# CAP444
# OBJECT ORIENTED PROGRAMMING USING C++


**Created By:**
**Kumar Vishal**
**(SCA), LPU**

# Class

Class is a collection of similar types of objects.

For example: Fruits is class of mango, apple , orange etc.

**Fruits**

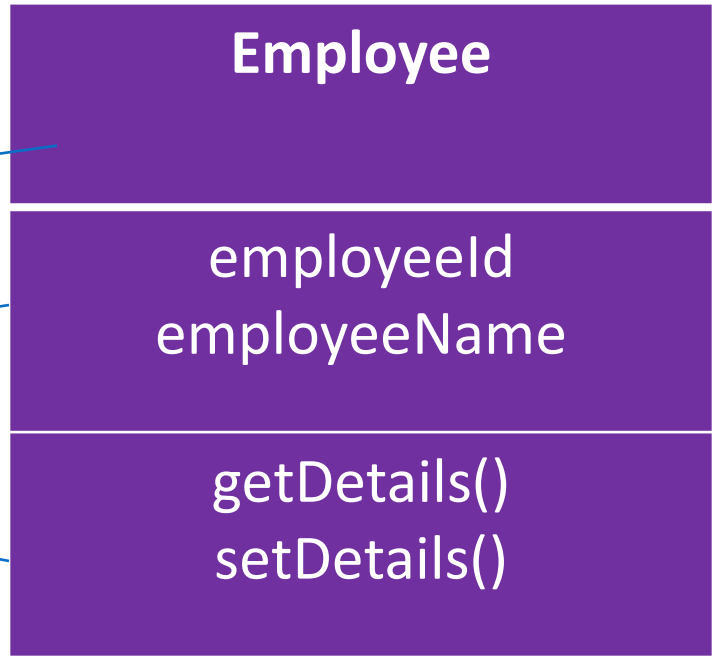

BP-6M
NOKIA
micro SD 16GB
M2
8GB

# Class

Class is a collection of data members and member functions.

Example:

```
class Employee{
//data members
// member functions
}
```



# Object

## Object is an instance of a Class.

```
class Employee
{
        int employeeId;
        char employeeName[20];
        public:
        void getDetails(){}
        void setDetails(){}
};
int main()
{
Employee e1; // e1 is a object
return 0;
}
```

**In this code which option is correct?**

```
class student
{
public:
int regno;
void getStudentDetails();
}
```

A: regno is data member

B. getStudentDetails() is member function

C. above both options are correct

# access specifier

- private
- public
- protected
- By default, all members of a class are private if you don't specify an access specifier.

