In object oriented programming language the program will be divided into modules called as objects these objects will interact with each other to achieve a particular task.

Ex: = c++, java, vb.net, asp.net, c#

# ADVANTAGES

* Rather writing functions objects will be created security of data will be maintained
* No need of following any procedure instead objects will procedure instead objects will interact with each other to achieve a task.

# Components of object oriented programming language.

* **Objects:=** objects are the run-time entities of program.
* **Classes:=**  Group of inter-related objects to achieve a task.
* **Polymorphism:=**  process of giving an additional meaning to an operator or a function.
* **Inheritance:=**  acquiring the properties of one class to another class.

# Encapsulation & data abstraction

Wrapping up of a data members and member functions in a single unit [class] called encapsulation.

Accessing the properties of class without using the back ground details.

# Introduction to c++

C++ was developed in AT&T laboratories at U.S.A & developed by B Jarne Strostrup.

C++ includes all the features of c with additional features.

* **Character set of c++** := A-Z, a-z, alphabets .0-9 numbers.
* **Special characters**:= . , : ; ‘ ‘ “ “ \_ () {} [] - :: << >> || ! !!.

# C++ Tokens

* **Keywords**: = Reserved words used by c++ ex:= cout, cin, class.
* **Identifies**: = are used to Identify functions variables, const class, structure.
* **Variables**: = A storage unit, the value of which will change during the execution of program.
* **Constants**: = A storage unit, the value of will not change during the execution of program.
* **Operators**: = An operator is a symbol to identify the type of operation.
* **Arithmetic Operators**: = +, - , \* , / , %.
* **Relational Operators**: = >, <, >=, <= , ==, !=.
* **Logical Operators**: = And &, Ampersand &&, Or || not ! .
* **Conditional Operators**: = to check the condition 🡪 ? : , <exp1>?, <exp2>:<exp3>, max=(a>b)?a:b.
* **Assignment Operators**:= to assign the values from:= Rhs to Lhs ,op:= , sum=a+b.
* **Increment/Decrement Operators**: = ++, --.

# Data Types

* **Data types**:= type of a data given to computer. There are two types of data types.
* **Primary data type**: = int, float, char, string.
* **Derived data types**: = Data type derived from primary data type, arrays, functions, pointers.
* **User defined data types**: =data type , derived from user, structures, classes, files, Quires.
* **Expressions**: = combination of constants variables & operators.

# Header \* Files

* **Iostream.h** 🡪 Input/output stream header files or input/output statements.
* **Conio.h** 🡪 consol input/output header files or screen output header files.
* **Input** 🡪 cin with insertion operator[>>]. Syntax:= cin<<\_ \_ \_ \_ << \_ \_ \_ \_ ;
* **Output**🡪 cout with extraction operator[<<]. Syntax:= cout<<\_ \_ \_ \_ << \_ \_ \_ \_ ;
* **Creating I/O Statements** :=Cin>>\_>>\_; Cin>>a>>b; cout<<\_ \_ << \_ \_;
* **Private**:= variables or a functions declared or defined under private section can be accessed within the class only.
* **Public**:= can be accessed within the class as well as outside the class.

# Assignment Programs

1. //program to display a simple message

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

cout<<"\n Welcome to C++";

getch();

}

1. //program to display a simple message

#include<iostream.h>

#include<conio.h>

void main()

{

clrscr();

cout<<"All"<<endl<<"The"<<endl<<"Best";

getch();

}

1. //program to find sum of two num’s

#include<iostream.h>

#include<conio.h>

void main()

{

int a,b,sum;

clrscr();

cout<<"Enter two nums";

cin>>a>>b;

sum=a+b;

cout<<"\n sum="<<sum;

getch();

}

1. //program to perform basic arithmetic operator

#include<iostream.h>

#include<conio.h>

void main()

{

float a,b;

clrscr();

cout<<"Enter two num’s";

cin>>a>>b;

cout<<"\n sum="<<(a+b);

cout<<"\n subtraction="<<(a-b);

cout<<"\n product="<<(a\*b);

cout<<"\n division="<<(a/b);

getch();

}

1. //program to find area of a circle

#include<iostream.h>

#include<conio.h>

#define pi 3.142

void main()

