Criterion : II - Teaching- Learning and Evaluation
Metric : 2.3 Teaching Learning Process

PROBLEM SOLVING

STANDARD DEVIATION

Aim :

To calculate the standard deviation of the given values.

Algorithm :

step 1 : Accept the given values

step 2 : calculate

```
mean = sum / float (count);
dev = value [i] - mean;
sumsqrt = dev * dev;
variance = sumsqr / float (count);
stddev = sqrt (variance);
```

step 3 : Display the result

step 4 : End.

OUTPUT

Input values : input -1 to end

34 56.2 12.9 78.0 32 -1

Number of items : 5

Mean = 42.620003

Stddev = 22.389855

Program

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
  const size =100;
int main()
 {
   int i;
   clrscr();
   cout<<"Input values : input -1 to end \n";
   int count =0;
   float value[size];
   float sum =0;
   for(i=1; i<=size; i++)
     {
      cin >> value[i];
      if (value[i] == -1)
      break;
      sum + = value[i];
      count + =1;
     }
   float mean;
   mean = sum / float(count);
   float sumsqr = 0;
   double dev;
   for(i=1; i<=count; i++)
     {
       dev = value[i] - mean;
```

Scanned with CamScanner

```
sum sqr += dev * dev;
double variance, stddev;
variance = sum sqr / float(count);
stddev = sqrt(variance);
cout << " Number of item : cout << " \n";
cout << " Mean = " << mean << '\n';
cout << " stddev = " << stddev << " \n";
getch();
return 0;
}
```

TEMPERATURE CONVERSION

Aim :

To convert the given temperature in farenheit into celsious.

Algorithm :

step 1 : Accept the given temperature

step 2 : calculate c = f - 32 / 1.8

step 3 : Display the result

step 4 : End.

OUTPUT

Enter the value of F 50

celsious = 9

Enter the value of F 35

celsious = 1

PRINT THE GIVEN OUTPUT

Aim :

To print the given output

Algorithm

step 1 : Accept the given value n

step 2 : Display the result

step 3 : End

OUTPUT

Enter the value for n

5

1

2 2

3 3 3

4 4 4 4

5 5 5 5 5

Program

```
#include <iostream.h>
#include <conio.h>
int main()
{
  int n;
  clrscr();
  cout << " Enter the value for n " << "\n";
  cin >> n;
  for (int i=1; i<=n; i++)
    {
     for (int j=1; j<=i; j++)
       {
        cout << i << "\t";
       }
     cout << "\n\n";
    }
 getch();
 return 0;
}
```

LARGEST OF TWO VALUES

Aim :

To find the Largest value of two values

Algorithm :

Step 1 : Accept the values m and n

Step 2 : check whether m >= n

If it is true, return the value of m.

Otherwise return the value of n

Step 3 : Display the result.

Step 4 : End.

OUTPUT

Input values of m and n

675 900

Largest value = 900

Input values of m and n

55 28

Largest value = 55

Program

```
#include <iostream.h>
#include <conio.h>
class set
{
 int m,n;
 public:
       void input(void);
       void display(void);
       int largest(void);
};
int set::largest(void)
  {
    if(m>=n)
    return(m);
    else
     return(n);
  }
void set::input(void)
  {
  cout<<"Input values of m and n"<<"\n";
   cin >>m>>n;
  }
void set::display(void)
  {
   cout<<"largest value = "
        <<largest()<<"\n";
  }
 int main()
 {
   set A;
   A.input();
   A.display();
   return 0;
```

LARGEST OF THREE VALUES

Aim :

To find the largest of three values

Algorithm :

step 1 : Accept the values of a, b and c

step 2 : check whether a >= b

If it is true, check whether a >= c,

If it is true, return the value of a

otherwise return c

step 3 : If a >= b if false, check whether b >= c

If it is true return b or return c

step 4 : Display the result

step 5 : End.

OUTPUT

Input values of a, b and c

34

67

78

Largest value = 78

Program