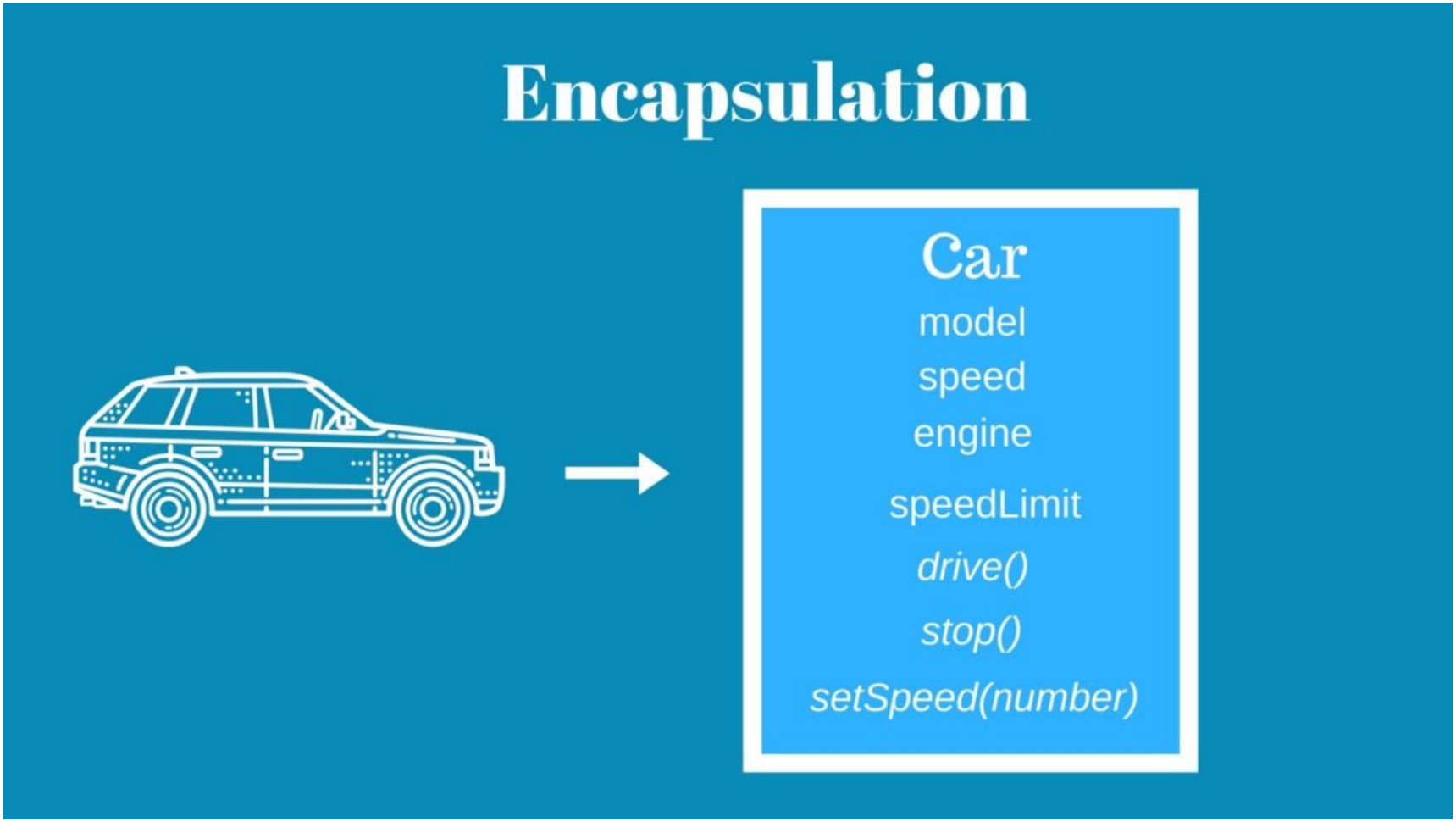
| Specifiers | within same class | in derived class | outside the class |
|---|---|---|---|
| Private | Yes | No | No |
| Protected | Yes | Yes | No |
| Public | Yes | Yes | Yes |

# Select correct option

```
class Employee
{
        int employeeId;
        char employeeName[20];
        public:
        void getDetails(){}
        void setDetails(){}
};
int main()
{
Employee e1;
e1. employeeId=1234
cout<< e1. employeeId;
return 0;
}
```

A. 1234
B. 0
C. Compilation error
D. Run time error

# How to achieve encapsulation features in C++?



How to achieve encapsulation features in C++?

- Define private data members in a class and
- Define public set and get accessors functions which can be used to access these private data members.

Why Encapsulation?

Increased security of data

# Constructor and destructor

# Constructors: types of constructors

- A special method which is used to initialize the object
- It is automatically called when an object of a class is created.
- it has the same name as the class name.
- it is always public
- it does not have any return type

# Types of constructor:

- Default constructor
- Parameterized constructor
- Copy constructor

# Default constructor

A constructor which has no argument is known as default constructor. It is invoked at the time of creating object.

```
class Employee
 {
   public:
        Employee()
        {
            cout<<"Default Constructor"<<endl;
        }
};
```

# Parameterized constructor

A constructor which has parameters is called parameterized constructor.

```
class Employee
 {
    int empId;
    string empName;
   public:
        Employee(int id, string name)
        {
            empId=id;
            empName=name;
        }
};
```

# Copy Constructor

Copy Constructor is a type of constructor which is used to create a copy of an existing object of a class.

Syntax:

class-_name**(**class_name **& object_name)**

**{**

**}**

**To call this:** class_name obj1(arguments);

class_name obj2 = obj1;

# this keyword

- this is a keyword that refers to the current instance of the class
- It is used to pass current object as a parameter to another method.

# Destructor

- Destructor is a special member function which destructs or deletes an object.
- A destructor is called automatically when object goes out of scope.
- Destructors have same name as the class preceded by a tilde (~)
- There can only one destructor in a class
- When a class contains a pointer to memory allocated in class, we should write a destructor to release memory

Which option is correct for defining the destructor?

Option1:

```
~mobile()
      {
 cout<<"destructor called"<<endl;
      }
```

Option2:

```
 ~mobile( string str)
      {
 cout<<"destructor called"<<endl;
      }
```

A. Option1 is correct
B. Option2 is correct
C. Both option is correct

## Any Query?