet received 5

Acknowledgement sent

Discontinue

3. Simulate and implement stop and wait protocol for noisy channel.

Program:

```
#include<iostream>
#include <time.h>
#include <cstdlib>
#include<ctime>
#include <unistd.h>
using namespace std;
class timer {
   private:
    unsigned long begTime;
   public:
    void start() {
     begTime = clock();
    }
  unsigned long elapsedTime() {
     return ((unsigned long) clock() - begTime) / CLOCKS_PER_SEC;
```

```
    }
   bool isTimeout(unsigned long seconds) {
      return seconds >= elapsedTime();
     }
};
int main()
{
 int frames[] = {1,2,3,4,5,6,7,8,9,10};
 unsigned long seconds = 5;
 srand(time(NULL));
 timer t;
 cout<<"Sender has to send frames : ";
 for(int i=0;i<10;i++)
     cout<<frames[i]<<" ";
 cout<<endl;
 int count = 0;
 bool delay = false;
 cout<<endl<<"Sender\t\t\t\tReceiver"<<endl;
 do
 {
     bool timeout = false;
     cout<<"Sending Frame : "<<frames[count];
     cout.flush();
     cout<<"\t\t";
     t.start();
     if(rand()%2)
     {
         int to = 24600 + rand()%(64000 - 24600)  + 1;
         for(int i=0;i<64000;i++)
```

```
        for(int j=0;j<to;j++) {}
}
if(t.elapsedTime() <= seconds)
{
    cout<<"Received Frame : "<<frames[count]<<" ";
    if(delay)
    {
        cout<<"Duplicate";
        delay = false;
    }
    cout<<endl;
    count++;
}
else
{
    cout<<"---"<<endl;
    cout<<"Timeout"<<endl;
    timeout = true;
}
t.start();
if(rand()%2 || !timeout)
{
    int to = 24600 + rand()%(64000 - 24600)  + 1;
    for(int i=0;i<64000;i++)
        for(int j=0;j<to;j++) {}
    if(t.elapsedTime() > seconds )
    {
        cout<<"Delayed Ack"<<endl;
        count--;
```

```
            delay = true;
        }
        else if(!timeout)
            cout<<"Acknowledgement : "<<frames[count]-1<<endl;
    }
  }while(count!=10);
  return 0;
 }
```

Input and Output Section:

Sender has to send frames : 1 2 3 4 5 6 7 8 9 10

Sender	Receiver
Sending Frame : 1	---
Timeout	
Sending Frame : 1	Received Frame : 1
Acknowledgement : 1	
Sending Frame : 2	Received Frame : 2
Acknowledgement : 2	
Sending Frame : 3	Received Frame : 3
Acknowledgement : 3	
Sending Frame : 4	---
Timeout	
Sending Frame : 4	Received Frame : 4
Acknowledgement : 4	
Sending Frame : 5	---
Timeout	

Sending Frame : 5 Received Frame : 5

Acknowledgement : 5

Sending Frame : 6 Received Frame : 6

Acknowledgement : 6

Sending Frame : 7 Received Frame : 7

Delayed Ack

Sending Frame : 7 ---

Timeout

Sending Frame : 7 Received Frame : 7 Duplicate

Delayed Ack

Sending Frame : 7 Received Frame : 7 Duplicate

Acknowledgement : 7

Sending Frame : 8 Received Frame : 8

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 Received Frame : 8 Duplicate