```
#include <iostream.h>
#include <conio.h>
class set
 {
  int a, b, c ;
 public :
    void input (void);
    void display (void);
    int largest (void);
};
int set :: largest (void)
    {
      if (a >= b)
        {
          if (a >= c)
          return (a);
          else
          return (c);
        }
    else
      {
        if (b >= c)
          return (b);
        else
        return (c);
    }
}
```

```
void set :: input (void)
  {
  cout << " Input values of a, b and c "<< "\n";
  cin >> a >> b >> c;
  }
void set :: display (void)
 {
  cout << " Largest value = "
       << largest () << "\n";
  }
int main ()
  {
  set A;
  A. input ();
  A. display ();
  return 0;
}
```

PROCESSING SHOPPING LIST

Aim :

To process the items of the shopping list

Algorithm :

step 1 : Accept the given appropriate numbers

step 2 : Accept the code and cost of items

step 3 : calculate the total cost

step 4 : delete any item

step 5 : display all items

step 6 : End.

OUTPUT

```
you can do the following : Enter appropriate number
1 : add an item
2 : display total value
3 : delete an item
4 : display all items
5 : quit
what is your option ? 1
enter item code : 111
enter item cost : 200
you can do the following : Enter appropriate number.
1 : add an item
2 : display total value
3 : delete an item
4 : display all items
5 : quit.
what is your option ? 1
enter item code : 222
enter item cost : 400
You can do the following : Enter appropriate number
1: add an item.
2 : display total value
3 : delete an item
4 : display all items
5 : quit
what is your option ? 2
Total value : 600
```

Program

```
#include <iostream.h>
#include <conio.h>
const int m=50;
class ITEMS
{
int itemcode[m];
float itemprice[m];
int count;
public:
void CNT(void) {count=0;}
void getitem(void);
void display sum(void);
void remove(void);
void display items(void);
};
void ITEMS :: get item(void)
{
cout << " Enter item code: ";
cin >> item code[count];
cout << " Enter item cost: ";
cin >> item price[count];
count++;
}
void ITEMS :: display sum(void)
{
float sum=0;
```

You can do the following : Enter appropriate number

1 : add an item

2 : display total value

3 : delete an item

4 : display all items

5 : quit

what is your option ? 3

enter item code : 111

You can do the following : Enter appropriate number

1 : add an item

2 : display total value

3 : display an item

4 : display all items

5 : quit.

what is your option ? 4

code price

111 0

222 400

you can do the following : Enter appropriate number

1 : add an item

2 : display total value

2 : display an item

4 : display all items

5 : quit

What is your option ? 5

```
for(int i=0; i<count; i++)
sum=sum+item price[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(item code[i]==a)
   item price[i]=0;
 }
void ITEMS::display ITEMS(void)
  {
   cout<<"\n code price \n";
   for(int i=0; i<count; 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 >> n;
switch (n)
 {
  case 1 : order. get item ();  break;
  case 2 : order. display sum (); break;
  case 3 : order. remove (); break;
  case 4 : order. display
  getch ();
  return 0;
}
```

BANK ACCOUNT

Aim :

To find the detail about the bank account.

Algorithm :

Step : 1 Accept the given name account number and amount.

Step 2 : check whether there is any customer with that account no and withdrawn amount is grater than the balance amount.

Step 3 : diplay all details

Step 4 : End.

OUTPUT

```
Enter your choice
1 - Add an account
2 - withdraw an amount from account
3 - display all accounts
4 - display a particular account
5 - quit
Enter Name : Basira
Enter Account no : 514
Enter amount : 10000
Enter your choice
1 - Add an account
2 - withdraw an amount from account
3 - display all accounts
4 - display a particular account
5 - quit
Enter Name : Amra
Enter Account No : 321
Enter amount : 5000
Enter your choice
1 - Add an account
2 - withdraw an account from account
3 - display all accounts
4 - display a particular account
5 - Quit
1
Enter Name : Teya
Enter Account no : 472
Enter amount : 12000
```

Program

