Type Conversion

- Already can create user-defined types
- Already have default convert capabilities among built-in types.
- Need the ability to convert:
  - built-in type to user-defined type
  - user-defined type to built-in type
  - user-defined type to user-defined type

Built-in to User-defined

The conversion of a built-in type to a user-defined type can be accomplished by the use of an appropriate constructor for the targeted user-defined type.

This makes the conversion as simple as an explicit cast of one built-in type to another built-in type.

A Date Class

```cpp
class Date {
private:
    int Month, Day, Year;
public:
    Date();
    Date(int M, int D, int Y);
    Date(int yyyymmdd); // conversion constructor
    void ShowDate(); // display function
};
```

A Date Class Implementation

```cpp
Date::Date() {
    Month = 7; Day = 4; Year = 2001;
}
Date::Date(int M, int D, int Y) {
    Month = M; Day = D; Year = Y;
}
void Date::ShowDate() {
    cout << setfill('0')
        << setw(2) << Month << '/'
        << setw(2) << Day << '/'
        << setw(2) << Year;
}
```

Using the conversion

```cpp
void main() {
    Date a;
    cout << "Date a is:" << endl;
    a.ShowDate();
    cout << endl;
    a = Date(20020101);
    cout << "Date a is now: " << endl;
    a.ShowDate();
    cout << endl << endl;
}
```

```
Date a is:
07/04/2001
Date a is now:
01/01/2002
```
User-defined to Built-in

The conversion of a user-defined type to a built-in type can be accomplished by the use of an appropriate conversion operator function as a member of the user-defined type.

This also makes the conversion as simple as an explicit cast of one built-in type to another built-in type.

Date to int conversion

Date::operator int() {
    int yyyymmdd;
    yyyymmdd = Year * 10000 + Month * 100 + Day;
    return yyyymmdd;
}

Using the conversion

void main() {
    Date a(4, 1, 1999);
    int b;
    b = (Date) a;
    cout << "a's date is: ";
    a.ShowDate();
    cout << endl
    << "This date, as an int, is: "
    << b << endl;
}

User-defined to User-defined

The conversion of a user-defined type to a user-defined type is also accomplished by the use of a member conversion operator function.

This makes the conversion as simple as an explicit cast of one built-in type to another built-in type.

Add an IntDate Class

// Dates.h
class IntDate;  // forward declaration

class Date {
private:
    int Month, Day, Year;
public:
    Date();
    Date(int M, int D, int Y);
    operator IntDate();
    void ShowDate();
};

// continues . . .
Add an IntDate Class

```cpp
// ... class IntDate {
private:
  int yyyymmdd;
public:
  IntDate(int ymd = 0);
  operator Date(); // conversion operator
  void ShowIntDate();
};
```

Date to IntDate conversion

```cpp
Date::operator IntDate() { int Temp; Temp = 10000 * Year + 100*Month + Day; return IntDate(Temp); }
```

IntDate Class Implementation

```cpp
IntDate::IntDate(int ymd) {
  yyyymmdd = ymd;
}
void IntDate::ShowIntDate() {
  cout << yyyymmdd;
}
```

IntDate to Date conversion

```cpp
IntDate::operator Date() { int M, D, Y;
  Y = yyyymmdd / 10000;
  M = (yyyymmdd - Y*10000) / 100;
  D = yyyymmdd - Y*10000 - M*100;
  return Date(M, D, Y);
}
```

Using the conversion

```cpp
void main() {
  Date a(4, 1, 1999), b;
  IntDate c(20011215), d;
  b = Date(c);
  d = IntDate(a);
  cout << "a's date is: ";
  a.ShowDate();
  cout << endl << "as an IntDate object this date is: ";
  d.ShowIntDate();
  // continues ...
```
Using the conversion

// . . . continued

cout << endl << "c's date is: ";
c.ShowIntDate();

cout << endl << "as a Date object this date is: ";
b.ShowDate();
cout << endl << endl;

Output

Usage of the conversion

a's date is: 04/01/1999
as an IntDate object this date is: 19990401

c's date is: 20011215
as a Date object this date is: 12/15/2001