Delayed Ack

Sending Frame : 8 ---

Timeout

Sending Frame : 8 ---

Timeout

Sending Frame : 8 Received Frame : 8 Duplicate

Acknowledgement : 8

Sending Frame : 9 ---

Timeout

Sending Frame : 9 Received Frame : 9

Acknowledgement : 9

Sending Frame : 10 Received Frame : 10

Delayed Ack

Sending Frame : 10 Received Frame : 10 Duplicate

Delayed Ack

Sending Frame : 10 Received Frame : 10 Duplicate

Acknowledgement : 0

4. Simulate and implement go back n sliding window protocol.

Program:

```
#include<iostream>
#include<ctime>
#include<cstdlib>
using namespace std;
int main()
{
 int nf,N;
 int no_tr=0;
 srand(time(NULL));
 cout<<"Enter the number of frames : ";
```

```
cin>>nf;
cout<<"Enter the Window Size : ";
cin>>N;
int i=1;
while(i<=nf)
{
   int x=0;
   for(int j=i;j<i+N && j<=nf;j++)
   {
      cout<<"Sent Frame "<<j<<endl;
      no_tr++;
   }
   for(int j=i;j<i+N && j<=nf;j++)
   {
      int flag = rand()%2;
      if(!flag)
         {
            cout<<"Acknowledgment for Frame "<<j<<endl;
            x++;
         }
      else
         {   cout<<"Frame "<<j<<" Not Received"<<endl;
            cout<<"Retransmitting Window"<<endl;
            break;
         }
   }
   cout<<endl;
   i+=x;
}
```

```
 cout<<"Total number of transmissions : "<<no_tr<<endl;
 return 0;
}
```

Input and Output Section:

Enter the number of frames : 9

Enter the Window Size : 3

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Frame 1 Not Received

Retransmitting Window

Sent Frame 1

Sent Frame 2

Sent Frame 3

Acknowledgment for Frame 1

Frame 2 Not Received

Retransmitting Window

Sent Frame 2

Sent Frame 3

Sent Frame 4

Acknowledgment for Frame 2

Frame 3 Not Received

Retransmitting Window

Sent Frame 3

Sent Frame 4

Sent Frame 5

Acknowledgment for Frame 3

Acknowledgment for Frame 4

Acknowledgment for Frame 5

Sent Frame 6

Sent Frame 7

Sent Frame 8

Acknowledgment for Frame 6

Frame 7 Not Received

Retransmitting Window

Sent Frame 7

Sent Frame 8

Sent Frame 9

Acknowledgment for Frame 7

Acknowledgment for Frame 8

Frame 9 Not Received

Retransmitting Window

Sent Frame 9

Acknowledgment for Frame 9

Total number of transmissions : 28

5. Simulate and implement selective repeat sliding window protocol.

Program:

```
#include<iostream>
using namespace std;
#include<conio.h>
#include<stdlib.h>
#include<time.h>
#include<math.h>
#define TOT_FRAMES 500
#define FRAMES_SEND 10
class sel_repeat
