Introduction to C++

- Programming Concept
- Basic C++
- C++ Extension from C

1

What programming is?

Programming is taking

A ***problem***

Find the area of a rectangle

A set of ***data***

length

width

A set of ***functions***

area = length * width

Then

Applying functions to data to get answer

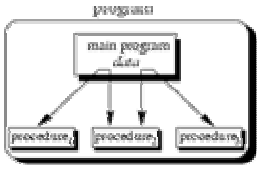
2

Programming Concept Evolution

- Unstructured
- Procedural
- Object-Oriented

3

Procedural Concept



- The main program coordinates calls to procedures and hands over appropriate data as parameters.

4

Procedural Concept (II)

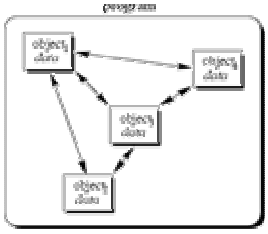
- Familiar languages, as C, Pascal, Basic, Fortran, Assembly, all encapsulate functions and data in procedures
- In C, we also encapsulate procedures within procedures. We start with *main()* or a top level routine. We then write subroutines that are called from *main()*. We introduce data into procedures using parameters or arguments and get information back using shared variables or return values.
- For the rectangle problem, mentally we form the model of what needs to be done => we develop a procedure

```
int compute_area (int l, int w)
{
      return ( l * w );
}
```

- All such languages are called *procedural languages.*

5

Object-Oriented Concept



- Objects of the program interact by sending messages to each other

6

Objects

An object is an encapsulation of both functions and data (not one or the other individually)

- ***Objects are an Abstraction***
  - *represent real world entities*
  - *Classes are data type that define shared common properties or attributes*
  - *Objects are instances of a class*
- ***Objects have State***
  - *have a value and a particular time*
- ***Objects have Operations***
  - *associated set of operations called methods that describe how to carry out operations*
- ***Objects have Messages***
  - *request an object to carry out one of its operations by sending it a message*
  - *messages are the means by which we exchange data between objects*

7

OO Perspective

Let's look at our earlier Rectangle through object oriented eyes:

Object

Rectangle

data - *encapsulated*

width

length

function (called a method)- *encapsulated*

*area = length * width*

In our object oriented program, we will have an instance of the class Rectangle.

If we wish to find the area of the rectangle, we send a request to the object instance telling the rectangle to return its area.

In C++, rather than writing a procedure, we define a class that encapsulates the knowledge necessary to answer the question - here, what is the area of the rectangle.

8

Example Object Oriented Code

```
class Rectangle
{
   private:
      int width, length;
   public:
      Rectangle(int w, int l)
         {
            width = w;
            length = l;
         }
      int area()
      {
          return width*length;
      }
}
```

```
main()
{
   Rectangle rect(3,5);
   cout<<rect.area()<<endl;
}
```

9

Object-Oriented Programming Languages

- Characteristics of OOPL:
  - Encapsulation
  - Inheritance
  - Polymorphism
- OOPLs support :
  - -modular programming
  - -ease of development
  - -maintainability

10

Characteristics of OOPL

- **Encapsulation**=combining data structure with actions

  -actions -> permissible behaviors of objects that are controlled through the member functions

  -data structure -> represents the properties, the state, or characteristics of objects

  -information hiding = process of making certain items inaccessible
- **Inheritance**=ability to derive new objects from old

  -permits objects of a more specific class to inherit the properties (data) and behaviors (functions) of a more general class

  -ability to define a hierarchical relationship between objects
- **Polymorphism**=how objects respond to certain kinds of messages

  -ability for different objects to interpret functions differently

11

Basic C++

- Inherit all ANSI C directives
- Inherit all C functions
- You don't have to write OOP programming in C++

12

Basic C++ Extension from C

- comments

```
/* You can still use the old comment style, */
/* but you must be // very careful about mixing them */
// It's best to use this style for 1 line or partial lines
/* And use this style when your comment
consists of multiple lines */
```

- cin and cout (and #include <iostream.h>)

```
cout << "hey";
char name[10];
cin >> name;
cout << "Hey " << name << ", nice name." << endl;

cout << endl;     // print a blank line
```

- declaring variables almost anywhere

```
// declare a variable when you need it
  for (int k = 1; k < 5; k++)
   {
       cout << k;
   }
```

13

Basic C++ Extension from C (II)

- const

  -in C=> replacement for the #define statement, but #define statements are handled by the preprocessor - no type checking.

  -the const specifier is interpreted by the compiler =>type checking is applied

  -a const is declared and initialized in a single place

- New data type
  - Reference data type "&". Much likes pointer

```
int ix; /* ix is "real" variable */
int & rx = ix; /* rx is "alias" for ix */
ix = 1; /* also rx == 1 */
rx = 2; /* also ix == 2 */
```

14

C++ - Advance Extension

- C++ allow function overloading

  -in C++, functions can use the same names, within the same scope, if each can be distinguished by its name *and* signature

  -the signature specifies the number, type, and order of the parameters expressed as a comma separated list of arg types

  -the name plus signature, then, uniquely identify a function

15

Take Home Message

- There are many different kinds of programming paradigms, OOP is one among them.
- In OOP, programmers see the execution of the program as a collection of dialoging objects.
- The main characteristics of OOPL include encapsulation, inheritance, and polymorphism.

16