```
#include <iostream.h>
#include <conio.h>
class account
 {
 char name [50] [30];
 int ano [50], count;
 float balance [50];
 public:
   void init()
  {
   count = 0;
  }
Void deposit();
Void withdrawn();
Void display();
Void display all();
};
Void account :: deposit()
 {
 Cout << " Enter name : ";
 cin >> name [count];
 cout << " Enter account no : ";
 cin >> ano [count];
 cout << " Enter amount : ";
 cin >> balance [count];
 count ++;
}
Void account :: withdrawn()
 {
```

Enter your choice

1 - Add an account

2 - withdraw an amount from account

3 - display all accounts

4 - display a particular account

5 - Quit

1

Enter Name : Deepa

Enter Account no : 912

Enter amount : 7000

Enter your choice

1 - Add an account

2 - withdraw an amount from account

3 - display all accounts

4 - display a particular account

5 - Quit

2

Enter account no & the amount to be withdrawn

333 6000

Enter your choice

1 - Add an account

2 - withdraw an amount from account

3 - display all accounts

4 - display a particular account

5 - Quit

3

```
int a, j = 0, j ;
float b, amt ;
cout << " Enter account no & the amount
                to be withdrawn " ;
cin >> a ;
cin >> amt ;
for ( i=0 ; i < count ; i++ )
{
  b = balance [i];
  j = 1 ;
  break ;
  }
}
if ( j == 0 )
cout << " There is no customer with the
          given account no " ;
else
  {
   if (amt >= b)
   cout << " withdrawn amount is greater than
          balance amount so we can't withdrawit;
  else
  balance [i] = b - amt ;
  }
}
void account :: display all ( )
 {
  cout << " Name account no Amount " << "\n";
  for (int i=0 ; i < count ; i++)
    {
     cout << name [i];
```

Name	Account no	Amount
Basira	514	10000
Amra	321	5000
Teja	472	12000
Deepa	910	7000
Lakshmi	573	19000

Enter your choice

1 - Add an account

2 - withdraw an amount from account

3 - display all accounts

4 - display a particular account

5 - Quit

Enter account no to be displayed

514

Name	Account no	Amount
Basira	514	10000

Enter your choice

1 - Add an account

2 - withdraw an account from account

3 - display all accounts

4 - display a particular account

5 - quit

5

```
cout << "          " << ano[i];
cout << "          " << balance[i] << "\n";
   }
}
void account :: display ()
   {
    int a, j = 0;
    cout << "Enter account no to be displayed";
    cin >> a;
    for (i=0; i< count; i++)
     { if (ano[i]==a)
       j = 1;
       break;
     }
}
if (j==1)
 {
  cout << "Name Account no amount " << "\n";
  cout << name[i];
  cout << "          " << ano[i];
  cout << "          " << balance[i];
 }
else
  {
  cout << "There is no such a record ";
  }
}
int main ()
  {
   account acc;
```

```
acc.init();
int n;
clrscr();
do
{
cout<<" Enter your choice \n";
cout<<" \n1 - Add an account ";
cout<<" \n2 - withdraw an amount from
                                account ";
cout<<" \n3 - display all account ";
cout<<" \n4 - display a particular account;
cout<<" \n5' - quit ";
cin>>n;
switch(n)
{
case 1 : acc.deposit();
         break;
case 2 : acc.withdrawn();
         break;
case 3 : acc.display all();
         break:
case 4 : acc.display();
         break:
case 5 : break;
default : cout<<" Try again \n";
  }
}
while(n!=5)
  getch();
  return 0;
}
```

CHANGE IN SIGN

Aim :

To overload the unary minus operator

Algorithm :

Step 1 : Accept the values

Step 2 : To overload the unary minus operator on this value.

Step 3 : display the result

Step 4 : End.

OUTPUT

s : 10 -20 30

s : -10 20 -30

Program