{
private:
int fr_send_at_instance;
int arr[TOT_FRAMES];
int send[FRAMES_SEND];
int rcvd[FRAMES_SEND];

char rcvd_ack[FRAMES_SEND];
int sw;
int rw;       //tells expected frame
public:
```

```
void input();
void sender(int);
void receiver(int);
};
void sel_repeat::input()
{
int n;     //no. of bits for the frame
int m;    //no. of frames from n bits
int i;
cout<<"Enter the no. of bits for the sequence no. : ";
cin>>n;
m=pow(2,n);
int t=0;
fr_send_at_instance=(m/2);
for(i=0;i<TOT_FRAMES;i++)
{
arr[i]=t;
t=(t+1)%m;
}

for(i=0;i<fr_send_at_instance;i++)
{
send[i]=arr[i];
rcvd[i]=arr[i];
rcvd_ack[i]='n';
}
rw=sw=fr_send_at_instance;
sender(m);
}
```

```
void sel_repeat::sender(int m)
{
for(int i=0;i<fr_send_at_instance;i++)
{
if(rcvd_ack[i]=='n')
cout<<"SENDER : Frame "<<send[i]<<" is sent\n";
}
receiver(m);
}
void sel_repeat::receiver(int m)
{
time_t t;
int f;
int j;
int f1;
int a1;
char ch;
srand((unsigned)time(&t));
for(int i=0;i<fr_send_at_instance;i++)
{
if(rcvd_ack[i]=='n')
{
f=rand()%10;
//if f=5 frame is discarded for some reason
//else frame is correctly recieved
if(f!=5)
{
for(int j=0;j<fr_send_at_instance;j++)
if(rcvd[j]==send[i])
```

```
{
cout<<"reciever:Frame"<<rcvd[j]<<"recieved correctly\n";
rcvd[j]=arr[rw];
rw=(rw+1)%m;
break;
}
int j;
if(j==fr_send_at_instance)
cout<<"reciever:Duplicate frame"<<send[i]<<"discarded\n";
a1=rand()%5;
//if al==3 then ack is lost
//else recieved
if(a1==3)
{
cout<<"(acknowledgement "<<send[i]<<" lost)\n";
cout<<"(sender timeouts-->Resend the frame)\n";
rcvd_ack[i]='n';
}
else
{
cout<<"(acknowledgement "<<send[i]<<" recieved)\n";
rcvd_ack[i]='p';
}
}
else
{int ld=rand()%2;
//if =0 then frame damaged
//else frame lost
if(ld==0)
```

```
{
cout<<"RECEIVER : Frame "<<send[i]<<" is damaged\n";
cout<<"RECEIVER : Negative Acknowledgement "<<send[i]<<" sent\n";
}
else
{
cout<<"RECEIVER : Frame "<<send[i]<<" is lost\n";
cout<<"(SENDER TIMEOUTS-->RESEND THE FRAME)\n";
}
rcvd_ack[i]='n';
}
}
}
for(int j=0;j<fr_send_at_instance;j++)
{
if(rcvd_ack[j]=='n')
break;
}
int i=0;
for(int k=j;k<fr_send_at_instance;k++)
{
send[i]=send[k];

if(rcvd_ack[k]=='n')
rcvd_ack[i]='n';
else
rcvd_ack[i]='p';
i++;
```

```
}
if(i!=fr_send_at_instance)
{
for(int k=i;k<fr_send_at_instance;k++)
{
send[k]=arr[sw];
sw=(sw+1)%m;
rcvd_ack[k]='n';
}
}
cout<<"Want to continue";
cin>>ch;
cout<<"\n";
if(ch=='y')
sender(m);
else
exit(0);

}
int main()
{
sel_repeat sr;
sr.input();
}
```