{

float ar,r;

clrscr();

cout<<"Enter radius";

cin>>r;

ar=pi\*r\*r;

cout<<"\n Area of a circle="<<ar;

getch();

}

1. //program to find area of a rectangle

#include<iostream.h>

#include<conio.h>

void main()

{

float ar,l,b;

clrscr();

cout<<"Enter length and breadth";

cin>>l>>b;

ar=l\*b;

cout<<"\n Area of a rectangle="<<ar;

getch();

}

1. //program to find area of a traingle

#include<iostream.h>

#include<conio.h>

void main()

{

float ar,h,b;

clrscr();

cout<<"Enter base and height";

cin>>b>>h;

ar=0.5\*b\*h;

cout<<"\n Area of a triangle="<<ar;

getch();

}

1. //program to find area of a traingle when 3 sides are given

#include<iostream.h>

#include<conio.h>

#include<math.h>

void main()

{

float ar,a,b,c,s;

clrscr();

cout<<"Enter 3 sides";

cin>>a>>b>>c;

s=(a+b+c)/2;

ar=sqrt(s\*(s-a)\*(s-b)\*(s-c));

cout<<"\n Area of a triangle="<<ar;

getch();

}

1. //program to find simple interest

#include<iostream.h>

#include<conio.h>

void main()

{

float p,t,r,si;

clrscr();

cout<<"Enter p t r values";

cin>>p>>t>>r;

si=(p\*t\*r)/100;

cout<<"\n Simple Interest="<<si;

getch();

}

1. //program to find the sum and charage of three numbers

#include <iostream.h>

#include <conio.h>

void main()

{

int a,b,c;

float sum,avg;

clrscr();

cout<<"enter three numbers:";

cin>>a>>b>>c;

sum=a+b+c;

avg=sum/3;

cout<<"\n sum="<<sum;

cout<<"\n Average"<<avg;

getch();

}

1. //program to find biggest of two numbers using conitional operator

#include <iostream.h>

#include <conio.h>

void main()

{

int a,b;

clrscr();

cout<<"enter two numbers:";

cin>>a>>b;

(a>b)?cout<<"\n a is big ":cout<<"\n b is big";

getch();

}

1. /\*program to check whether given no is even or odd using

conitional operator\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int n;

clrscr();

cout<<"enter any numbers:";

cin>>n;

(n%2==0)?cout<<"\n Even no ":cout<<"\n odd no";

getch();

}

1. /\*program to check whether given year is a leap year or not using

conitional operator\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int year;

clrscr();

cout<<"enter any numbers:";

cin>>year;

(year%4==0)?cout<<"\n Leap Year ":cout<<"\n Not a leap year";

getch();

}

1. /\*program to demo sizeof operator\*/

#include <iostream.h>

#include <conio.h>

void main()

{

clrscr();

cout<<"size of ineteger"<<sizeof(int);

cout<<"size of float"<<sizeof(float);

cout<<"size of character"<<sizeof(char);

cout<<"size of double"<<sizeof(double);

cout<<"size of signed"<<sizeof(signed);

cout<<"size of unsigned"<<sizeof(unsigned);

cout<<"size of short"<<sizeof(short);

cout<<"size of long"<<sizeof(long);

getch();

}

1. /\*program to demo increment operator\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int i=10,j=20;

clrscr();

cout<<"\n value of i"<<i;

cout<<"\n after incrementing i value"<<i++;

cout<<"\n value of j"<<i<<endl;

cout<<"\n after incrementing j value"<<++i;

getch();

}

1. /\*program to demo decrement operator\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int i=10,j=20;

clrscr();

cout<<"\n value of i"<<i;

cout<<"\n after decrementing i value"<<i--;

cout<<"\n value of j"<<j<<endl;

cout<<"\n after decrementing j value"<<--j;

getch();

}

1. /\*program to find biggest using simple if\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int i=10,j=20;

clrscr();

if(i>j)

cout<<"\n i is big";

if(i<j)

cout<<"\n j is big";

getch();

}

1. /\*program to find check given no is +ve or -ve using simple if\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int n;

clrscr();

cout<<"\n Enter any no";

cin>>n;

if(n>0)

cout<<"\n +ve no";

if(n<0)

cout<<"\n -ve no";

getch();

