1)Introduction of c++:- simple, rich library support, oops, pointers, compiled language, high spped of execution, object oriented programming, platform independent

2)Header files vs Libraries:-.h,.lib (available inside include sub directory, available inside lib sub directory), (we can include them using include command whereas we cannot include libraries,

they are included in last stage using linker) iostream. Header files contains functions like printf, scanf. But actual code was inside librararies. Declaration inside header file,Definition inside library.

human readable, non human readable

3)Basic structure of c++

4)<< , >>, ::

5)#include is a specific preprocessor command that effectively

copies and pastes the entire text of the file, specified between

the angle brackets, into the source code.

6)HLL VS LLL:- easy to understand, simple to debuag, maintain. human friendly, portable,

7)Variable, Data types and its kinds

8)Global variables vs Local variables

9)Scope

10)Operators

11)Type catsing:- conversion of data type of a variable from one tpe to other. Implicit(automatic type conversion) and explicit(type casting, done by assignment or by using cast operator static\_cast<int>(f)).

lower data types are converted into higher data types in an expresiion.

12)constants

13)Manipulators:-

->cout<<std::boolalpha<<endl,cout<<std::noboolalpha[used to print true, false]

->hex,oct,dec

->showbase

->noshowbase

->showpos

->noshowpos

->uppercase

->nouppercase

->setw(5)

->setfill('\*')

->left

->right

14)Precedence and associativity

15)control structures :- sequential, selection, repetative

16)endl flushes the buffer

17)Escape sequences

18)inline function

19)call by refeernce

20)Difference between c and c++:- type of language, number of keywords, polymorphisim, inheritence, encapsulation, data and funcations are seperated,

use of functions inside structures, Data hiding, Virtual and friend functions, DMA functions, functions used for inut output, access modifiers

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

c++ prgrams execution starts from the main function

cout represents standard output stream in c++

cin represnts standard input stream in c++

cin and cout are predefined objects in c++

The operators << and >> are overloaded bitwise operatots

<< - insertion or put-to operator

>>extraction or get from operator

c++ supports reference variable

A reference variable provides an alternative name for a variable

concept of namespace is introduces by c++

1)c++ supports many oop features like objectsm polymorphisim, encapsulation, data abstraction, dynamic binding

2)anci c++ include many new keywords such as try, catch, class, mutable, inline etc

3)Although c++ is oop language we can create programs in c++ without any class

4)c++ allows declarartion of variable anywhere in the scope

5)c++ supports operator overloadung

6)c++ manipulators helps in formatting data

7)c++ contains more than 60 keywords

8)c++ provides exception handling

9)c++ provide data security

10)c++ provides different visibility modes: public, private, protected

11)we may assign float value to integer variable and vice virsa

12)A variable is a memory location that is used to store data value. variable has ots size, range, name(identifier) and address

13)c++ supports initialization of multiple variables in one statement

14)in c++,integers and character are allowed to express as hexadecimal or octal

15)identifiers in c++ are case senitive

16)identifiers first character must be an alphabet or underscore

17)Exceept \_ identifier must not contain any other symbol

18)any aithematic operation bw two integers gives integer result

19)any aithmetic operation between two floats gives floating point result

20)Any airthmetic operation between one integer and float gives floating point result

21)in c++ we cannot display division % to floating point values

22)while performing modulo division, the sign of result is always the sign of first operand

23)Relational operators gives logical values as output

24)we cannot apply increament and decreament operators to constants

25)The operator operates on two operands is known as binary operator

26)c++ doesnot have any operator for exponentiation

27)Bitwise operators may not be applied float or double values

28)typedef is used to assigning new name to exisiting data type

29)keyword const is used to make the value of a variable constant

30)keyword volatile is used to tell the compiler that the value of a variable may be changes by some external sources any time

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

sorage classes:- automatic, register, external(global), static

scope is defined as region in which variable is available for use

lifetime of a variable may be defined as the duration of time upto which a variable exisits in the memory during execution

----------

1)auto is used to declare an automatic variable

2)any variable inside a function without storage class is an automatic variable by default

3)automatic variables are local to function , they are created when a function is called and destroyed automatically on the function exit

4)keyword register is used to declare register variable

5)If regsiter variable cannot get the register to store the value then they automatically caonverted into automatic variable

