Chapter 7: Classes (A First Look)

1. Defining Abstract Data Types
2. Access Control and Encapsulation

Built-in types

int

int i;

User-defined types

string

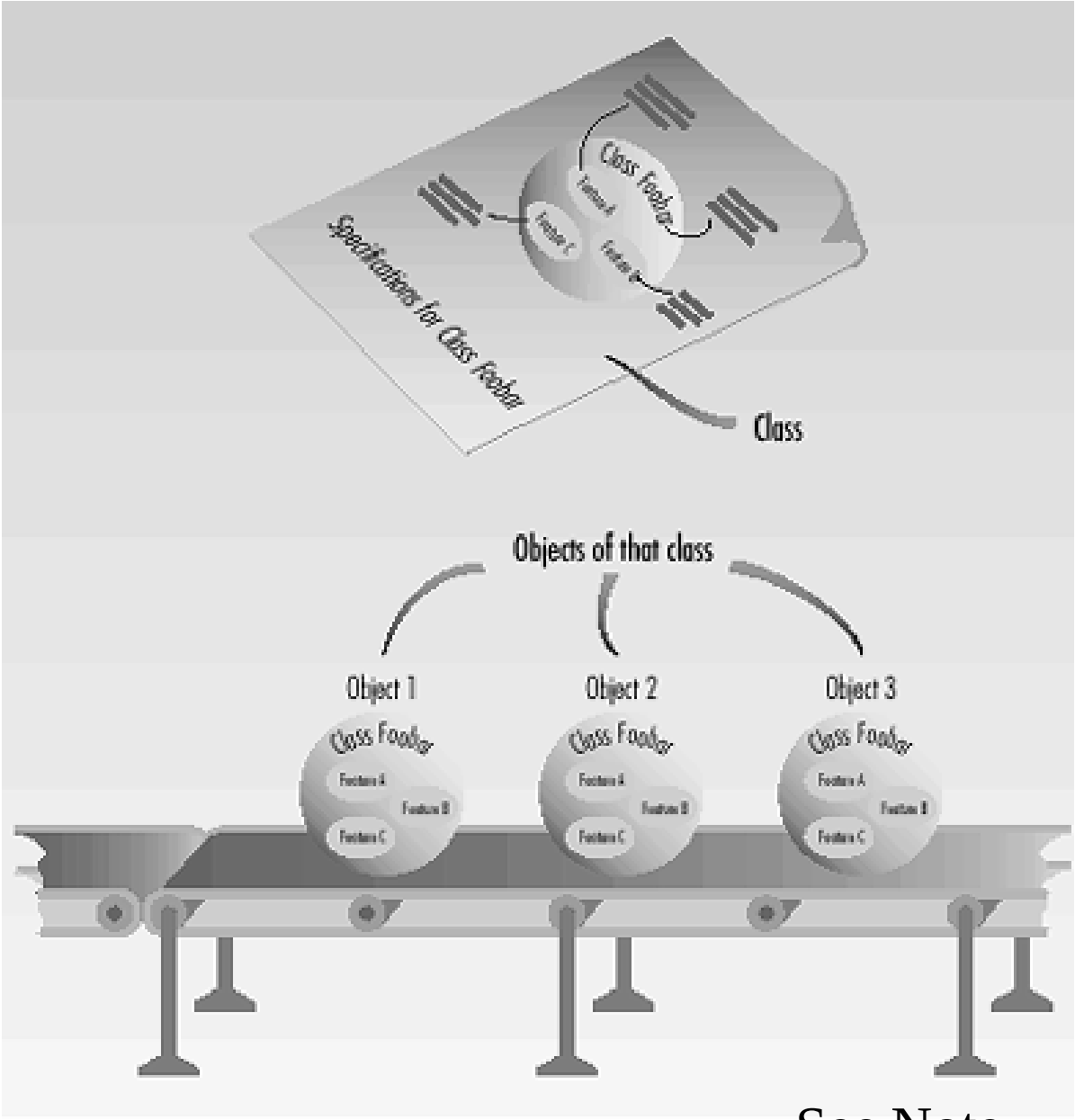
string s;

Sales_item

Sales_item s1;

