

# C++ Programming

Trainer : Rohan Paramane

Email: [rohan.paramane@sunbeaminfo.com](mailto:rohan.paramane@sunbeaminfo.com)



# Agenda

---

- Default Argument
- Class
- Data Members and Member Functions
- Object
- this Pointer
- Access Specifier



# Default Arguments

- In C++, functions may have arguments with the default values. Passing these arguments while calling a function is optional.
- A default argument is a default value provided for a function parameter/argument.
- If the user does not supply an explicit argument for a parameter with a default argument, the default value will be used.
- If such argument is not passed, then its default value is considered. Otherwise arguments are treated as normal arguments.
- Default arguments should be given in right to left order.
  - ```
int sum (int a, int b, int c=0, int d=0) {  
    return a + b + c + d;  
}
```
  - The above function may be called as
    - Res=sum(10,20);
    - Res=sum(10,20,40);
    - Res=sum(10,30,40,50);



# Class

- Building block that binds together data & code.
- Program is divided into different classes
- Class is collection of data member and member function.
- Class represents set/group of such objects which is having common structure and common behavior.
- Class is logical entity.
- Class has
  - Variables (data members)
  - Functions (member functions or methods)
- By default class members are private( not accessible outside class scope)
- Classes are stand-alone components & can be distributed in form of libraries
- Class is blue-print of an object



# Data Members and Member Functions

## Data Members

- Data members of the class are generally made as private to provide the data security.
- The private members cannot be accessed outside the class.
- So these members are always accessed by the member functions.

## Member Functions

- Member functions are generally declared as public members of class.
- Constructor : Initialize Object
- Destructor : De-initialize Object
- Mutators : Modifies state of the object
- Inspectors : Don't Modify state of object
- Facilitator : Provide facility like IO



# Object

- Object is an instance of class.
- Entity that has physical existence, can store data, send and receive message to communicate with other objects.
- An entity, which get space inside memory is called object.
- Object is used to access data members and member function of the class
- Process of creating object from a class is called instantiation
- **Object has**
  - Data members (**state** of object)
    - Value stored inside object is called state of the object.
    - Value of data member represent state of the object.
  - Member function (**behavior** of object)
    - Set of operation that we perform on object is called behaviour of an object.
    - Member function of class represent behaviour of the object.
    - is how object acts & reacts, when its state is changed & operations are done
    - Operations performed are also known as messages
  - Unique address(**identity** of object)
    - Value of any data member, which is used to identify object uniquely is called its identity.
    - If state of object is same the its address can be considered as its identity.



# Few Points to note

- Member function do not get space inside object.
- If we create object of the class then only data members get space inside object. Hence size of object is depends on size of all the data members declared inside class.
- Data members get space once per object according to the order of data member declaration.
- Structure of the object is depends on data members declared inside class.
- Member function do not get space per object rather it gets space on code segment and all the objects of same class share single copy of it.
- Member function's of the class defines behaviour of the object.



# this pointer

- To process state of the object we should call member function on object. Hence we must define member function inside class.
- If we call member function on object then compiler implicitly pass address of that object as a argument to the function implicitly.
- To store address of object compiler implicitly declare one pointer as a parameter inside member function. Such parameter is called this pointer.
- this is a keyword. "this" pointer is a constant pointer.
- this is used to store address of current object or calling object.
- The invoking object is passed as implicit argument to the function.
- *this* pointer points to current object i.e. object invoking the member function.
- Thus every member function receives *this* pointer.
- Following functions do not get this pointer:
  1. Global Function
  2. Static Member function
  3. Friend Function.





# Access Specifier

- - If we want to control visibility of members of structure/class then we should use access Specifier.
- Defines the accessibility of data member and member functions
- **Access specifiers in C++**
  1. private( - )
  2. protected( # )
  3. public( + )
- 1. Private - Can access inside the same struct/class in which it is declared Generally data members should declared as private. (data security)
- 2. public - Can access inside the same struct/class in which it is declared as well as inside out side function(like main()). Generally member functions should declared as public.



---

# Thank You