6)we cannot get the address of register variable using address of operator

7)External or global variables are alive throughpur the program and acccessed by any function in the program

8)As the external variable is accessible from all functions , any function can use it and change its values,in this way, subsequent functions always get updated value of global variable

9)The local varuables will have precedence over global varaibles, if both variables have the same name

10)global or external variables may be declared as static

11)static variables aredeclared using static

12)lifetime of static variable is same as lifetime of program

13)static variables are accessible by the function inside which it is declared

14)stati variable initializedd only once

15)visibi;ty of static variable is same as the visibility of automatic variable and life time is same as life time of global variable

16)in c++, static variables can be initialized with a non constant expression

17)Default value for automatic and resgister variables is garbage and for externl and static variable is 0

18\_We cannot define a variable with more than one storage class

19)We cannot use static and auto storage class as the function argument

20)If we not mention any datatype with storage class then it will be treated as integer by default

21)we may use :: to access lobal variable when we have a local variable having same name as global variable

22)keyword extern is used to tell the compiler that variable is declared already elsewhere and no need to create storage space for it

23)mutable keyword is used to specify that the meber may be updates or modified even if it is member of constant object

24)Mutable specifier cannot be used with variable that is declared as static or const

25)static data member of class must be defined outside the class to allocate memory

26)All teh static data member are initialized to 0 by default

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

1)in if-else statemtn only one portion of code is executed i.e; if the condition is true then code blok of if is executed. If it is false else block is executed

2)we may use if without else

3)we can use conditional operator to replace if elase statemnts

4)any non zero number is treated as true

5)In switch statment only integer or character expression is allowed

6)switch case: when we use character constatnts as switch case they are treated as integers

7)switch statment doesnot support float point values

8)switch case: all cases must be constant or constant expressions i.e; variables are not allowed as case value in switch

9)switch case: only one default case is allowed in switch

10)no case label can have same value

11)default case is optional

12)break statement in switch is optional.

13)ordering of cases in unimportant, u can define cases in any order

14)nested switch statemnts are supported by c++

15)nested switch may have same constants

16)goto is an unconditional statement in c++ and by using it we can perform bacjward as well as forward jump in our program

17)for and while loop are entry controlled loop because teh test condition is evaluated first and then the code is executed

18)for loop consists of three parts i.e, initialization, condition, and increament

19)for loop supports variatiosn but it must have basic structure retained i.e, it must have two semicolons

20)Empty test condition is not allowed in while loop

21)do while is a post test condition loop and it will execute atleast once even if its condition is false

22)do while loop is an exit controlled loop because the code of the loop is execured first and then teh test condition is evaluated

23)do while loop executes at least once as it is an exit controlled loop

24)do while loop os basically used in menu selection functions because one will always want a menu function to execute atleast once

25)keyword break is used to get out of the switch statemtn

26)keyword break is also used with thr loop to break/ter,minate the execution of the loop even if the loop test condition is true

27)break used in a swtch / loop will affect only that switch /loop, it doesnt affect any switch or loop in which it is nested

28)continue is used to come out of the loop for a particukar iteration , when continue executes the programs control goes to begining of the loop by skipping further statements

29)continue does not work with switch statement. However if switch us placed inside any loop then continue in switch case is valid and willcontinue loop

30)

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

1)Array is a collection of variables having similar data type

2)Array elements are always storde in contigious memory locations

3)Array data type is derived data type and is derived from primary data type

4)Array indexing always starts from 0

5)The size of an array should be numeric or a symbolic constant

6)Referncing array element outsid the declared limits will not give error but produce unpredicatble result

7)array elements contain garbage values if we cannot initialize array at the time of initalization

8)If we declare atleast one elemnt of array at te time of arry declaration, then rest of elements are initilized by zero automatically

9)it is a compile time error if we initialize more elements than size of an array

10)c++ doesnot perfrom any bound checking

11)name of array without any index specify its base address i.e. the address of first element

12)we cannot compare to arrays using comparsion operator, if we want to do so, we have to compare elements of both arrays individualy

13)Array is always pass by address as teh function argument i.e. we cannot pass an entire array as an argument to a function, we pass its base address

