C++ Object Oriented Programming

EE 213A
Spring 2000

Differences between C and C++

• Use of struct in C
• Lack of functions inside a struct
• Concept of OOP in C++
• Ability to have an interface for users in C++
Motivation for Objects

• Treat things as objects in real life
• In real life, interact with INTERFACE
• In C++, also interact with INTERFACE
• In real life, “inner” parts of objects are kept away from user
• In C++, some data are kept away from user access

Object Examples

• A car
  – Lots of parts inside (“inner” parts), but you care only about the gas pedal, brake, radio
  – You don’t care about how these things work
• A bank account
  – Lots of data, account number, balance, interests. You can’t access them
  – You give the checks to the teller
C Structure Recall

- struct in C
  struct name{
    int balance;
    ..........
  };
- Note: You can’t have functions associated with a struct!

Class Structure in C++

class name {
    public:
    // all the interface functions user can access
    private:
    // all the data that you do not want user to
    // have access. Outside world cannot access
    // any data here
};
Class Example

• Want to have a class for BankAccount
• Data: accountno, amount, interestrate
• Allow open an account with or without a balance
• Functions available: deposit, calculate interest, display info, compare two accounts

Constructor

• A function that is called when an object of that class is declared
• Inside public:
  BankAccount(); // default constructor
  BankAccount(int amount);
  BankAccount(int accountno, int amount);
Constructor (Cont’d)

- Outside the class, implement function
  BankAccount::BankAccount();
  BankAccount::BackAccount(…….);
- You can have more than one constructor for the same class!
- Function overloading

Data Members

- Declare data fields as in struct
- If put under public, user can have access to them directly
- If put under private, user cannot access them without any public functions
- For example accountno, amount, and interestrate should be under private
Member Functions

- Similar to those in C or C++
- Inside class (under public or private):
  void info(int accountno);
- Outside class, implement function:
  void BankAccount::info(int account) {
      .......
      .......
  }

Overload Operators

- In C, string c1 and c2, can you do…
  if (c1 == c2) then
- No!!! “==” is not defined for strings!
- In C++, you can overload any operators you want so you can make your life easier
- Especially useful for classes, since no operators are available for them.
Overloading Operators (Cont’d)

• Example: overload “==” for BankAccount
• Under public:
  friend int operator ==(BankAccount& a, BankAccount& b);
• int operator ==(BankAccount& a, BankAccount& b){
      ........;
      ........;
      if (same) return 1;
      else return 0;
  }