
# Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

## Session: Constructor and Destructor Functions

# Quick Recap of Relevant Topics

IIT Bombay

- Structures and classes
- Data members and member functions
- Accessing members
- Access control of members
  - public and private members

# Overview of This Lecture

IIT Bombay

- Special member functions
  - Constructor functions
  - Destructor functions

# Acknowledgment


- Much of this lecture is motivated by the treatment in

  **An Introduction to Programming Through C++**

  **by Abhiram G. Ranade**

  **McGraw Hill Education 2014**

# Recap: Member Functions and Their Usage

IIT Bombay

- A class can have public or private member functions

```
class V3 {
  // 3-dimensional vector with printLength()
  private: double x, y, z;
  public:
    … Other member functions …
    void printLength() {
       cout << length() << endl; return;
    }
  private:
    double length() {return sqrt(x*x + y*y + z*z);}
};
```

```
int main() {
  V3 a, * ptr;
  … Some code here …
  a.printLength();
  ptr = new V3;
  if (ptr == NULL) return -1;
  … Some code here …
  ptr->printLength();
  delete ptr;
  return 0;
}
```

# Two Special Member Functions of Every Class

IIT Bombay

- **Constructor:** Invoked **automatically** when an object of the class is allocated
  - Convenient way to initialize data members
  - Just like any other member function
    - Accepts optional input parameters
    - Can be used to perform tasks other than initialization too
- **Destructor:** Invoked **automatically** when an object of the class is de-allocated
  - Convenient way to do book-keeping/cleaning-up before de-allocating object
  - Accepts no parameters
  - Can be used to perform other tasks before de-allocating object

# Example Constructor of Class V3

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
       x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
      … Other member functions of V3 …
};
```

**Constructor of class V3**

# Example Constructor of Class V3

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
       x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
     … Other member functions of V3 …
};
```

**Constructor of class V3**

- A member function
- No return type
- Same name as that of class (i.e. V3)
- Optional input parameters
- Mostly used for initialization

# Example Constructor of Class V3

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
      x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
      … Other member functions of V3 …
};
```

**Another constructor of class V3**

**Typed list of parameters different from that of the previous constructor**

# Multiple Constructors of Same Class

IIT Bombay

- A class can have multiple constructors as long as each one has a distinct list of parameter types
  - V3 (double vx, double vy, double vz) and V3()
- When allocating an object of the class, the types of parameters passed to the constructor determine which constructor is invoked
  - V3 myObj1;  V3 *myObj2 = new V3(1.0, 2.0. 3.0);
- Allocated object serves as the receiver object for the constructor call

# Usage of Constructors

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
      x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
   … Other member functions of V3 …
};
```

Note the "public" declaration

```
int main() {
  V3 a (0.0, 0.0, 0.0);
  V3 b;
  V3 *p, *q;
   … Some code here …
  p = new V3 (1.0, 2.0, 3.0);
  q = new V3;
   … Some code here …
  delete p; delete q;
  return 0;
}
```

# Usage of Constructors

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
      x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
    … Other member functions of V3 …
};
```

**Note the "public" declaration**

```
int main() {
  V3 a (0.0, 0.0, 0.0);
  V3 b;
  V3 *p, *q;
   … Some code here …
  p = new V3 (1.0, 2.0, 3.0);
  q = new V3;
   … Some code here …
  delete p; delete q;
  return 0;
}
```

# Usage of Constructors

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
      x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
   … Other member functions of V3 …
};
```

```
int main() {
  V3 a (0.0, 0.0, 0.0);
  V3 b;
  V3 *p, *q;
   … Some code here …
  p = new V3 (1.0, 2.0, 3.0);
  q = new V3;
   … Some code here …
  delete p; delete q;
  return 0;
}
```

# Usage of Constructors

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
  public:
    V3 (double vx, double vy, double vz) {
       x = vx; y = vy; z = vz;  return;
    }
    V3 () { x = y = z = 0.0;  return; }
    … Other member functions of V3 …
};
```

```
int main() {
  V3 a (0.0, 0.0, 0.0);
  V3 b;
  V3 *p, *q;
   … Some code here …
  p = new V3 (1.0, 2.0, 3.0);
  q = new V3;
   … Some code here …
  delete p; delete q;
  return 0;
}
```

# Two Special Member Functions of Every Class


- **Constructor:** Invoked **automatically** when an object of the class is allocated
  - Convenient way to initialize data members
  - Just like any other member function
    - Accepts optional input parameters
    - Can be used to perform tasks other than initialization
- **Destructor:** Invoked **automatically** when an object of the class is de-allocated
  - Convenient way to do book-keeping/cleaning-up before de-allocating object
  - Accepts no parameters
  - Can be used to perform other tasks before de-allocating object

# Example Destructor of Class V3

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
    double length() { … }
  public:
    … Constructors of class V3 …
    ~V3() { if (length() == 0.0)
                {cout << "Zero vector!!! " << endl;}
              return;
      }
    … Other member functions of class V3 …
};
```

**Destructor of class V3**

- A member function
- No return type
- Name: ~ followed by name of class (i.e. ~V3)
- No input parameters
- Mostly used for book-keeping/clean-up before de-allocation of objects

**Multiple destructors of same class not allowed in C++**

# Example Destructor of Class V3

IIT Bombay

```
class V3 {
  private:
    double x, y, z;
    double length() { ... }
  public:
    ... Constructors of class V3 ...
    ~V3() { if (length() == 0.0)
              {cout << "Zero vector!!! " << endl;}
            return;
      }
    ... Other member functions of class V3 ...
};
```

**Note the "public" declaration**

```
int main() {
  V3 a (1.0, 2.0, 3.0);
  { V3 b;
    a = b;
  }
  V3 *p =
       new V3(1.0, 1.0, 1.0);
  a = *p;
  delete p;
  return 0;
}
```

# Summary


- Constructor and destructor functions of classes
- Simple usage of above special member functions
  - More complex usage coming later …