MA122 - Computer Programming and Applications

Indian Institute of Space Science and Technology

March 29, 2017

Lecture 24

initialization


```
1  int year = 2001;      // valid initialization
2  struct thing
3  {
4    char * pn;
5    int m;
6  };
7  thing amabob = {"wodget", -23}; // valid
       initialization
8
9  Stock hot = {"Sukies  Autos, Inc.", 200, 50.25};
10 // NO! compile error
```

default arguments


```
int harpo(int n, int m = 4, int j = 5);          // VALID
int chico(int n, int m = 6, int j);              // INVALID
int groucho(int k = 1, int m = 2, int n = 3);    // VALID
```

For example, the `harpo()` prototype permits calls with one, two, or three arguments:

```
beeps = harpo(2);        // same as harpo(2,4,5)
beeps = harpo(1,8);      // same as harpo(1,8,5)
beeps = harpo (8,7,6);   // no default arguments used
```

Declaring and Defining Constructors


```
1  // constructor prototype with some default arguments
2  Stock(const string & co, long n = 0, double pr = 0.0);
3
4  // constructor definition
5  Stock::Stock(const string & co, long n, double pr)
6  {
7    company = co;
8    if (n < 0)
9    {
10     std::cerr << "Number of shares cant be negative; "
11     << company << " shares set to 0.\n";
12     shares = 0;
13   }
14   else
15     shares = n;
16   share_val = pr;
17   set_tot();
18 }
```

example


```
1 // NO!
2 Stock::Stock(const string & company, long shares,
      double share_val)
3 {
4   ...
5 }
```

Constructor arguments do not represent the class members; they represent values that are assigned to the class members, thus they must have different names.

Using constructors


```
1 Stock food = Stock("World Cabbage", 250, 1.25);
2
3
4
5 Stock garment("Furry Mason", 50, 2.5)
```

C++ uses a class constructor whenever you create an object of that class.

```
1 Stock *pstock = new Stock("Electroshock Games", 18,
      19.0);
```

using constructors

Constructors are used differently from the other class methods. Normally, you use an object to invoke a method:

```
1 stock1.show(); // stock1 object invokes show() method
```

However, you cant use an object to invoke a constructor because until the constructor finishes its work of making the object, there is no object.

Default Constructor

A default constructor is a constructor that is used to create an object when you dont provide explicit initialization values.

```
1 Stock fluffy_the_cat; // uses the default constructor
2
3 //For the Stock class, the default constructor look
      like this:
4
5 Stock::Stock() { }
```

The compiler provides default constructor only if you dont define any constructors.

using constructor

If you wish to create objects without explicit initialization, you must define your own default constructor. Two ways:

```
1  //provide default values to the existing constructor
2
3  Stock(const string & co = "Error", int n = 0, double
       pr = 0.0);
4
5  //another method, using function overloading
6
7  Stock();
8  //Defenition
9  Stock::Stock()      // default constructor
10 {
11   company = "no name";
12   shares = 0;
13   share_val = 0.0;
14   total_val = 0.0;
15 }
```

using constructor


```
1  Stock first;  // calls default constructor implicitly
2
3  Stock first = Stock();   // calls it explicitly
4
5  Stock *prelief = new Stock; // calls it implicitly
6
7
8  Stock first("Concrete Conglomerate"); // calls
       constructor
9
10 Stock second(); // a function which returns Stock (
       prototype)
11
12 Stock third;          // calls default constructor
```

Lecture 24

Destructor


```
1  ~Stock(); //prototype, destructor must have no
       arguments
2
3  Stock::~Stock() //do-nothing function
4  {
5  }
6
7
8  Stock::~Stock()  // class destructor
9  {
10   cout << "Bye, " << company << "!\n";
11 }
```

Lecture 24

Destructor


```
1  class Stock
2    {
3    private:
4      std::string company;
5      long shares;
6      double share_val;
7      double total_val;
8      void set_tot() { total_val = shares * share_val; }
9    public:
10     Stock();        // default constructor
11     Stock(const std::string & co, long n = 0, double
           pr = 0.0);
12     ~Stock();
13     void buy(long num, double price);
14     void sell(long num, double price);
15     void update(double price);
16     void show();
17   };
```

Stock Class


```
1  Stock::Stock()      // default constructor
2  {
3    company = "no name";
4    shares = 0;
5    share_val = 0.0;
6    total_val = 0.0;
7  }
8  Stock::Stock(const std::string & co, long n, double pr
       )
9  {
10   shares = n;
11   share_val = pr;
12   set_tot();
13 }
14
15 Stock::~Stock() {}
```

```
1  int main(){
2
3  Stock stock1("NanoSmart", 12, 20.0);
4
5  stock1.show();
6
7  Stock stock2 = Stock ("Boffo Objects", 2, 2.0);
8
9  stock2.show();
10
11 cout << "Assigning stock1 to stock2:\n";
12
13 stock2 = stock1;
```

Destructor


```
cout << "Listing stock1 and stock2:\n";

stock1.show();

stock2.show();

cout << "Using a constructor to reset an object\n";

stock1 = Stock("Nifty Foods", 10, 50.0);

cout << "Revised stock1:\n";

stock1.show();

cout << "Done\n";

return 0;
}
```