14)Array can be accessed using a pointer variable

15)Array can be declared as member of a class

16)c++ supports multi dimensional array

17)In two d arrsy first set of bracket represent rows and second represents column

18)At the time of initialization of md arrays, it is an error to omit the array size for any dimension other than first

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

1)c++ doesnt support built in string data type

2)in c++, there are two kinds of strings that are commonly used

a)C style string

b)string class in ANSI c++

3)dealing c strings is complex and inefficient as compared to the string objects of standard string class

4)The string class provide many constructors and functions for performing operations on the string

5)we can use relational operators with the string objects for verifing relations between two string objects

6)getline() function is used to read a line of text from user containing white spaces getline(cin,string\_name)

7)length() function return length of a string str.length()

8)append() function us used to append string to other string str.append(new\_str)

9)assign() function is used to assigns a partial string str.assign(new\_str)

10)at() or get\_at() function return the character stored at specified location str.at(index\_no)

11)empty() function is used to check whether a string object is empty or not. return true if empty. str.empty()

12)erase() function is used to remove characters from string object str.erase(pos,char\_count)

13)find() function is used to find occurrence of a specified substring in a string. str.find(sub\_str);

14)insert() function s used to insert a character at specified location. str.insert(pos,str)

15)maxsize() function return maximum possible size of a string object in that system

16)replace() function replaces a specific character with a given string

17)resize() function is used to change the size of string as specified

18)size() function returns number of character in string

19)swap() function is used to swap a string with other string

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

-

1)Break, continue

2)Pointer[p1+5,p1-5,\*p1+\*p2,\*p1 \* \*p2,\*p1 + 2, \*p1/\*p2 + 5,p1+1(pointer not moved)],Arrays. pointer is a derived data type which is used to strore address of a variable

3)structure, typedef(used for defining users data type,typedef is used for defining our own data type and typedef is a kw used to assign new name to existing DT)

4) unions(can handle one member at one time, shares common memory location),enum is user defined data type which can be assigned some user defined data types.

5)Data memberes inside structure can be variables, nested structures, arrays,pointers etc...

6)call by reference using pointers

7)15.cpp,28.cpp,29.cpp,

Opeartions on pointers:-

a)adding integer to pointer

b)subract integer from pointer

c)subtract one pointer from other pointer

operations not allowed on pointer:-

a)addition of pointers

b)multiply/ divide a pointer in an expression

Structure is user defined data type which is a collection of hertrogeneous data type. c++ structure can have data members, member functions, pointers,references....

typedef is user defined data type which is used for defining our own data type and typedef is a kw used to assign new name to existing DT

Union is user defined data type which shares a common memory locations and it is useful when we want only one value in memory

Enum, short for "enumerated," is a data type that consists of predefined values. A constant or variable defined as an enum can store one of the values listed in the enum declaration.

1)structure provides a way of grouping varibles of different data types uder a single name. It is a way of creating user defined data type

2)For declaring structure we use struct

3)The name of s atructurfe is called tag and the fields of a structure is known as structure elments or members

4)We can access membersof a strcutre using dot operator and selection operartor

5)we cannot initialize members of structure inside structure template

6)We may declare a structure variable at the time of defintion of a strcutreu by placing in between closing brace and semicolon

7)The tag name of structure is optional

8)Between two members of a stucture some space in byte is kept unoccupied which is known as slack byte

9)We can assign the data of one structure to other structure variable of same type using assignment operator

10)c++ doesnot permit any relational operations on structure variables i.e, we cannot perform equality check between two structure variables using comparsion operator

11)c++ supports array of structure variables

12)c++ supports nesting of structures

13)structure varuables is used as structure member inside other structure

14)We can pass structure variable to the function argument

15)We cannot use a structure variable as a member of its own struct type variable

16)We cannot assign structure of one type t the structure of other type

17)passing structure variable to a functions by pointer is more efficient as compare with passing by value

18)c++ allow pointer to structure

19)structure elements are always arranges in contigious memory allocation

20)In c++ we may define function inside structure

21)struactue member are bu default private

22)members of structure are public by default but we can ccess specifieres in order to modify accessability of members

23)Concept of union adopted from structure

24)For declaring union we use union keyword

25)The member of structure has its own storage allocation

26)All member of union use the shared/common mmeory location

27)At any point of time only one union member contains valid data

28)