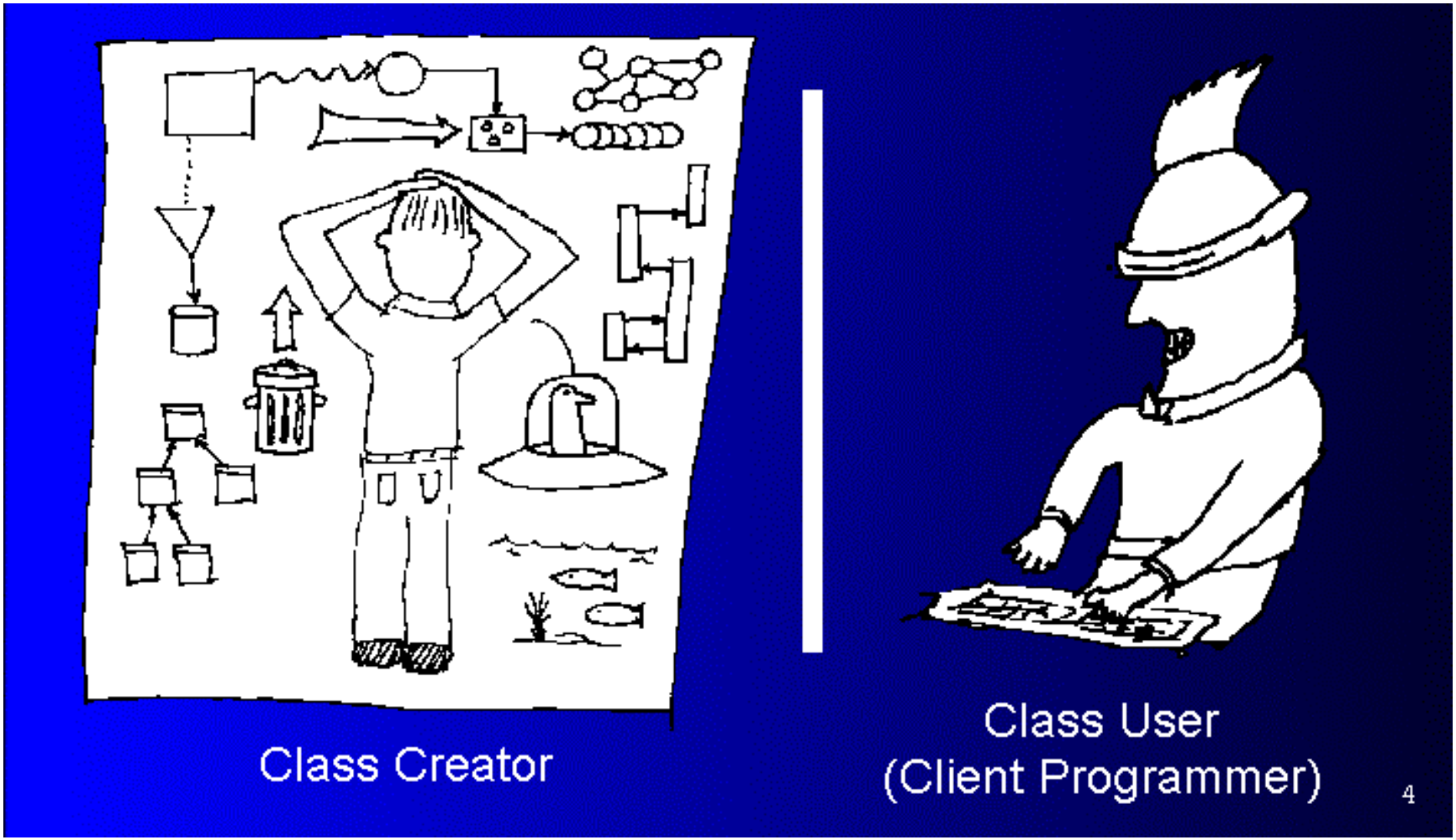
Sales_data

Sales_data s2;