}

1. /\*program to find check given no is +ve or -ve using if else\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int n;

clrscr();

cout<<"\n Enter any no";

cin>>n;

if(n>0)

cout<<"\n +ve no";

else

cout<<"\n -ve no";

getch();

}

1. /\*program to check result of a student using if else if\*/

#include <iostream.h>

#include <conio.h>

void main()

{

int per;

clrscr();

cout<<"\n Enter percentage";

cin>>per;

if(per>=80)

cout<<"\n Distinction";

else if(per>=60)

cout<<"\n First Class";

else if(per>=50)

cout<<"\n Second Class";

else if(per>=35)

cout<<"\n Just pass";

else

cout<<"\n Fail";

getch();

}

1. /\*program to read a character and check whether given character is a

alphabet, digit or a special character using if-else-if\*/

#include <iostream.h>

#include <conio.h>

void main()

{

char ch;

clrscr();

cout<<"\n Enter any character";

cin>>ch;

if(ch>='A' && ch<='Z'||ch>='a'&&ch<='z')

cout<<ch<<" is a alphabet";

else if(ch>='0' && ch<='9')

cout<<ch<<" is a digit";

else

cout<<ch<<" is a special character";

getch();

}

1. //program to read a character and check whether given character is a

alphabet, digit or a special character using ascii values.

#include <iostream.h>

#include <conio.h>

void main()

{

char ch;

clrscr();

cout<<"\n Enter any character";

cin>>ch;

if(ch>=65 && ch<=90 ||ch>=97 &&ch<=122)

cout<<ch<<" is a alphabet";

else if(ch>=48 && ch<=57)

cout<<ch<<" is a digit";

else

cout<<ch<<" is a special character";

getch();

}

1. /\*program to read a character and check whether given character is a

alphabet, digit or a special character using built in functions\*/

#include <iostream.h>

#include <conio.h>

#include<ctype.h>

void main()

{

char ch;

clrscr();

cout<<"\n Enter any character";

cin>>ch;

if(isalpha(ch))

cout<<ch<<" is a alphabet";

else if(isdigir(ch))

cout<<ch<<" is a digit";

else

cout<<ch<<" is a special character";

getch();

}

1. //program to print nums 1 to 10

#include <iostream.h>

#include <conio.h>

void main()

{

int i=1;

clrscr();

while(i<=10)

{

cout<<i<<endl;

i++;

}

getch();

}

1. //program to print nums 10 to 1

#include <iostream.h>

#include <conio.h>

void main()

{

int i=10;

clrscr();

while(i>=1)

{

cout<<i<<endl;

i--;

}

getch();

}

1. //program to print nums 1 to 10 even nos

#include <iostream.h>

#include <conio.h>

void main()

{

int i=2;

clrscr();

while(i<=10)

{

cout<<i<<endl;

i=i+2;

}

getch();

}

1. //program to print nums 1 to 100 nos which are div by 5

#include <iostream.h>

#include <conio.h>

void main()

{

int i=5;

clrscr();

while(i<=100)

{

cout<<i<<endl;

i=i+5;

}

getch();

}

1. //program to print nums 1 to 100 nos which are div by 3 and 5

#include <iostream.h>

#include <conio.h>

void main()

{

int i=1;

clrscr();

while(i<=100)

{

if(i%3==0 && i%5==0)

cout<<i<<endl;

i++;

}

getch();

}

1. //program to read a no and find factorial a given no

#include <iostream.h>

#include <conio.h>

void main()

{

int i=1,fact=1,n;

clrscr();

cout<<"\n enter any no";

cin>>n;

while(i<=n)

{

fact=fact\*i;

i++;

}

cout<<"Factorial of a given no="<<fact<<endl;

getch();

}

1. //program to read a no and find sum of N nos

#include <iostream.h>

#include <conio.h>

void main()

{

int i=1,sum=0,n;

clrscr();

cout<<"\n enter any no";

cin>>n;

while(i<=n)

{

sum=sum+i;

i++;

}

cout<<"sum of N nos="<<sum<<endl;

getch();

}

1. //program to read a no and find sum of odd and even nos

#include <iostream.h>

#include <conio.h>

void main()

{

int i=1,sume=0,sumo=0,n;

clrscr();

cout<<"\n enter any no";

cin>>n;

while(i<=n)

{

if(i%2==0)

{

sume=sume+i;

}

else

{

sumo=sumo+i;

}

i++;

}

cout<<"sum of even nos="<<sume<<endl;

cout<<"sum of odd nos="<<sumo<<endl;

getch();

}

1. //program to find sum and average of two no’s.

