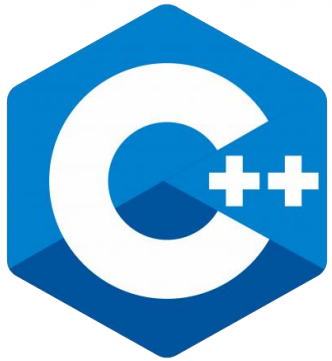

C++ Programming

Trainer : Pradnyaa S Dindorkar

Email: pradnya@sunbeaminfo.com

we did -

1. Struct in C and Cpp
2. Class and object (data member, member functions)
3. Live examples of class and object
4. this pointer
5. Types of Member Functions within class
6. Constructor

Todays Topics

1. Destructor
2. Mutators / setter
3. Inspector / getter
4. facilitator
5. namespace

Destructor

- It is a member function of a class which is used to release the resources.
- It is considered as special function of the class
  - Its name is same as class name and always preceds with tilde operator(~)
  - It doesn't have return type and doesn't take parameter.
  - It is designed to call implicitly.
- Destructor calling sequence is exactly opposite of constructor calling sequence.
- Destructor is designed to call implicitly.
- If we do not define destructor inside class then compiler generates default destructor for the class.
- Default destructor do not release resources allocated by the programmer. If we want to release it then we should define destructor inside class.

Other Member functions of class

- Mutators / setter : modify state of object
- inspector/getter : read the data member but do not change the state of the object
- Facilitator : Provide extra facility to work with object

Scope Resolution Operator (: :)

- :: operator is used to bind a member with some class or namespace.
- It can be used to define members outside class.
- Also used to resolve ambiguity.
- It can also be used to access global members.
  - Example :- ::a =10; access global var.
- Scope resolution Operator is used to :
  - to call global functions
  - to define member functions of class outside the class
  - to access members of namespaces

Namespace

- To prevent name conflicts/ collision / ambiguity in large projects
- to group/ organize functionally equivalent / related types together.
- If we want to access value of global variable then we should use scope resolution operator (::)
- We can not instantiate namespace.
- It is designed to avoid name ambiguity and grouping related types.
- If we want to define namespace then we should use **namespace** keyword.
- We can not define namespace inside function/class.
- We can not define main function inside namespace.
- Namespace can contain:
  1. Variable
  2. Function
  3. Types[structure/union/class]
  4. Enum
  5. Nested Namespace

Note :
•If we define member without namespace then it is considered as member of global namespace.
•If we want to access members of namespace frequently then we should use "**using**" directive.

cin and cout

- C++ provides an easier way for input and output.
- Console Output : Monitor
  - iostream is the standard header file of C++ for using cin and cout.
  - cout is external object of ostream class.
  - cout is member of std namespace and std namespace is declared in iostream header file.
  - cout uses insertion operator(<<)
- Console Input : Keyboard
  - cin is an external object of istream class.
  - cin is a member of std namespace and std namespace is declared in iostream header file.
  - cin uses Extraction operator(>>)
- The output:
  - cout << “Hello C++”;
- The input:
  - cin >> var;

Complex class :- Ex = 5+j7

Data member = real , imaginary

member functions = complex()
complex(int r,int i)
acceptComplexNumber()
printComplexNumber()
~complex()

Thank You