See Note

Different Kinds of Programming Role in C++

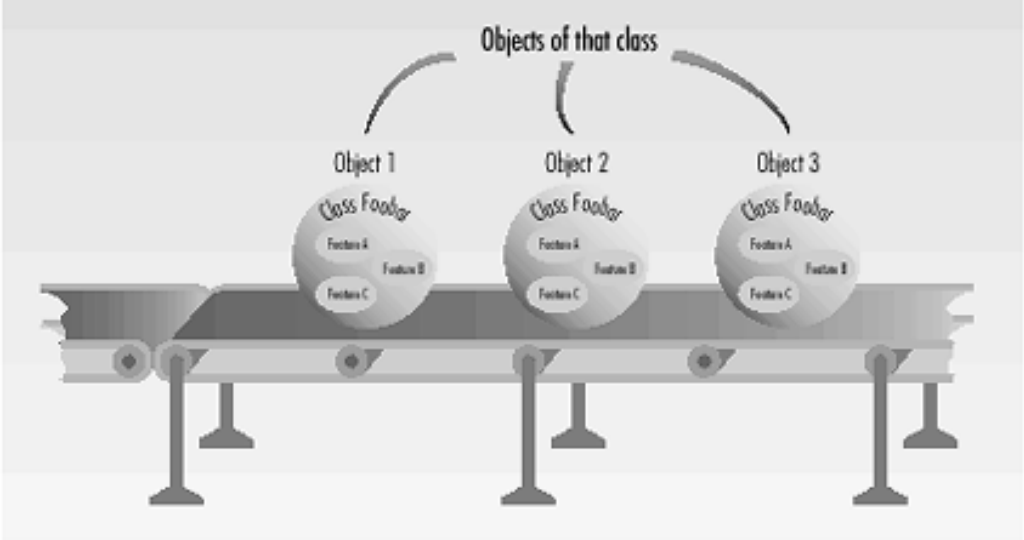


See Note

Constructor

- Class data members are initialized through a **constructor**.
- Constructor is a special member function with the same name as its class.
  - Unlike other member functions, constructors have no return type.
  - Like other member functions they take a (possibly empty) parameter list and have a function body.
  - A class can have multiple constructors. Each constructor must differ from the others via parameters.

- The constructor's parameters specify the initializers that may be used when creating objects of the class type.
- These initializers are used to initialize the data members of the newly created object. Constructors usually should ensure that every data member is initialized.



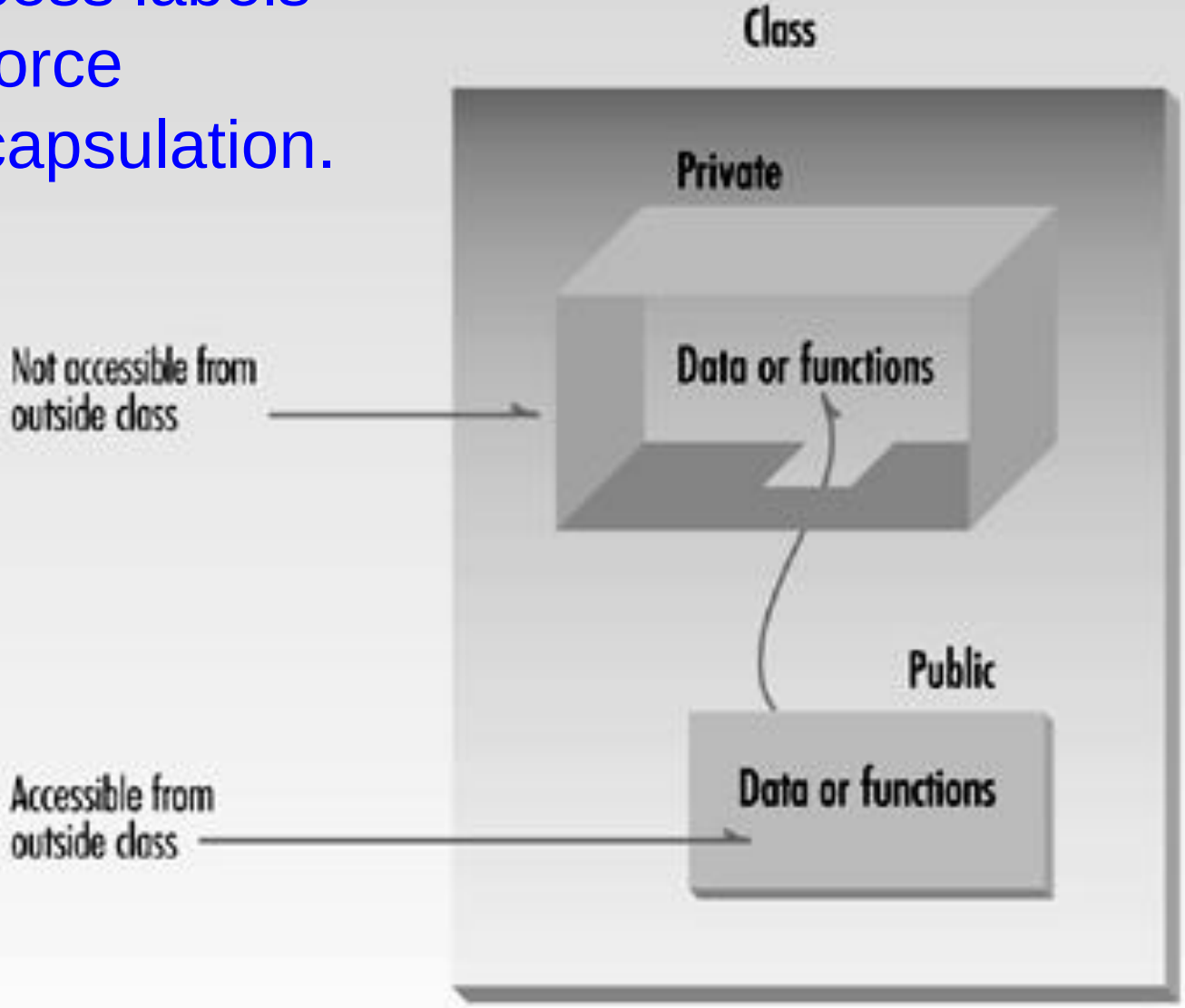

```
string s("hello");  // constructor: takes string literal
string s;  // default constructor: empty string
```