```
#include <iostream.h>
#include <conio.h>
class space
 {
  int x;
  int y;
  int z;
public:
  void getdata (int a, int b, int c);
  void display (void);
  void operator - ();
};
void space :: getdata( int a, int b, int c)
 {
   int x = a;
       y = b;
       z = c;
  }
void space :: display (void)
 {cout << x << "  ";
  cout << y << "  ";
  cout << z << "\n";
}
void space :: operator - ()
  {
    x = -x;
    y = -y;
    z = -z;
  }
int main ()
 {
  space s;
s.getdata (10, 20, 30);
cout << "S : ";
 s.display();
 -s;
cout << "S : ";
s.display ();
return 0;
}
```

ADDING TWO COMPLEX VALUE

Aim :

To add two complex numbers using for

Algorithm :

step 1 : Accept the values of real and imaginary parts of two complex numbers.

step 2 : Add two complex numbers

step 3 : Display the result.

step 4 : End.

OUTPUT

c1 = 2.5 + 3.5 i

c2 = 1.6 + 2.7 i

c3 = 4.1 + 6.2 i

Program

```
#include<iostream.h>
#include<conio.h>
class complex
{
 float x;
 float y;
public:
  complex(){}
  complex(float real, float imag)
  {x = real; y=imag;}
  complex operator + (complex);
  void display(void)
};
complex complex :: operator + (complex c)
{
 complex temp;
 temp.x = x+c.x;
 temp.y = y+c.y;
 return (temp);
}
void complex :: display(void)
{
 cout << x << "+" << y << "i" << "\n";
}
int main()
{
 complex c1, c2, c3;
 clrscr();
 c1 = complex(2.5, 3.5);
 c2 = complex(1.6, 2.7);
 c3 = c1 + c2;
cout << "c1 = "; c1.display();
cout << "c2 = "; c2.display();
cout << "c3 = "; c3.display();
getch();
return 0;
}
```

OVERLOADING OPERATORS USING FRIEND

Aim :

write a program to overload operators using friends.

Algorithm :

step 1 : Accept a scalar 'a' and a vector x, y, z

step 2 : Multiply this vector with a scalar to overload * operator using friends

step 3 : display the result

step 4 : End.

OUTPUT

Enter elements of vector m

5 16 22

m = (5, 16, 22)

P = (10, 32, 44)

9 = (4, 8, 12)

Program

```
#include <iostream.h>
#include <conio.h>
const int 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 << (istream &, 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;
  clrscr ();
  cout << " Enter elements of vector m " << '\n';
  cin >>m ;
  cout <<"\n " ;
  cout << " m = " << m << "\n ";
  vector p,q ;
   p = 2 * m ;
   q = m * 2 ;
cout << "\n";
cout << " p =  " << p << "\n " ;
cout << " q =  " << q << "\n";
getch ();
return 0 ;
}
```

ADD TO COMPLEX NUMBER
(USING FRIEND FUNCTION)

Aim :

To add two complex numbers using friend function.

Algorithm :

Step 1 : Accept the values of real and imag parts of two complex numbers.

Step 2 : Add two complex numbers.

Step 3 : Display the result.

Step 4 : End.

OUTPUT

C1 = 2.5 + 3.5i

C2 = 1.6 + 2.7i

C3 = 4.1 + 6.2i

program

```
#include <iostream.h>
#include <conio.h>
class complex
 {
   float x,y;
public:
complex() {}
complex (float real, float imag)
 { x=real; y=imag; }
friend complex operator +(complex, complex)
 void display(void);
};
complex operator +(complex a, complex b)
  {
   complex temp;
   temp.x = a.x+b.x;
   temp.y = a.y+b.y;
   return(temp);
  }
void complex :: display(void)
{
 cout << x << "+i" << y << "\n";
}
int main()
{
 complex c1, c2, c3;
 c1 = complex(2.5, 3.5);
 c2 = complex(1.6, 2.7);
 c3 = operator +(c1, c2);
 cout << "c1 = "; c1.display();
 cout << "c2 = "; c2.display();
 cout << "c3 = "; c3.display();
 getch();
 return 0;
}
```

2 MULTIPLICATION OF TWO COMPLEX NUMBERS.

Aim :

To multiply two complex numbers using operator overloading

Algorithm :

Step 1 : Accept the values of real and imaginary parts of two complex numbers.

Step 2 : Multiply the two complex numbers.

Step 3 : Display the result.

Step 4 : End.

OUTPUT

$c1 = 2.5 + 3.5i$

$c2 = 1.6 + 2.7i$

$c3 = -5.45 + 10.35i$

Program