#include <iostream.h>

#include <conio.h>

class myclass

{

private:

int a,b;

float sum,avg;

public:

void getdata()

{

cout<<"enter the value of a and b:";

cin>>a>>b;

}

void calc()

{

sum=a+b;

avg=sum/2;

}

void show()

{

cout<<"\nsum of a two no’s:="<<sum;

cout<<"\n average fof a two no’s:="<<avg;

}

};

void main()

{

Myclass m;

clrscr();

m.getdata();

m.calc();

m.show();

getch();

}

1. //program to find simple interest.

#include <iostream.h>

#include <conio.h>

class simple

{

private:

int p,t;

float si,r;

public:

void getdata()

{

cout<<"enter the value of principle amount:";

cin>>p;

cout<<"enter the time:";

cin>>t;

cout<<"enter the value of rate:";

cin>>r;

}

void calc()

{

Si=(p\*t\*r)/100;

}

void show()

{

cout<<"\nsimple interest:="<<si;

}

};

void main()

{

Simple s;

clrscr();

s.getdata();

s.calc();

s.show();

getch();

}

1. //program to find simple interest.

#include <iostream.h>

#include <conio.h>

class simple

{

private:

int p,t;

float si,r;

public:

void getdata()

{

cout<<"enter the value of p,t & r:";

cin>>p>>t>>r;

}

void calc()

{

Si=(p\*t\*r)/100;

}

void show()

{

cout<<"\nsimple interest:="<<si;

}

};

void main()

{

Simple s1,s2,s3;

clrscr();

s1.getdata();

s1.calc();

s1.show();

s2.getdata();

s2.calc();

s2.show();

s3.getdata();

s3.calc();

s3.show();

getch();

}

1. //program to find simple interest.

#include <iostream.h>

#include <conio.h>

class simple

{

private:

int p,t;

float si,r;

public:

void getdata()

{

cout<<"enter the value of principle amount:";

cin>>p;

cout<<"enter the time:";

cin>>t;

cout<<"enter the value of rate:";

cin>>r;

}

void calc()

{

Si=(p\*t\*r)/100;

}

void show()

{

cout<<"\nsimple interest:="<<si;

}

};

void main()

{

Simple s;

Int i;

clrscr();

for(i=1;i<=3;i++)

{

s.getdata();

s.calc();

s.show();

}

getch();

}

1. //program to find student details.

#include <iostream.h>

#include <conio.h>

class student

{

private:

int rollno,total;

float per;

char name[10];

public:

void getdata()

{

cout<<"enter the name:";

cin>>name;

cout<<"enter the rollno:";

cin>>rollno;

cout<<"enter the subject kannada:";

cin>>kan;

cout<<"enter the subject English:";

cin>>eng;

cout<<"enter the subject Hindi:";

cin>>hin;

}

void calc()

{

Total=kan+eng+hin;

Per=total/3;

}

void show()

{

cout<<"\nstudent name:="<<name;

cout<<"\nstudent rollno:="<<rollno;

cout<<"\nTotal marks:="<<total;

cout<<"\n percentage:="<<per;

}

};

void main()

{

student s;

clrscr();

s.getdata();

s.calc();

s.show();

getch();

}

1. //program to find commission of sales.

#include <iostream.h>

#include <conio.h>

class commission

{

private:

int code,qty;

float rate,comm.,amount;

char product[10];

public:

void getdata()

{

cout<<"enter the code:";

cin>>code;

cout<<"enter the product:";

cin>>product;

cout<<"enter the quantity:";

cin>>qty;

cout<<"enter the rate:";

cin>>rate;

}

void calc()

{

Amount=qty\*rate;

Comm.=amount\*0.01;

}

void show()

{

cout<<"\ncode:="<<code;

cout<<"\nproduct:="<<product;

cout<<"\n quantity:="<<qty;

cout<<"\n rate:="<<rate;

cout<<"\n amount:="<<amount;

cout<<"\n commission:="<<comm;

}

};

void main()

{

student s;

clrscr();

s.getdata();

s.calc();

s.show();

getch();

}

1. //program to find employee salary details.

