Date:.....

ITEM 12 Copy All Parts of an Object.

We can declare our own copy functions. i.e. copy constructor & copy assignment operator.

eg:

```
class Customer
{ public:
    Customer (const Customer &rhs);
    Customer &operator = (const Customer &rhs);
  private:
    String name;
};

Customer::Customer (const Customer & rhs) :
    : name(rhs.name)
{   log << "Copy Const";
}

Customer :: Customer & operator = (const Customer &rhs)
{
    name = rhs.name;
    return * this;
}
```

Above program is perfectly fine.

Date:

what if we add another data member to customer.

class Date {...};

Class Customer

{ ⁝

private:

String name;

Date last_transaction;

}

last_transaction will not be copied ~~not~~ neither compiler will display any error for this.

∴ Make sure to update the copying functions, if a new data member is added to class.

Date:

This issue can also arise through inheritance.

If we frame a derived class & we do not mention the base class parts in the copying function of derived class, the default constructor of base class is called & none of the base class members is initialized with the copy func^n rather they are initialized with default value such as zero.

∴ We must copy the base class parts while writing copy func^n of derived class.

```
Priority Cust :: Priority Cust (const Priority Cust (&rhs)
{ : Customer (rhs),
    priority (rhs.priority)
  { Log << "Cons";
  }
```

Date:.....

PriorityCust & PriorityCust::operator=(const
PriorityCust & rhs)
{ Customer::operator=(rhs);
priority = rhs.priority;
return *this;
}

Since both copy constructor & copy assignment operator may look same & you want to avoid code duplication then create a third member function that both call.

Dont call copying functions inside each other.