# #22 DAY

## C++ COMPLETE BOOTCAMP

INSPIRE CLUB, MANIT BHOPAL

BRINGS

# C++ Complete Bootcamp

Learn How To Apply Problem Solving Skills

# Hello CPPBudddies

## Day No. 22

---

Welcome

To

# C++ COMPLETE BOOTCAMP

**Your Guide To A Solid Foundataion in C++**

Let us begin

# Structures:

## Way to Store Multiple Data Items

# What is a structure?

**Structure is a collection of variables of different data types under a single name.**

**It is similar to a class in that, both holds a collecion of data of different data types.**

## For example:

**You want to store some information about a person: his/her name, citizenship number and salary.**
**You can easily create different variables name, citNo, salary to store these information separately.**

**However, in the future, you would want to store information about multiple persons.**
**Now, you'd need to create different variables for each information per person: name1, citNo1, salary1, name2, citNo2, salary2**

## How to declare a structure in C++ programming?

The `struct` keyword defines a structure type followed by an identifier (name of the structure).

Then inside the curly braces, you can declare one or more members (declare variables inside curly braces) of that structure. For example:

```
struct Person
{
    char name[50];
    int age;
    float salary;
};
```

Here a structure `person` is defined which has three members: `name`, `age` and `salary`

## How to define a structure variable?

Once you declare a structure `person` as above. You can define a structure variable as:

```
Person bill;
```

Here, a structure variable `bill` is defined which is of type structure P`erson`.

When structure variable is defined, only then the required memory is allocated by the compiler.

## How to access members of a structure?

The members of structure variable is accessed using a **dot (.)** operator.

Suppose, you want to access `age` of structure variable `bill` and assign it 50 to it.
You can perform this task by using following code below:

```
bill.age = 50;
```

# DEMO

# Very Important Topic

# Please Pay Attention

# OOPS !!


Object
Oriented
Programming

The major purpose of C++ programming is to introduce the concept of object orientation to the C programming language.

**Object Oriented Programming** is a **paradigm** that provides many concepts

# Truly Object Oriented Programming Language

The programming paradigm where everything is represented as an object is known as truly object-oriented programming language.

Smalltalk is considered as the first truly object-oriented programming language.