#include <iostream.h>

#include <conio.h>

class accounts

{

private:

int code;

float basic, hra, da ,pf, lic ,ded, gross, ns;

char name[10];

public:

void getdata()

{

cout<<"enter the employee name:";

cin>>name;

cout<<"enter the code:";

cin>>code;

cout<<"enter the basic salary:";

cin>>basic;

}

void calc()

{

Hra=basic\*0.3;

Da=basic\*0.2;

Pf=basic\*0.1;

Lic=500;

Gross=basic+hra+da;

Ded=pf+lic;

Ns=gross-ded;

}

void show()

{

cout<<"\n employee name:="<<name;

cout<<"\n employee code:="<<code;

cout<<"\n net salary:="<<ns;

}

};

void main()

{

Employee e;

Clrscr ();

e.getdata ();

e.calc ();

e.show ();

getch ();

}

1. //program to find bank account details.

#include <iostream.h>

#include <conio.h>

class accounts

{

private:

int accno;

float bal, interest, amount;

Char name [10], account [10];

Public:

void getdata()

{

cout<<"enter the account holder name:";

cin>>name;

cout<<"enter the account:";

cin>>account;

cout<<"enter the account number:";

cin>>accno;

cout<<"enter the balance:";

cin>>bal;

}

void calc()

{

interest=bal\*0.1;

amount=bal+interest;

}

void show()

{

cout<<"\naccount holder name:="<<name;

cout<<"\naccount type:="<<account;

cout<<"\naccount number:="<<accno;

cout<<"\naccount balance:="<<bal;

cout<<"\n interest:="<<interest;

cout<<"\n total amount:="<<amount;

}

};

void main()

{

accounts a;

Clrscr ();

a.getdata ();

a.calc ();

a.show ();

getch ();

}

1. //program function returns values type of distance.

#include<iostream.h>

#include<conio.h>

class distance

{

private:

int feet;

float inches;

public:

distance()

{

feet=0;

inches=0.0;

}

distance(int ft,float in)

{

feet=ft;

inches=in;

}

void getdist()

{

cout<<"\n enter feet:";

cin>>feet;

cout<<"enter inches:";

cin>>inches;

}

void showdist()

{

cout<<feet<<"\"\_"<<inches<<"\"";

}

distance add\_dist(distance);

};

distance distance::add\_dist(distance d2)

{

distance temp;

temp.inches=inches+d2.inches;

if(temp.inches>=12.0)

{

temp.inches-=12.0;

temp.feet=1;

}

temp.feet+=feet+d2.feet;

return temp;

}

void main()

{

distance dist1,dist3;

distance dist2(11,6-25);

clrscr();

dist1.getdist();

dist3=dist1.add\_dist(dist2);

cout<<"\ndist1=";dist1.showdist();

cout<<"\ndist2=";dist2.showdist();

cout<<"\ndist3=";dist3.showdist();

getch();

}

1. /\*//program to illustrates function overloding.

//function volume () is overloaded three times.

Using namespace std;\*/

#include<iostream.h>

#include<conio.h>

int volume(int);

double volume(double, int);

long volume(long , int, int);

int main()

{

clrscr();

cout<<volume(10)<<"\n";

cout<<volume(2.5,8)<<"\n";

cout<<volume(100,75,15)<<"\n";

getch();

return 0;

}

int volume(int s)

{

return(s\*s\*s);

}

double volume(double r, int h)

{

return(3.14519\*r\*r\*h);

}

long volume(long l, int b, int h)

{

return(l\*b\*h);

}

1. //program to demonstrate function overloading.

#include<iostream.h>

#include<conio.h>

int volume(int s)

{

return(s\*s\*s);

}

double volume(double r, int h)

{

return(3.142\*r\*r\*h);

}

long volume(long l ,int b, int h)

{

return(l\*b\*h);

}

void main()

{

clrscr();

cout<<"volume of a cube="<<volume(10)<<"\n";

cout<<"volume of a cylinder="<<volume(2.5,8)<<"\n";

cout<<"volume of a cuboid="<<volume(100,75,15)<<"\n";

getch();

}

1. // program to demonstrate passing the program in a function.