```
#include <iostream.h>
#include <conio.h>
class complex
{
 float x,y;
public:
 complex() {}
 complex(float real, float imag)
 { x = real; y = imag; }
 complex operator * (complex);
 void display(void);
};
complex complex :: operator * (complex c)
 {
 complex temp;
 temp.x = x * c.x - y * c.y;
 temp.y = x * c.y + y * c.x;
 return(temp);
}
void complex :: display(void)
 {
cout << x << " +" << y << "i" << "\n";
 }
int main()
 {
 complex c1, c2, c3;
 clrscr();
 c1 = complex(2.5, 3.5);
 c2 = complex(1.6, 2.7);
 c3 = c1 * c2;
cout << " c1 = "; c1.display();
cout << " c2 = "; c2.display();
cout << " c3 = "; c3.display();
 getch();
 return 0;
}
```

MULTIPLICATION OF TWO COMPLEX NUMBERS (using Friend Function)

Aim :

To multiply two complex numbers using friend function.

Algorithm :

Step 1 : Accept the values of real & imaginary parts of two complex numbers.

Step 2 : Multiply two complex numbers.

Step 3 : Display the result.

Step 4 : End.

OUTPUT

C1 = 2.5 + 3.5i

C2 = 1.6 + 2.7i

C3 = −5.45 + 10.35i

Program

```
#include <iostream.h>
#include <conio.h>
class complex
 {
  float x,y;
public:
   complex(){}
   complex(float real, float imag)
 {x=real; y=imag;}
friend complex operator*(complex, complex)
 void display(void);
};
complex operator*(complex a, complex b)
  {
   complex temp;
   temp.x = a.x*b.x - a.y*b.y;
   temp.y = a.x*b.y + a.y*b.x;
   return(temp);
  }
void complex::display(void)
 {
 cout<<(x<<"+"<<y<<"i"; "<<"\n";
 }
 int main()
  {
   complex c1, c2, c3
    c1 = complex(2.5, 3.5);
    c2 = complex(1.6, 2.7);
    c3 = operator*(c1, c2);
   cout<<"c1 = "; c1.display();
   cout<<"c2 = "; c2.display();
   cout<<"c3 = "; c3.display();
   getch();
  return 0;
  }
```

LIBRARY DETAILS USING FUNCTIONS

Aim:

To maintain and display library details using friends.

Algorithm:

Step 1: Accept the details of book stored in the library

Step 2: Maintain the details

Step 3: Display the result

Step 4: End.

OUTPUT

1. Add a book to library
2. display a particular book details
3. display all book details
4. check whether a particular book is available or not
5. Exit

Enter your choice

1

Enter Book no : 111

Enter book name : algebra

Enter author name : arumugam

Enter price of the book : 150

LIBRARY MANAGEMENT

1. Add a book to library
2. Display a particular book details
3. display all book details
4. check whether a particular book is available or not
5. Exit

Enter your choice

1

Enter book no : 222

Enter book name : complex

Enter author name : Issac

Enter price of the book : 200

LIBRARY MANAGEMENT

1. Add a book to library
2. Display a particular book details
3. display all book details
4. check whether a particular book is available or not
5. Exit

Program