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

1)pointer is a special type of variable that stores address of a variable

2)for declaring pointer we use \* along with the data type and variable name

3)pointer helps in DMA

4)pointer are more efficient in handling arrays

5)pointer provide flexibility, more power, increase in execution speed of c++ program

6)\* is known as indirection operator or dereferncing operator

7)pointer variable is initialized with garbage value by default

8)A single pointer variable can point to different variable in different statements of the program

9)assigning constant to a pointer giver unpredicted result

10)the value of a variable cannot be assigned to a pointer

11)storeing adress of one type into pointer of another type gives incorrect result

12)we can create a pointer that points to other pointer

13)c++ supports generic/ void pointer

14)we can perform addition or subtraction on pointer variables. when we do so pointer points to next valid address

15)we cannot perform multiplications or divisions on a pointer

16)we can compare pointers of same using reational operators

17)Two pointer variables cannot be added

18)c++ supports pointer to a function

19)we can pass pointer to a function as argument

20)we can create an array of pointer variables

21)we can declare pointer to a structure

22)we can create pointer to a class

23)-> used to access members of a class with the pointer objects

24)A single pointer variable is used to point objects of different classes

25)we can declare pointer as a class member

26)this pointer is an object pointer that points to currently calling objects

27)void pointer cannot be derferenced as it has no object

28)pointer that points to null or noyhinmh is known as null pointer

29)uninitilazed pointer is also known as wild pointer

30)when a pointer is pointed to a memory location of deleted varuables is known as dangling pointer

41)wild pointer, NULL pointer,Generic ponter, Constant pointer, Dangling pointer

a)uninitialized pointer is known as wild pointer. points towards anonymous memory location location. solution is to initialize wild pointer with some valid address value

b)Null pointer points to zero as an address value or it points no where initially but we can assign some valid address to null pointer in later stages of program.

->we cannot dereference null pointer and we csn asssign 0 as an address value if we dont know the addreess initially. It is better to have a null pointer instead of wild pointer

c)Generic pointer(void). declared using void keyword. can point to any type of variable. while dereferencing void pointer typecasting is must.

d)constant pointer points toward only one memory location in its entire life time. If we try to change the address hold by pointer, compilation error may raise

e)dangling pointer is a pointer which is pointing towards already deleeted memory location. solution is assign the pointer to the null valur.

42) We cannot add, multiply, divide,modulo two pinter vaiables

43)We cannot modulo, mutiply, divide a constant from pointer variabe

44)We can increament or decreament a pointer variable

45)we can add or sub a constrant from a pointer variable

46)We can subtract two pointer if they are pinting towards samr arrsy

47)We can compare two pointer if they points towrds same array

48)

1)private data members and member functions can be accessed within a class but cannot be accessed outside the class()

2)Data members are by default private.

3)In structures there is no provacy to data. But In c++, there is a concept of classes, which helps in data security

4)Private member functions can be acceesed inside public functions of those which are declared inside class but defined outside the class

5)We can declare private member functions inside class and we can define them outside the class

6)Data members inside class should be variables, arrays, pointers, classs itself............same as structure

7)void className :: FunctionName()

8)static variables can intitilized only once. If we want to initialize them we have to do outside class using above syntax. DV is 0

9)we cannot access non static data members inside static functions. static void display(){}

10)object is not necessary to call static function. We can call it with object also. className :: functionName()

Objects can be passed as an arguments

11)Friend Function

a)friend return\_type function\_name(complex o1,complex o2);//should be in class from where we are accessing private members

means that non member - sumcomplex function is allowd to do anything with private data mambers

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

friend data\_type function\_name();

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

class\_name function\_name();//syntax of friend function outside class

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

b)since it is not in the scope of the cl;ass, it cannot be called from the object of that class. c1.sumcomplex() is not allowd

c)can be invoked without the help of any object

d)usually contains objects as arguments

e)can be declared inside private or public section of class

f)cannot access the members directly by their name and need objcet\_name.member\_name to access any member

12)How to access private member of a class in other class function??10\_.cpp 15.cpp

13)18.cpp 20.cpp

14)Swapping values using Friend function

15)Constructors

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

To make a function that is declared outside the class “friendly” to that class, we have to declare the function as a friend function, as seen below: and in this scope resolution is not necessary

class className

{

friend returnType functionName(arg list);

};

returnType functionName(arg list)

{

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

}

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

A friend class can have access to the data members and functions of another class in which it is declared as a friend.