# Why do we use OOP ?

**Object-Oriented Programming**
**is a methodology or paradigm**
**to design a program using**
**classes and objects**

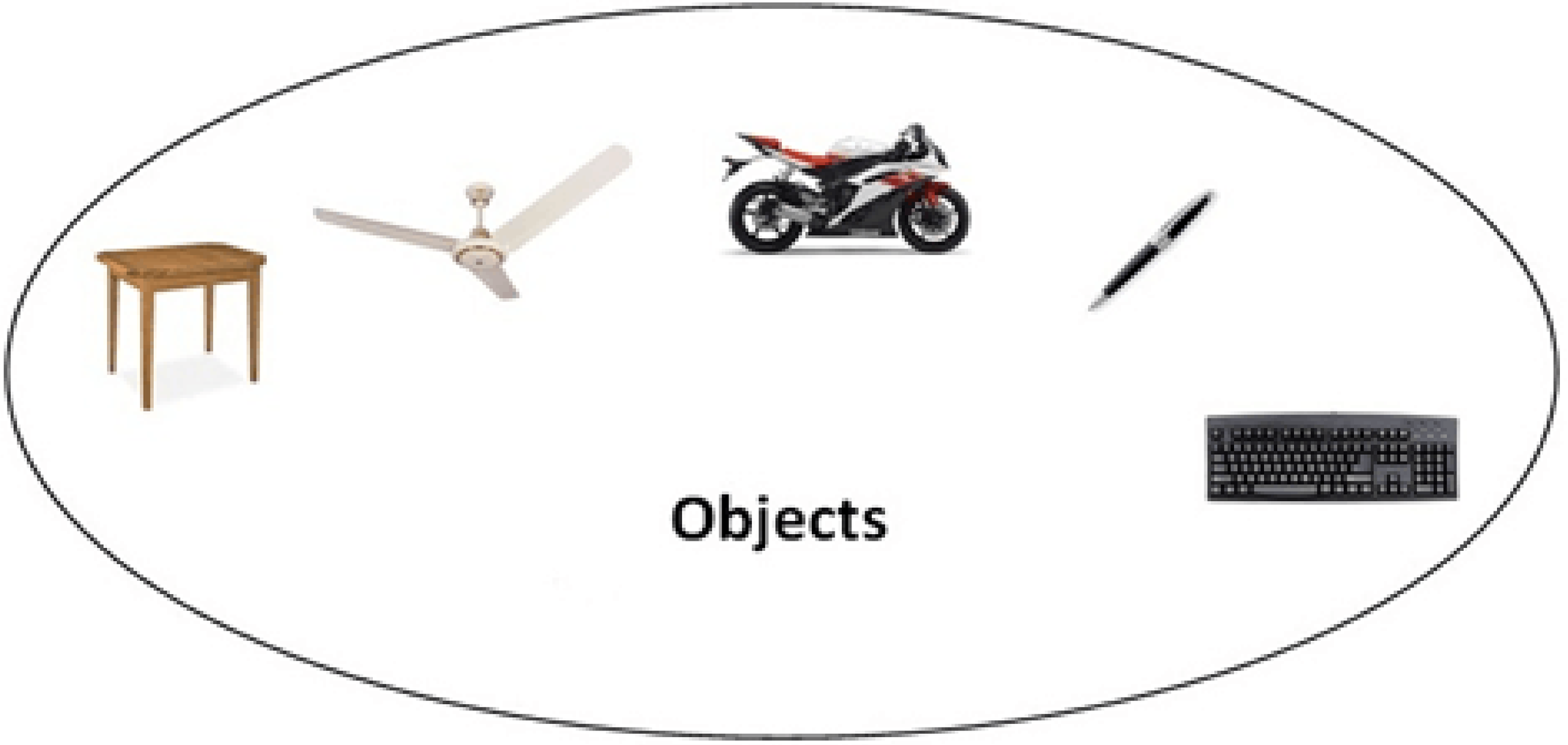
It simplifies
the software development
and
the software maintenance

What is an **Object** ?

**Object means a real word entity such as pen, chair, table etc.**

In Real World,

**Any entity that has**
**state and behavior**
**is known as an object.**
**For example:**
**chair, pen, table, keyboard, bike etc.**
**It can be physical and logical.**


Objects

What is a **CLASS** ?

**Collection of objects**
**is called class.**
**It is a logical entity.**

The building block of C++ that leads to
Object-Oriented programming is a class.

It is a user-defined data type,
which holds its own data members and member functions,
which can be accessed and used by
creating an instance of that class.

A class is like a blueprint for an object

A Class is a user-defined data-type
which has data members and member functions.

Data members are the data variables used to
hold the data values
and
Member functions are the functions used to
manipulate these variables.

And together these
data members and member functions
define the properties and behaviour of
the objects in a Class.