- The default constructor is the one that takes no arguments. The default constructor says what happens when we define an object but do not supply an (explicit) initializer:

```
vector<int> vi;  // default constructor: empty vector
string s;  // default constructor: empty string
Sales_data total;   // default constructor: initialize ???
```

See Note

Access labels enforce encapsulation.



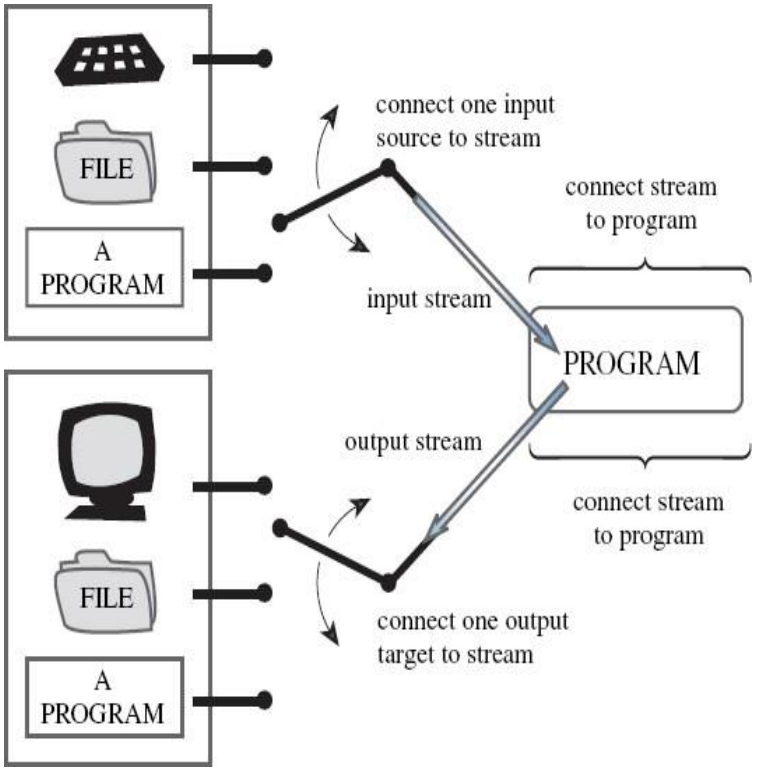
- See note

Chapter 8: IO Library

IO: Basic Concept

- **streams** - I/O uses the concept of *streams* - a flow of characters. *streams* may flow into or out of files or console. They can also flow into and out of strings. C++ tries to offer the same set of commands whatever the nature of the source and destination.

- **state** - Each stream has state information indicating whether an error has occurred, etc.
- **buffer** – Default is ok.
- **locale** - Default is ok.



IO Library

Table 8.1. IO Library Types and Headers

Header	Type
`iostream`	`istream` reads from a stream
	`ostream` writes to a stream
	`iostream` reads and writes a stream; derived from `istream` and `ostream`,
`fstream`	`ifstream`, reads from a file; derived from `istream`
	`ofstream` writes to a file; derived from `ostream`
	`fstream`, reads and writes a file; derived from `iostream`
`sstream`	`istringstream` reads from a `string`; derived from `istream`
	`ostringstream` writes to a `string`; derived from `ostream`
	`stringstream` reads and writes a `string`; derived from `iostream`

String Stream

- **In-memory input/output**: a stream is attached to a string within the program's memory.
- That string can be written to and read from using the iostream input and output operators.

Header	Type
`sstream`	`istringstream` reads from a `string`; derived from `istream`
	`ostringstream` writes to a `string`; derived from `ostream`
	`stringstream` reads and writes a `string`; derived from `iostream`

Sample usage, see note

Until Next Time

- Lab4
- HW4
- [Reading] Chapter 9 (Sequential Containers).