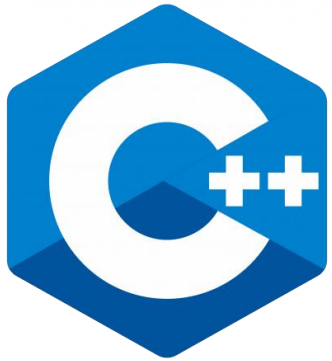

C++ Programming

Trainer : Pradnyaa S Dindorkar

Email: pradnya@sunbeaminfo.com

We did…..

1. Inheritance
2. Protected data member
3. Types of inheritance
4. Mode of inheritance
5. Diamond problemVirtual Function

Today's topics

1. **Virtual function**
2. Function overriding
3. Early Binding and late Binding
4. Pure virtual function and Abstract class
5. Exception Handling
6. Template
7. Difference between Procedure Oriented and Object Oriented
8. MCQ

Virtual Keyword

- **Virtual function** = It is the function which is called depending on type of object rather than type of pointer

- If class contains at least one virtual function then such class is called **polymorphic class**.

- If signature of base class and derived class member function is same and if function in base class is virtual then derived class member function is by default considered as virtual.

Function overriding.

Process of redefining, virtual function of base class, inside derived class with same signature is called function overriding.

- Virtual function redefined inside derived class is called overrided function.
- For function overriding:
    1. Functions must be exist inside base class and derived class.
    2. Signature of base class and derived function must be same.
    3. At least, function in base class must be virtual

Program Demo

Early Binding

create a class Base and Derived (void show() in both classes)

create base *bptr;

bptr=&d;

bptr->show()

Late Binding

create a class Base and Derived (void show() in both classes one as virtual in base class)

create base *bptr;

bptr=&d;

bptr->show()

Pure virtual function and Abstract class

- Virtual fun which is equated to zero such function is called as Pure virtual function
- Pure virtual function does not have body.
- A class which contains at least one Pure virtual function such class is called as "Abstract class".
- If class is Abstract we can not create object of that class but we can create pointer or reference of that class .
- It is not compulsory to override virtual function but It is compulsory to override Pure virtual function
- If we not override pure virtual function in derived class at that time derived class can be treated as abstract class.

Upcasting and downcasting

- Upcasting and downcasting :-

Upcasting - process of converting derived class pointer into base class pointer

or Storing address of derived class object into base class pointer.

eg : Person *p=new student();

Downcasting - process of converting base class pointer into derived class pointer

or storing address of base class object into derived class pointer

eg : Student *s=new person();

Exception Handling

- If we give wrong input to the application then it generates runtime error/exception.
- To handle exception then we should use 3 keywords:
- 1**. try**
  - try is keyword in C++.
  - If we want to inspect exception then we should put statements inside try block/handler.
  - Try block may have multiple catch block but it must have at least one catch block.
- **2. catch**
  - If we want to handle exception then we should use catch block/handler.
  - Single try block may have multiple catch block.
  - Catch block can handle exception thrown from try block only.
  - A catch block, which can handle any type of exception is called generic catch block / catch-all handler.
  - For each type of exception, we can write specific catch block or we can write single catch block which can handle all types of exception. A catch block which can handle all type of exception is called generic catch block.
- **3. throw**
  - throw is keyword in C++.
  - If we want to generate exception explicitly then we should use throw keyword.
  - "throw statement" is a jump statement.
  - To generate new exception, we should use throw keyword. Throw statement is jump statement.

Note : For thrown exception, if we do not provide matching catch block then C++ runtime gives call to the std::terminate() function which implicitly gives call to the std::abort() function.

Consider the following code

In this code, int type exception is thrown

but matching catch block is not available.

Even generic catch block is also not

available. Hence program will terminate.

Because , if we throw exception from try block then catch block can handle it. But with the help of function we can throw exception from outside of the try block.

```
int main( void )
{
	int num1;
	accept_record(num1);
	int num2;
	accept_record(num2);
	try
	{
		if( num2 == 0 )
			throw 0;
		int result = num1 / num2;
		print_record(result);
	}
	catch(char ex )
	{
		cout<<ex<<endl;
	}
	return 0;
}
```

Template

- If we want to write generic program in C++, then we should use template.
- This feature is mainly designed for implementing generic data structure and algorithm.
- If we want to write generic program, then we should pass data type as a argument. And to catch that type we should define template.
- Using template we can not reduce code size or execution time but we can reduce developers effort.

int num1 = 10, num2 = 20; swap_object<**int>(num1, num2);** string str1="Pune", str2="Karad"; swap_object<string>(str1, str2);	In this code, <int> and <string> is considered as type argument.

template<typename T> //or
template<class T> //T : Type Parameter
void swap(b obj1, **T** obj2 **)**
{
T temp = obj1;
obj1 = obj2;
obj2 = temp;
}
	template and typename is keyword in C++. By passing datatype as argument we can write generic code hence parameterized type is called template

Difference between Procedure Oriented and Object Oriented

Procedure Oriented

- Emphasis on steps or algo
- Programs are divided into small code units called Functions.
- Most function shares global data and can modify it
- Data moves from function to function
- Follows Top-down approach
- Example= C

Object Oriented

- Emphasis on data of program
- Programs are divided into small data units called classes
- Data is hidden and not accessible outside the class
- Objects are communicating with each other
- Follows Bottom-up approach
- Example =C++,JAVA,C#.NET, python

We covered ….

- 1 : Introduction to C++
- 2 : Function features
- 3 : class and object
- 4 : cin - cout
- 5 : static, const, friend
- 6 : memory management
- 7: Operator overloading
- 8 : Object oriented concept
- 9 : composition
- 10 : Inheritance
- 11: virtual function
- 12: Advance C++ feature

Thank You