Input and Output Section

Enter the no. of bits for the sequence no. : 4

SENDER : Frame 0 is sent

SENDER : Frame 1 is sent

SENDER : Frame 2 is sent

SENDER : Frame 3 is sent

SENDER : Frame 4 is sent

SENDER : Frame 5 is sent

SENDER : Frame 6 is sent

SENDER : Frame 7 is sent

reciever:Frame0recieved correctly

(acknowledgement 0 recieved)

reciever:Frame1recieved correctly

(acknowledgement 1 recieved)

reciever:Frame2recieved correctly

(acknowledgement 2 recieved)

reciever:Frame3recieved correctly

(acknowledgement 3 recieved)

reciever:Frame4recieved correctly

(acknowledgement 4 lost)

(sender timeouts-->Resend the frame)

reciever:Frame5recieved correctly

(acknowledgement 5 lost)

(sender timeouts-->Resend the frame)

reciever:Frame6recieved correctly

(acknowledgement 6 recieved)

reciever:Frame7recieved correctly

(acknowledgement 7 recieved)

Want to continue n

6. Simulate and implement Hamming Code error correction.

Program:

```
#include<iostream>
#include<cmath>
#include<string>
using namespace std;
 class Hamming

{

 string message;
 int codeword[50],temp[50];
 int n,check;
 char parity;
public:
 Hamming()
 {
    parity = 'E';
    message = "";
    n=check=0;
    for(int i=0;i<50;i++)
    {
       temp[i]=codeword[i]=0;
```

```
    }
}

void generate()
{
    do
    {
        cout<<"Enter the message in binary : ";
        cin>>message;
    }while(message.find_first_not_of("01") != string::npos);
    n=message.size();
    cout<<"Odd(O)/Even(E) Parity ? ";
    cin>>parity;
    for(unsigned int i=0;i<message.size();i++)
    {
        if(message[i] == '1')
            temp[i+1]=1;
        else
            temp[i+1]=0;
    }
    computeCode();
}
 void computeCode()
{
    check = findr();
    cout<<"Number of Check Bits : "<<check<<endl;
    cout<<"Number of Bits in Codeword : "<<n+check<<endl;
    for(int i=(n+check),j=n;i>0;i--)
    {
```

```
        if((i & (i - 1)) != 0)
            codeword[i] = temp[j--];
        else
            codeword[i] = setParity(i);
    }
    cout<<"Parity Bits - ";
    for(int i=0;i<check;i++)
     cout<<"P"<<pow(2,i)<<" : "<<codeword[(int)pow(2,i)]<<"\t";
    cout<<endl;
    cout<<"Codeword :"<<endl;
    for(int i=1;i<=(n+check);i++)
        cout<<codeword[i]<<" ";
    cout<<endl;
}
 int findr()
{
    for(int i=1;;i++)
    {
        if(n+i+1 <= pow(2,i))
            return i;
    } }
 int setParity(int x)
{
    bool flag = true;
    int bit;
    if(x == 1)
    {
        bit = codeword[x+2];
        for(int j=x+3;j<=(n+check);j++)
```

```
        {
            if(j%2)
            {
                bit ^= codeword[j];
            }
        }
    }
    else
    {
        bit = codeword[x+1];
        for(int i=x;i<=(n+check);i++)
        {
            if(flag)
            {
                if(i==x || i==x+1)
                    bit = codeword[x+1];
                else
                    bit ^= codeword[i];
            }
            if((i+1)%x == 0)
                flag = !flag;
        }
    }
    if(parity == 'O' || parity == 'o')
        return !bit;
    else
        return bit;
}
    void correct()
```

```
{
   do
   {
      cout<<"Enter the received codeword : ";
      cin>>message;
   }while(message.find_first_not_of("01") != string::npos);
   for(unsigned int i=0;i<message.size();i++)
   {
      if(message[i] == '1')
         codeword[i+1]=1;
      else
         codeword[i+1]=0;
   }
   detect();
}
void detect()
{
   int position = 0;
   cout<<"Parity Bits - ";
   for(int i=0;i<check;i++)
   {
      bool flag = true;
      int x = pow(2,i);
      int bit = codeword[x];
      if(x == 1)
      {
         for(int j=x+1;j<=(n+check);j++)
         {
            if(j%2)
```

```
            {
                bit ^= codeword[j];
            }
        }
    }
    else
    {
        for(int k=x+1;k<=(n+check);k++)
        {
            if(flag)
            {
                bit ^= codeword[k];
            }
            if((k+1)%x == 0)
                flag = !flag;
        }
    }
    cout<<"P"<<x<<": "<<bit<<"\t";
    if((parity=='E' || parity == 'e') && bit==1)
        position += x;
    if((parity=='O' || parity == 'o') && bit==0)
        position += x;
}
cout<<endl<<"Received Codeword :"<<endl;
for(int i=1;i<=(n+check);i++)
    cout<<codeword[i]<<" ";
cout<<endl;
if(position != 0)
{
```

```
        cout<<"Error at bit : "<<position<<endl;
        codeword[position] = !codeword[position];
        cout<<"Corrected Codeword : "<<endl;
        for(int i=1;i<=(n+check);i++)
            cout<<codeword[i]<<" ";
        cout<<endl;
    }
    else
        cout<<"No Error in Received code."<<endl;
    cout<<"Received Message is : ";
    for(int i=1;i<=(n+check);i++)
        if((i & (i - 1)) != 0)
            cout<<codeword[i]<<" ";
    cout<<endl;
 }
};
 int main()
{
 char choice;
 do
 {
    Hamming a;
    cout<<"At Sender's side : "<<endl;
    a.generate();
    cout<<endl<<"At Receiver's Side : "<<endl;
    a.correct();
    cout<<endl<<"Enter another code ? (Y/N) : ";
    cin>>choice;
    cout<<endl;
```

```
 }while(choice == 'y' || choice == 'Y');
 return 0;
}
```

Input and Output Section:

At Sender's side :

Enter the message in binary : 1001101

Odd(O)/Even(E) Parity ? E

Number of Check Bits : 4

Number of Bits in Codeword : 11

Parity Bits - P1 : 0 P2 : 1 P4 : 1 P8 : 0

Codeword :

0 1 1 1 0 0 1 0 1 0 1

At Receiver's Side :

Enter the received codeword : 01110010101

Parity Bits - P1: 0 P2: 0 P4: 0 P8: 0

Received Codeword :

0 1 1 1 0 0 1 0 1 0 1

No Error in Received code.