#include<iostream.h>

#include<conio.h>

class myclass

{

int a,b,sum;

public:

void getdata(int p,int q=5)

{

a=p;

b=q;

}

void calc()

{

sum=a+b;

}

void show()

{

cout<<"sum="<<sum;

}

};

void main()

{

myclass m;

clrscr();

m.getdata(7);

m.calc();

m.show();

getch();

}

1. //program to demonstrate scope resolution operator.

#include<iostream.h>

#include<conio.h>

class myclass

{

long int p;

int t;

float si,r;

public:

void getdata();

int calc();

};

void myclass::getdata()

{

cout<<"enter the values of p,t & r:";

cin>>p>>t>>r;

}

int myclass::calc()

{

return((p\*t\*r)/100);

}

void main()

{

myclass m;

clrscr();

m.getdata();

cout<<"simple interest="<<m.calc();

getch();

}

1. //program to demonstrate scope resulation operator.

#include<iostream.h>

#include<conio.h>

class myclass

{

long int p;

int t;

float si,r;

public:

void getdata();

int calc();

};

void myclass::getdata()

{

cout<<"enter the values of p,t & r:";

cin>>p>>t>>r;

}

int myclass::calc()

{

return((p\*t\*r)/100);

}

void main()

{

myclass m[3];

int i;

clrscr();

for(i=1;i<=3;i++)

{

m[i].getdata();

cout<<"\nsimple interest="<<m[i].calc();

}

getch();

}

1. //program to demonstrate objects as orguments.

#include<iostream.h>

#include<conio.h>

class time

{

int hours;

int minutes;

public:

void gettime(int h,int m)

{

hours=h;

minutes=m;

}

void puttime(void)

{

cout<<hours<<"hours and";

cout<<minutes<<"minutes"<<"\n";

}

void sum(time,time);

};

void time::sum(time t1,time t2)

{

minutes=t1.minutes;t2.minutes;

hours=minutes/60;

minutes=minutes%60;

hours=hours+t1.hours+t2.hours;

}

void main()

{

time t1,t2,t3;

t1.gettime(2,45);

t2.gettime(3,30);

t3.sum(t1,t2);

clrscr();

cout<<"t1=";t1.puttime();

cout<<"t2=";t2.puttime();

cout<<"t3=";t3.puttime();

getch();

}

1. friend function:=function

which is used to access the private properties of class.

1) a friend function is indicated by a keyword"friend"

2)a friend function is declared withinthe class defined outside the class

3)while definding we will not use a keyword friend.\*/

//program a function friendly to use two classes.

#include<iostream.h>

#include<conio.h>

class sample

{

int a;

int b;

public:

void setvalue()

{

a=25;

b=40;

}

friend float mean(sample);

};

float mean(sample s)

{

return float(s.a+s.b)/2.0;

}

void main()

{

sample x;

clrscr();

x.setvalue();

cout<<"mean:"<<mean(x)<<"\n";

getch();

}

1. /\*//inline function:= a function defined in a line or a two.

A functions uses a keyword inline a function has a return type inline

Functions executes directly without copying it to main function.\*/

//program to demonstrate inline funcation

#include<iostream.h>

#include<conio.h>

inline double si(long p,int t,float r)

{

return ((p\*t\*r)/100);

}

void main()

{

int p=1200,t=6,r=7.5;

clrscr();

cout<<"simple interest="<<si(p,t,r);

getch();

}

1. //program to demonstrate inline funcation

#include<iostream.h>

#include<conio.h>

inline double si(long p,int t,float r)

{

return ((p\*t\*r)/100);

}

void main()

{

long int p=12000;

int t=6;

float r=7.5;

clrscr();

cout<<"simple interest="<<si(p,t,r);

getch();

}

1. //program to demonstrate simple interest using inline funcation.

#include<iostream.h>

#include<conio.h>

inline double si(long p,int t,float r)

{

return ((p\*t\*r)/100);

}

void main()

{

clrscr();

cout<<"simple interest="<<si(12000,5,9.5);

getch();

}

1. //program to find biggest of two no's using inline funcation.