They are used in situations where we want a certain class to have access to another class’s private and protected members.In this case scope resolution is used because, we are making functions of class P

to access data members of class P

class S; //forward declaration

class P{

// Other Declarations

friend class S;

};

class S{

// Declarations

};

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

1)A class is a blue print from which objects are created.class contains data members to store data and member function to operate on data

2)class is a group of objects having similar properties

3)class doesnot actually define any data, but it actually does define, what an object of the class consists of and what operations can be performed on such object

4)Idea of classes is the extension of structure concept c

5)class is a user defined data type

6)keyword class is used to declare class

7)The functions of class are known as member functions and member of data are known as data members

8)In c++, members of data are private by default

9)we can define all member functions inside as well as outside the class

10)In c++ class definciton must be terminated with semi colon

11)An objet s an instance of a class. Basically it is a real world implementation of the class having all those properties and data that are defined or structured inside the class

12)The objects of same class have same data members but may have different values for a particular value

13)The memory space for object is allocated when tjey are created

14)The memory space for data members are allocated speratey for each object, but no eperate memory is allocated for the member functions

15)we can access members of a class by using dot operator along with the object of that class

16)we can declare an array of objects

17)objects can be passed as function arguments

18)An object can be return by a function

19)we can copy data of one object to the data of same class using assignment operatr

20)Access specifiers define the visibility and accessibility of the members of a class

21)member functions declared inside a class are inline functins by default

22)Private members of a class are accessible only from the other members of the same class within the class

23)protected members are accessible from members of theri same class and from members of their derived class

24)public members are accessed anywhere

25)we can declare data members and member functions of a class as static

26)A const member function is a function that cannot modify or alter any data of a class member

27)Friend function is a function with the helpof which we can access private members of a class

28)static data members of a class are also known as static members

29)c++ supports local classes

30)A local class can use global and static variables declared inside function or blockbut cannot use local or automatic variables

31)local functions cannot have static data members

32)the member function of a class must be defined inside the local class

33)we can create pointer to object class

34)class provides data hiding

35)c++ supports nested class

36)An abstract class is a class whose object cannot be craeted

37)an abstract class is a class which have only one pure virtual function

38)a derived class can never be made an abstract class

39)

1)constructor is called as soon as object is created

2)Constructor with default arguments is possible

3)constructor overloading, if inheritence is not there, we can easily guesss which constructor will be called

4)Dynamic initialization of constructor:- After giving input, it will decide which constructor to call

5)Default constructor is not needed if we pass all parameteres at the time of creating object

6)Copy constructor:- ClassName Object(Object1); className(ClassName &obj);

No problem even if we remove copy constructor

7)Different ways of calling copy constructor

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

#include<iostream>

using namespace std;

class number

{

int a;

public:

number(){a=0;}//default constructor is mandatory

number(int num)

{

a=num;

}

number(number &ob)//will work properly even this code is removed

{//if copy constructor is not there, compiler will supply its own copy constructor

cout<<"copy constructopr called"<<endl;

a=ob.a;

}

void display()

{

cout<<"a's value is "<<a<<endl;

}

};

int main()

{

number x,y,z(45),z2;

x.display();

y.display();

z.display();

number z1(x);//copy constructor invoked

z1.display();

z2=z;//copy constructor will not called

number z3=z;//copy constructor called

return 0;

}

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

example obj1(50);

example obj2(obj1);

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

example obj1(50);

example obj2;

obj2=obj1;

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

example obj1(50);

example obj2=obj1;

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

8)Destructor called automatically when an object is destroyed

9)Constructors object is automatically woke up when an object is created

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

1)constructor is a special member function that gets called automatically whenever an object of its class is created

2)constructor is responsible for the initialization and memory allocation for thr objects of its classes

3)constructor has same name as class name and have no return type even void

4)when no constructor is defined in the class the compiler itself build one, knwn as implicit r default constructor