Received Message is : 1 0 0 1 1 0 1

Enter another code ? (Y/N) : N

7. Simulate and implement distance vector routing algorithm.

Program:

```
#include<stdio.h>
#include<iostream>
using namespace std;
struct node
{
   unsigned dist[6];
   unsigned from[6];
}DVR[10];
int main()
{
   cout<<"\n\n PROGRAM TO IMPLEMENT DISTANCE VECTOR ROUTING ALGORITHM ";
   int costmat[6][6];
   int nodes, i, j, k;
   cout<<"\n\n Enter the number of nodes : ";
   cin>>nodes; //Enter the nodes
   cout<<"\n Enter the cost matrix : \n" ;
   for(i = 0; i < nodes; i++)
    {
      for(j = 0; j < nodes; j++)
      {
         cin>>costmat[i][j];
         costmat[i][i] = 0;
         DVR[i].dist[j] = costmat[i][j]; //initialise the distance equal to cost matrix
         DVR[i].from[j] = j;
```

```
        }
    }
        for(i = 0; i < nodes; i++) //We choose arbitary vertex k and we
calculate the
        //direct distance from the node i to k using the cost matrix and add
the distance from k to node j
        for(j = i+1; j < nodes; j++)
        for(k = 0; k < nodes; k++)
            if(DVR[i].dist[j] > costmat[i][k] + DVR[k].dist[j])
            {   //We calculate the minimum distance
                DVR[i].dist[j] = DVR[i].dist[k] + DVR[k].dist[j];
                DVR[j].dist[i] = DVR[i].dist[j];
                DVR[i].from[j] = k;
                DVR[j].from[i] = k;

            }
        for(i = 0; i < nodes; i++)
        {
            cout<<"\n\n For router: "<<i+1;
            for(j = 0; j < nodes; j++)
                cout<<"\t\n node "<<j+1<<" via "<<DVR[i].from[j]+1<<"
Distance "<<DVR[i].dist[j];
        }
    cout<<" \n\n ";
    return 0;
}
```

Input and Output Section:

PROGRAM TO IMPLEMENT DISTANCE VECTOR ROUTING ALGORITHM

Enter the number of nodes : 3

Enter the cost matrix :

0

2

7

2

0

1

7

1

0

For router: 1

node 1 via 1 Distance 0

node 2 via 2 Distance 2

node 3 via 2 Distance 3

For router: 2

node 1 via 1 Distance 2

node 2 via 2 Distance 0

node 3 via 3 Distance 1

For router: 3

node 1 via 2 Distance 3

node 2 via 2 Distance 1

node 3 via 3 Distance 0

8. Simulate and implement Dijkstra algorithm for shortest path routing.

Program:

```
#include<iostream>
#include<conio.h>
#include<stdio.h>
using namespace std;
int shortest(int ,int);
int cost[10][10],dist[20],i,j,n,k,m,S[20],v,totcost,path[20],p;
main()
{
int c;
cout <<"enter no of vertices";
cin >> n;
cout <<"enter no of edges";
```

```
cin >>m;
cout <<"\nenter\nEDGE Cost\n";
for(k=1;k<=m;k++)
{
cin >> i >> j >>c;
cost[i][j]=c;
}
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
if(cost[i][j]==0)
cost[i][j]=31999;
cout <<"enter initial vertex";
cin >>v;
cout << v<<"\n";
shortest(v,n);
 }

int shortest(int v,int n)
{
int min;
for(i=1;i<=n;i++)
{
S[i]=0;
dist[i]=cost[v][i];
}
path[++p]=v;
S[v]=1;
dist[v]=0;
for(i=2;i<=n-1;i++)
```

```
{
k=-1;
min=31999;
for(j=1;j<=n;j++)
{
if(dist[j]<min && S[j]!=1)
{
min=dist[j];
k=j;
}
}
if(cost[v][k]<=dist[k])
p=1;
path[++p]=k;
for(j=1;j<=p;j++)
cout<<path[j];
cout <<"\n";
//cout <<k;
S[k]=1;
for(j=1;j<=n;j++)
if(cost[k][j]!=31999 && dist[j]>=dist[k]+cost[k][j] && S[j]!=1)
 dist[j]=dist[k]+cost[k][j];
}
}
```

Input and Output Section:

enter no of vertices5

enter no of edges7

enter

EDGE Cost

1

2

10

1

3

2

1

5

100

2

4

3

3

25

5

4

5

5

4

3

15

enter initial vertex1

1

13

12

124