```
# include <iostream.h>
# include <conio.h>
# include <string.h>
class lib
{
  int bno[20];
  char bname[20][30];
  char aname[20][30];
  float price[20];
  int count;
  int i;
public:
void init() { count = 0; }
 void add();
 void display all();
 void find();
 void display();
};
void lib :: add()
 {
 cout << "\n Enter book no : ";
  cin >> bno[count];
 cout << "\n Enter book name : ";
  cin >> bname[count];
  cout << "\n Enter author name : ";
  cin >> aname[count];
  cout << "\n Enter price of the book : ";
   cin >> price[count];
   count++;
 }
```

Enter your choice

2

Enter the book no to be displayed

222

Bookno	Bookname	Authorname	price
222	complex	Isaac	200

LIBRARY MANAGEMENT

1. Add a book to library
2. display a particular book details
3. display all book details
4. check whether a particular book is available
5. Exit

Enter your choice

3

Bookno	Bookname	Authorname	price
111	algebra	arungam	150
222	complex	Isaac	200

LIBRARY MANAGEMENT

1. Add a book to library
2. display a particular book details
3. display all book details
4. check whether a particular book is available
5. Exit.

Enter your choice

4

Enter the book name to be displayed found

algebra

The given book is available in our library

```
void lib :: display ()
{
 int a, flag ;
 flag = 0 ;
 cout << "Enter the book no to be displayed";
 cin >> a ;
 for (i=0 ; i < count ; i++)
  {
   if ( a = = bno[i] )
    {
     flag = 1 ;
     break ;
    }
  if (flag == 1)
  {
 cout << "\n Book no     Book name     Authorname
                                        price  \n";
  cout << bno[i] ;
  cout << "    " << bname [i];
  cout << "    " << aname [i];
  cout << "    " << price [i];
 }
else
 {
 cout << "\n There is no book available for
            that book no" ;

  }
 }
void lib :: find ()
```

LIBRARY MANAGEMENT

1. Add a book to library
2. Display a particular book details
3. Display all book details
4. Check whether a particular book is available or not
5. Exit

Enter your choice

4

Enter the book name to be found

Tamil

The given book is not available in our library

LIBRARY MANAGEMENT

1. Add a book to library
2. Display a particular book details
3. Display all book details
4. Check whether a particular book is available or not
5. Exit

Enter your choice

6

Error Try again

LIBRARY MANAGEMENT

1. Add a book to library
2. Display a particular book details
3. Display all book details
4. Check whether a particular book is available or not
5. Exit

Enter your choice

5

```
{
char b[30];
int flag;
flag=0;
cout<<"\n Enter the bookname to be found";
cin>>b;
for(i=0;i<count;i++)
{
if(strcmp(b,name[i])==0)
{
flag=1;
break;
}
else
{
flag=0;
}
}
if(flag==1)
{
cout<<"\n The given book is available in our
library";
}
else
{
cout<<"\n The book is not available in
our library";
}
}
void lib::displayall()
{
cout<<"\n Bookno Bookname Authorname price\n";
```

```
for (i=0 ; i<count ; i++)
 {
cout << bno[i] :
cout << "    " << bname[i] ;
cout << "    " << aname[i] ;
cout << "    " << price[i];
cout << "\n";
  }
}
int
void main()
 {
clrscr();
lib l;
l.init();
int c;
do
{
cout << "\n LIBRARY MANAGEMENT \n";
cout << "\n1. Add a book in library ";
cout << "\n2. display a particular book details";
cout << "\n3. display all book details" ;
cout << "\n4. check whether a particular book
                is available or not";
cout << "\n5. Exit :
cout << "\n\n Enter your choice :
cin >> c;
switch(c)
 {
  case 1 :
      l. add(); break;
```

```
case 2 :
    l. display ( ) ; break ;

case 3 :
    l. displayall ( ) ; break ;

case 4 :
    l. fuid ( ) ; break ;

case 5 :
    break ;

default :
  cout << " In Error. try again " ;
    }
} while (c1 = 5);
  getch ( ) ;
  return 0 ;
}
```

DATA CONVERSION

Aim :

To maintain stock details using class.

Algorithm :

Step 1 : Accept the item name, code, value and price

Step 2 : Maintain the details using class

Step 3 : Display the result

Step 4 : End.

OUTPUT

product details - invent 1 type

code : 100

Items : 5

value : 140

Stock value

value : 700

product details - invent 2 type

code : 100

value : 700

program