5)constructor with no default arguments is known as default constructor

6)generally constructor should be declared in public

7)constructors can be defined inside as well as outside a class

8)constructors cannot be inherited

9)we can call constructor of base class from derived class

10)In c++, constructor may have default constructors

11)constructors cannot be declared as virtual

12)we cannot refer to address of any constructor

13)constructor can make implicit call to new and delete operators when memory allocatin is required

14)constructor taking arguments is known as parameterized constructor

15)we can call parameterized constructors by two ways:-

a)implicit calling [demo obj(1,2) obj.show()]

b)explicit calling [demo d2=demo(10,20) d2.show()]

16)constructors can be overloaded

17)constructors can take a reference to its own class as a arameter, this type is known as copy constructor

18)constructor can take parameters of any type except that of the class class but it may take a reference to its own parameter

19)compiler supplies default copy constructor

20)when an object is created and initialized at the same time, copy constructor gets called

21)when a derived class constructor is created, its base constructor is called before its own onstructor

22)constructors of virtual base class is invoked before any non virtual base class

23)When an object is passed or returned to/from a function by value, copy constructor called automatically

24)Destructor is a special member function that is used to destroy or deallocates the objects that are called by the constructor

25)Destructor have the same name as the class preceded by ~

26)Destructors cannot takeany arguments and doesnot return any value

27)Destructors are called in reverse order as off the order of constructor calls

28)constructors may be declared as private. objects of a class having a private (non-public) constructors cannot be called directly

29)Destructor an be declared as private

30)when a class have private constructors or destructors then only dynamic objects of that class can be created

31)Destructors can be declared as virtual

32)Destructors can be virtual

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

10)Inheritence(for reusability- DRY- Don't repeat youself

11)Single inheritence

12)Multiple inheritence(mutiple base classes)

13)Hierarchial inheritance(multiple derived classes)

14)Multilevel inheritence

15)Hybrid inheritance

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

Private data members are never inherited

private visibility mode: public members of the base class becomes private members of the derived class

public visibility mode: public members of the base class becomes public members of the derived class

class newClassName : visiilityMode BaseClassNAme

Protected data members are same as private data members but they can be accessed outside the class

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

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

public derivation private derivation protected derivatiom

1.private members: not inherited not inherited not inherited

2:protected members: protected private protected

3:public members: public private protected

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

Compiler will not create default constructor if we have any other constructor in the class

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

1)When a class inherits or acquires the properties of other class , this is know as inheritence

2)parent class/ base class/superclass

3)child class / derived class/ base class

4)It provides the concept of reusability as it is a process of creating new classes from existing classes

5)There are five type of inheritance

6)The private data members of a class are not inheritable

7)We can derive the properties of a base class in one of the three visibility modes: public, private or protected

8)When base class is publicaly inherited by derived class, the public members of base class are public members of derived class

9)

10)multipath inheritence (combination of Heirarchial and multiple inheritence)

11)In multi path inheritence the common base class is declared as virtual base class inorder to avoid duplication of inherited members

12)construtors and destructors never get inherited

13)polymorphisim means many forms

14)there are two types of forms:- compile time polymorphisim, run time polymorphisim

15)CTP achieved using function overloading and operator overloading

16)RTP achieved using function overriding via virtual functions'

17)In CTP function overloading which overloaded version of the function get invoked decided at compile time

18)In RTP which overloaded version of function gets invoked is decided at run time

19) RTP is achived only when a virtual function is accesssed through a pointer to the base class. If we use dot operator along with object name to call a virtual function, then RTP is not achived

20)

1)Multiple inheritance

2)Ambiguity in multiple inheritence:- happens when we are accessing the member function/data member which is common in both base classes. So when we are calling that function which is common in both classes,

compile dont know which one to call

3)Ambiguity resolvance:- In the derived class we specify which one to call using scope resolution operator

EX: void greet()

{

base1::greet();

}

4)This ambiguity can happen in case of simple inheritence also. But in that case, derived class function will override base class function. so we create object of derived class and access that function we

will access function of derived and ...