## Object:

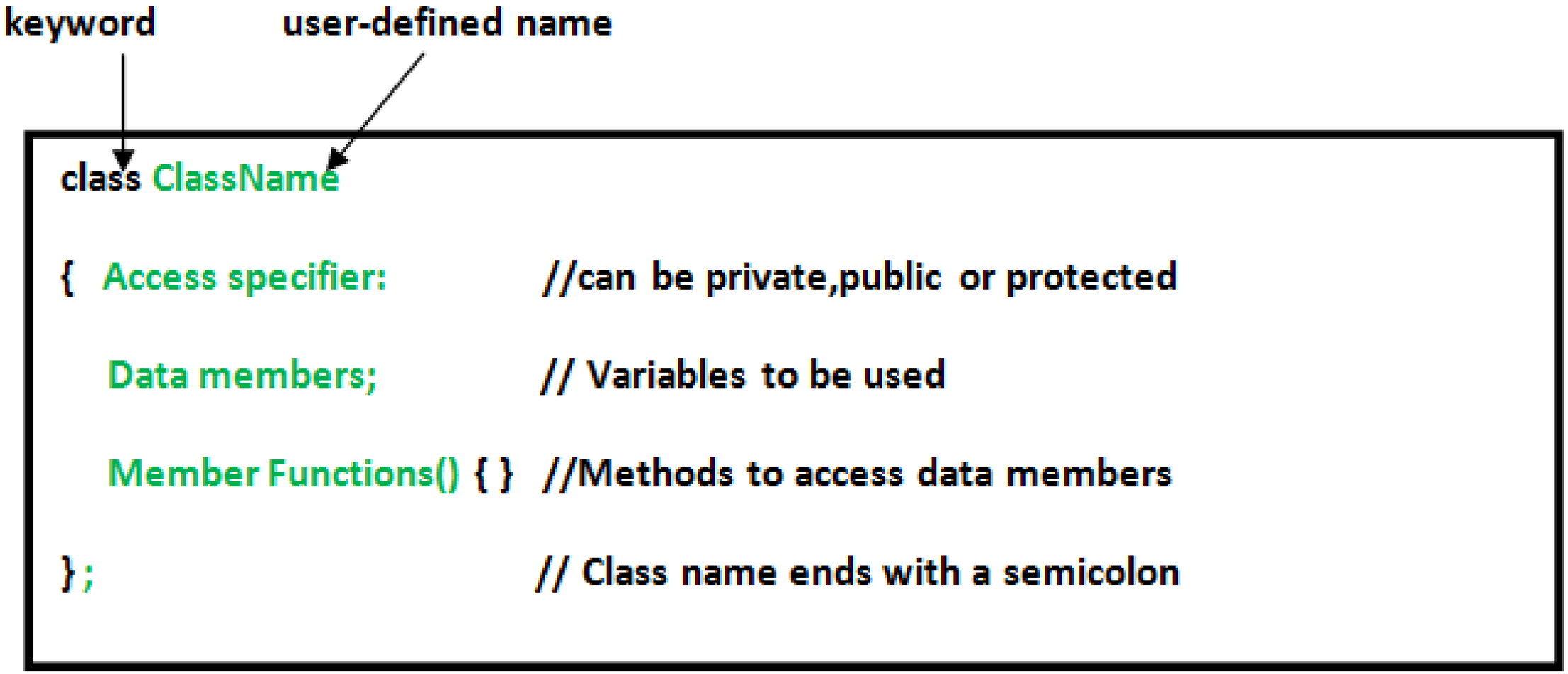
An Object is an identifiable entity with some characteristics and behaviour.

An Object is an instance of a Class.

When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

# DEMO

# Defining Class



# Object-oriented programming has several advantages over procedural programming:

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs
- OOP helps to keep the C++ code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
- OOP makes it possible to create full reusable applications with less code and shorter development time

# Constructors & Destructors

## Constructors

Constructors are special class members which are called by the compiler every time an object of that class is instantiated.

Constructors have the same name as the class and may be defined inside or outside the class definition.

### There are 3 types of constructors:

- Default constructors
- Parameterized constructors
- Copy constructors

# Destructors

**Destructor** is another **special member function** that is called by the **compiler** when the **scope** of the object ends.

# Access Specifiers

Access specifiers define how the members (attributes and methods) of a class can be accessed

## In C++, there are three access specifiers:

- **public** - members are accessible from outside the class
- **private** - members cannot be accessed (or viewed) from outside the class
- **protected** - members cannot be accessed from outside the class, however, they can be accessed in inherited classes

```cpp
class MyClass {
  public:    // Public access specifier
    int x;   // Public attribute
  private:   // Private access specifier
    int y;   // Private attribute
};

int main() {
  MyClass myObj;
  myObj.x = 25;  // Allowed (public)
  myObj.y = 50;  // Not allowed (private)
  return 0;
}
```

# THANK YOU

keep calm,
wear mask,
and
study hard

ISSUED IN PUBLIC INTEREST

whoami


**AKASH MAJI**
**[TCS DIGITAL]**
Your Mentor