#include<iostream.h>

#include<conio.h>

inline int big(int a,int b)

{

if(a>b)

return a;

else

return b;

}

void main()

{

clrscr();

cout<<"biggest of a number="<<big(6,9);

getch();

} \*/

1. /\*//program to find maximum of two no's using inline funcation.

#include<iostream.h>

#include<conio.h>

inline int max(int a,int b)

{

if(a>b)

return (a);

else

return (b);

}

void main()

{

Clrscr ();

Cout<<"maximum of a number="<<max(6,9);

getch ();

}

1. //program to find biggest of two numbers using conitional operator

#include <iostream.h>

#include <conio.h>

Void main ()

{

Int a,b;

Clrscr ();

Cout<<"enter two numbers:";

Cin>>a>>b;

(a>b)?cout<<"\n a is big ":cout<<"\n b is big";

getch();

}

1. **Constructer**: = A constructer is a special type of which shares the name of the class it initializes the object to zero. We may have more than one constructer in a class

**Destructor**:= it destructor the constructor

**Syntax**:= an constructor().

**Type of Constructor**:= 1) normal constructor,

2) parameterized constructor

3) Overloaded constructor,

4) copy constructor.

1) **constructor**:= program

#include<iostream.h>

#include<conio.h>

Class construct

{

Int a, b, sum;

Public:

Construct () {}

Construct (int p, int q) 🡪 (parameterized)

{

A=p;

B=q;

}

Int calc()

{

Return(a+b);

}

};

Void main()

{

Construct c(4,5);

Clrscr();

c.calc();

cout<<sum=”<<c.calc();

getch();

}

**Inheritance**: = Acquiring the properties of class to another class reusability of the code can be achieved through inheritance.

**Types**:= **1)single inheritance** A-B(base-Derived),

2)**Multiple Inheritance** A-C-B, A single derived class inheritance the properties from more than one base class.

**3) Multiple-Multilevel-Inheritance** A-B-C. Inheritance takes place in more than one level.

4) **Hybrid-Inheritance** A-B-D-C. Combination of single multiple & multilevel inheritance.

**5) Hierarchical inheritance** Reverse of multiple inheritances more than are more than one derived class inheritance the properties from single base class. B-A-C.

1. //program to demonstrate to find the sum and average of three number using inheritances.

#include<iostream.h>

#include<conio.h>

Class sum

{

Protected:

Int a, b, c, s;

Public:

Void getdata ()

{

Cout<<”enter the value of a , b & c:”

Cin>>a>>b>>c;

}

Void add()

{

S=a+b+c;

}

};

Class average: public sum

{

Float avg;

Public:

Void d ()

{

Avg=s/3;

}

Void show();

{

Cout<<”sum=”<<sum;

Cout<<”average=”<<avg;

}

};

Void main ()

{

Average a;

a.getdata ();

a.add ();

a.d

a.show ();

getch();

}

1. //program to demonstrate unique inheritance.

#include<iostream.h>

#include<conio.h>

Class base

{

Protected:

Int a, b, sum;

Public:

Void getdata ()

{

Cout<<”enter the value of a and b:”

Cin>>a>>b;

}

Void calc ()

{

Sum=a+b;

}

};

Class derived

{

Public:

Void show ()

{

Cout<<”sum=”<<sum;

}

};

Void main()

{

Derived d;

d.getdata ();

d.calc ();

d.show ();

getch();

}

1. **Interchange**

//program to demonstrate class template with multiple parameters.

#include<iostream.h>

#include<conio.h>

Class construct

{

T1 a;

T2 b;

Public:

test (T1 x, T2 y)

{

A=x;

B=y;

}

Void show ()

{

Cout<<a<<”and”<<b<<”\n”;

}

};

Void main ()

{

Test<float, int>test1 (1.23, 123);

Test<int, char>text2 (100,’w’);

Clrscr ();

Test1.show ();

Test2.show ();

getch ();

}

**Polymorphism**: = giving an additional meaning to the properties of the class

**Types:= Compile time Polymorphism**

1) Function overloading

2) Operator overloading

Operator overloading giving an additional meaning to operator can be achieved through a special function operator

Syntax: =Run type operator-()

{

\_ \_ \_ \_ \_

\_ \_ \_ \_ \_

}

Operator can be overloaded: = +,-,\*, /, ++,--, >>, <<, ==.