5)concept of virtual base class (problem in heirarchial + multiple inheritence )

6)In virtual base class only one copy of data member or member function will be passed

class B:virtual public A

{}

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

Constructors in Derived Class in C++

1)We can use constructors in derived classes in C++

2)If the base class constructor does not have any arguments, there is no need for any constructor in the derived class

3)But if there are one or more arguments in the base class constructor, derived class need to pass argument to the base class

constructor

4)If both base and derived classes have constructors, base class constructor is executed first

->In multiple inheritances, base classes are constructed in the order in which they appear in the class deceleration. For example

if there are three classes “A”, “B”, and “C”, and the class “C” is inheriting classes “A” and “B”. If the class “A” is written

before class “B” then the constructor of class “A” will be executed first. But if the class “B” is written before class “A” then

the constructor of class “B” will be executed first.

->In multilevel inheritance, the constructors are executed in the order of inheritance.

For example if there are three classes “A”, “B”, and “C”, and the class “B” is inheriting classes “A”

and the class “C” is inheriting classes “B”. Then the constructor will run according to the order of inheritance

such as the constructor of class “A” will be called first then the constructor of class “B” will be called and at the end

constructor of class “C” will be called.

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

-->Special Syntax

C++ supports a special syntax for passing arguments to multiple base classes

The constructor of the derived class receives all the arguments at once and then will pass the call to the respective base classes

The body is called after the constructors is finished executing

Syntax Example:

Derived-Constructor (arg1, arg2, arg3….): Base 1-Constructor (arg1,arg2), Base 2-Constructor(arg3,arg4)

{

….

} Base 1-Constructor (arg1,arg2)

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

-->Special Case of Virtual Base Class

The constructors for virtual base classes are invoked before a non-virtual base class

If there are multiple virtual base classes, they are invoked in the order declared

Any non-virtual base class are then constructed before the derived class constructor is executed

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

Case1:-

class B:public A//first base class,then derived class

{

order of execution of constructor->first A() then B()

};

Case2:-

class A:public B,public C//multiple inheritence, depends on which one is inherited first

{

//order of execution of constructor first B(),then C(),then A()

};

Case3:-

class A:public B,virtual public C

{

//order of execution of constructor first C(),then B(),then A()

};

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

Initiallization list in constructors:-used to intialize data members

syntax for initialization list in constructor:

constructor(argument-list):intitialization section

{

assignment + other code

}

🡪

1)Array of objects using pointers

2)Arrow, new operator

3)This operator

4)Polymorphisim:- we cannot overload on the basiis of return type

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

1)Compile Time Polymorphism

In compile-time polymorphism, it is already known which function will run. Compile-time polymorphism is also called early binding, which means that you are already bound to the function call and you

know that this function is going to run. There are two types of compile-time polymorphism:

-->Function Overloading

This is a feature that lets us create more than one function and the functions have the same names but their parameters need to be different. If function overloading is done in the program and function calls are made

the compiler already knows that which functions to execute.

-->Operator Overloading

This is a feature that lets us define operators working for some specific tasks. For example, we can overload the operator “+” and define its functionality to add two strings. Operator loading is also an example of

compile-time polymorphism because the compiler already knows at the compile time which operator has to perform the task.

2)Run Time Polymorphism

In the run-time polymorphism, the compiler doesn’t know already what will happen at run time. Run time polymorphism is also called late binding. The run time polymorphism is considered slow because function

calls are decided at run time. Run time polymorphism can be achieved from the virtual function.

-->Virtual Function

A function that is in the parent class but redefined in the child class is called a virtual function. “virtual” keyword is used to declare a virtual function.

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

5)pointers to derived class

6)Base class ponter can be pointed to derived class object

7)we cannot access derived data members with the help of base class pointer

8)Virtua; function

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

\*

Rules for virtual:-

1)They cannot be static

2)They are accessed by object pointers

3)Virtual functions can be a friend of another class

4)A virtual function in the base class might not be used.

5)If a virtual function is defined in a base class, there is no necessity of redefining it in the derived class

6)if virtual function is not redefined inside derived then base class virtual function will be accessed

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

abstract class:- a class in which atleast one virtual function should be there

->Pure virtual function is a function that doesn’t perform any operation and the

function is declared by assigning the value 0 to it. Pure virtual functions are declared in abstract classes.

->Abstract base class is a class that has at least one pure virtual function in its body. The classes which

are inheriting the base class must need to override the virtual function of the abstract class otherwise compiler will throw

an error.

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

this operator