```
#include <iostream.h>
#include <conio.h>
class invent2;
class invent1
{
 int code;
 int items;
 float price;
public:
 invent1(int a, int b, int c)
  {
   code = a;
   items = b;
   price = c;
 }
void putdata()
 {
 cout << " code : " << code << "\n";
 cout << " Items : " << items << "\n";
 cout << " value : " << price << "\n";
 }
int getcode() { return code; }
int getitems() { return items; }
int getprice() { return price; }
operator float() { return (items * price); }
};
class invent2
 {
 int code;
 float value;
```

```
public :
invent2()
{
code = 0 ; value = 0 ;
}
invent2 (int x, float y)
{
code = x ;
value = y ;
}
void putdata()
{
cout << " code : " << code << "\n" ;
cout << " value : " << value << "\n\n" ;
}
invent2 (invent1 P)
{
code = P.getcode() ;
value = p.getitems() * P.getprice() ;
}
};
int main()
{
invent1 s1(100, 5, 140.0) ;
invent2 , d1 ;
float total_value ;
total_value = s1 ;
d1 = s1 ;
cout << " product details invent1 type " << "\n" ;
s1.putdata() ;
```

```
cout << " In stock value " << "\n";
cout << " value = " << total - value << "\n\n";
cout << " product details - current type " << "\n";
d1.putdata();
getch();
return 0;
}
```

EMPLOYEE INFORMATION USING INHERITANCE

Aim :

To Maintain employees information using inheritance.

Algorithm :

step : 1 : Accept employee's name, basic pay, deduction.

step : 2 calculate DA, HRA, TA, LIC, PF

step 3 : Display the details

step 4 : End

OUTPUT

Enter empno and name : 10

Raja

Enter basic pay, dearness allowance, travelling allow
house rent allowance.

10000

200

500

300

Enter provident fund and LIC details

1000

500

number : 10

name : Raja

Basic pay : 10000

Dearness Allowance : 200

Travelling allowance : 500

House rent Allowance : 300

Deduction details :

provident fund : 1000

LIC : 500

gross pay : 11000

net pay : 9500

Program

```
#include <iostream.h>
#include <conio.h>
class temp emp
 {
public :
  int empno ;
  char empno ;
   void get() ;
    void put() ;
} ;
Void emp :: get()
 {
cout << "\n Enter empno. and name: " ;
 cin >> empno ;
  cin >> name ;
 }
Void emp :: put
 {
  cout << " Number :  " << empno ;
  cout << " Name  :  " << name ;
 }
class salary : public emp
 {
public :
    float BP ;
    float DA , TA , HRA ;
```

```
float PF, LIC;
 void getlic();
 void putlic();
};
void salary :: getlic()
{
cout << " Enter basic pay, Dearness allowance,
      Travelling allowance, House rent allowance;
cin >> BP >> DA >> TA >> HRA;
cout << " Enter provident fund and
                    LIC details ";
cin >> PF >> LIC;
}
void salary :: putlic()
{
cout << "\n Basic pay : " << BP;
cout << "\n Dearness Allowance : " << DA;
cout << "\n Travelling Allowance : " << TA;
cout << "\n House rent Allowance : " << HRA;
cout << "\n Deduction details : ";
cout << "\n provident Fund : " << PF
cout << "\n LIC : " << LIC;
}
class Netsalary : public salary
{
public :
  float GP;
  float NP;
```

```
void netsal()
  {
  GF = BP + DA + TA + HRA ;
  NF = GF - PF - LIC ;
  cout << "\n Gross pay : " << GF ;
  cout << "\n Net pay : " << NF ;
    }
};
int
void main()
  {
  netsalary ns ;
  ns.get();
  ns.get1();
  ns.put();
  ns.put1();
  ns.netsal();
  getch();
}
```

STUDENT FILE

Aim:

write a program to create student file.

Algorithm:

Step 1: Accept the information of students

Step 2: create a student file

Step 3: display the results

Step 4: End

OUTPUT

Enter details for three students

Enter name : Basira

Enter rollno : 111

Enter m1, m2 : 80 70

Enter name : Amra

Enter rollno : 450

Enter m1, m2 : 92 96

Enter name : Jeya

Enter rollno : 222

Enter m1, m2 : 96 98

Basira	111	80	70	150
Amra	450	92	96	188
Jeya	222	96	98	194

Program :