Operator cannot be overloaded: =:: , 0,🡪, size().

1. **Runtime polymorphism**

1) Virtual function: =

#include<iostream.h>

#include<conio.h>

Class construct

{

Char name [20];

Float age;

Public:

Person (char \*s, float a)

{

Strcpy (name, s);

age=a;

}

Person & person:: greater(person &x)

{

If (x.age>=age)

Return x;

Else

Return \* this;

}

Void display (void)

{

Cout>>”name:”<<name<<”\n”;

Cout>>”\nage:”<<age<<”\n”;

}

};

Int main ()

{

Person p1 (“join”`, 37.50);

P2 (“anamed”, 29.0);

P3 (“hebbar”, 40.25);

Person p=p1.greater (p3);

Cout<<”enter person is: \n”;

p.display();

getch();

return 0;

}

1. //program to demonstrate pointer to objects.

#include<iostream.h>

#include<conio.h>

Class city

{

Protected:

Char\*name;

Int Len;

Public:

City ()

{

Len=0;

Name=new char [len+1];

}

Void getname(void)

{

Char \* s;

S= new char [30];

Cout<<”enter city name:”;

Cin>>s;

Name=new char [len+1];

Strcpy(name,s);

}

Void printname(void)

{

Cout<<name<<”\n”;

}

};

Void main ()

{

City \* cptr[10];

Int n=1;

Int option;

Do

{

Cptr[n]=new city;

Cptr[n]🡪getname();

n++;

cout<<”do you want to enter one more name?/n”;

cout>>”(enter 1 for yes 0 for no☺”;

cin>>option;

}

While(option);

Cout<<”\n\n”;

For(int i=1; i<=n; i++)

{

Cptr[i]🡪printname();

}

getch();

}

1. // program to demonstrate new & delete operator.

#include<iostream.h>

#include<conio.h>

Void main ()

{

Int \*p=new int;

Clrscr ();

Cout<<”enter the value of p:”;

Cin>>\*p;

Cout<<”the value of p:”<<\*p;

delete p;

getch(); }

**Templates**: = 1) class template. 2) Member function template.

3) Function template.

4) Program to illustrate the rules of illustrate class template for performing the scalar product of int line illustrate as i-rules as float lqu-illustrates.

5) Class templates with multiple parameter generics data type in a class data.

6) Function templates.

7) Bubble sort using template functions.

8) An application of template functions.

9) Function with two generics types.

**Type conversion**: = 1) conversion of basic to class type through constructor.

2) Through operator overloads.

3) One class to another class conversion.

4) Conversion of one data type to another data type. 1) Basic data type to basic data type

Ex: = conversion of int data type to float data type[built-in]

Ex: = conversion of float data type to int data type int x=5; float y=1.142; y-float(x);

2) basic type to class data types using constructor we can achieve this type of convn in constructor class variables are get initialized.

3) Class type to basic type:= this type of convn can be achieved through operator overloading.

1) operator-fun must be a class member.

2) It must not satisfy a return type.

3) It must not have any arguments. Ex: =operator type name ()

{

\_ \_ \_ \_

Fn body

\_ \_ \_ \_

}

4) Class type to class type: = assigning the properties of one class to another class this can be achieved through copy constructor.

1. Program to demonstrate maximum of two numbers of using function templates.

#include<iostream.h>

#include<conio.h>

Template<class T>

Void max (Ta, Tb)

{

If(a>b)

Cout<<”maximum=”<<a;

Else

Cout<<”maximum=”<<b;

}

Void main()

{

Clrscr();

Int x,y;

Cout<<”enter the values of x and y”<<endl;

Cin>>x>>y;

Max(x,y);

getch();

}

1. **Exception Handling**: = is an error handling technique.

**Syntax**: = try

**{**

\_ \_ \_ \_

\_ \_ \_ \_

}

Catch c exception ()

{

\_ \_ \_ \_

}

Ex: = int a=b, b=5, c=0, x;

Try

{

X=a+b/c; 5+5/0 = 10/0

}

Catch (exception c)

{

Cout<<”caught an exception”;

Cout<< “divided by zero”;

}