```
#include <iostream.h>
#include <fstream.h>
#include <iomanip.h>
#include <conio.h>
class studentfile
{
char name[20];
int rollno;
int m1, m2;
int total;
public:
  void readdata(void);
  void writedata(void);
};
void studentfile :: readdata(void)
{
cout << "\n Enter name : "; cin >>name;
cout << " \n Enter rollno : "; cin >> rollno;
cout << " \n Enter m1, m2 : "; cin >>m1>>m2;
}
void studentfile :: writedata(void)
{
 total = m1+m2;
 cout << setiosflags(ios::left)
   << setw(16) << name << setiosflags(ios::right)
   << setw(10) << rollno << setprecision(2) << setw(8)
   << m1 << "\t"
   << setiosflags(ios::right) << setw(10) <<
                                  total << endl;
}
```

```
int main()
{
int total;
clrscr();
student fil marks[3];
fstream fil;
fil.open("stu.dat", ios::in|ios::out);
cout<<" Enter details for three students\n";
for(int i=0; i<3; i++)
{
 marks[i].readdata();
 fil.write((char *) &marks[i],
                    sizeof(marks[i]));
}
fil.seekg(0);
cout<<"\n output \n\n";
for(i=0; i<3; i++)
 {
fil.read((char *) &marks[i], sizeof
                    (marks[i]));
   marks[i].writedata();
 }
fil.close();
getch();
}
```

EMPLOYEE INFORMATION USING FILES

Aim:

To maintain employee's information using files

Algorithm:

Step 1: Accept employees details

Step 2: Maintain this using files

Step 3: Display the result

Step 4: End

OUTPUT

```
enter details for three employees
enter no of items
3
enter name : radha
enter empno : 1
enter bp : 1000
enter ded : 200
enter name : rahul
enter empno : 2
enter bp : 1000
enter ded : 300
enter name : roshan
enter empno : 3
enter bp : 1000
enter ded : 500
```

radha	1	1000	200	100	900
rahul	2	1000	300	100	800
roshan	3	1000	500	100	600

Algorithm

```
#include <iostream.h>
#include <fstream.h>
#include <iomanip.h>
#include <conio.h>
class employee
{
 char name[10];
 int empno;
float bp, ded, allow;
 float gp;
public:
   void readdata(void);
   void writedata(void);
};
void employee :: readdata(void)
 {
 cout << " enter name : ";
 cin >> name;
 cout << " enter empno : ";
 cin >> empno;
 cout << " enter bp : ";
 cin >> bp;
 cout << " enter ded : ";
 }
void employee :: writedata(void)
 {
   allow = 0.10 * bp;
   gp = allow + bp - ded;
cout << setiosflags(ios::right) << setw(10) << name
   << setiosflags(ios::right) << setw(10) << empno
```

```
<< setiosflags(ios::right) << setprecision(2)
                << setw(10) << bP
<< setiosflags(ios:right) << setprecision(2) <<
                setw(10) << dled
<< setiosflags(ios::right) << setprecision(2)
                << setw(10) << allow
<< setiosflags(ios::right) << setprecision(2)
                << setw(10) << gP
   << endl;
}
void main()
{
 int size;
 employee item[10];
 clrscr();
 fstream file;
file.open("emp.dat", ios::in|ios::out);
cout<<"enter details for three employees\n";
cout<<"enter no of items"<<"\n";
cin>>size;
for(int i=0; i<size; i++)
{
 item[i].readdata();
 file.write((char*) &item[i], sizeof(item[i]));
}
file.seekg(0);
cout<<"\n output\n\n";
 for(i=0; i<size; i++)
  {
file.read((char*) &item[i], sizeof
                (item[i]));
```

```
item fi.writedata();
}
fi.close();
getch();
}
```