

Inheritance

## Using Constructors and Destructors in Derived Classes

- Knowing that a derived-class inherits its base-class' members... When an object of a derived class is instantiated, the base-class' constructor must be called
- A base-class initializer can be provided in the derived class constructor to call the base-class constructor explicitly...

```
Circle::Circle(double r, int a, int b) : Point(a,b) {
    ...
};
```

Otherwise base class's default constructor called implicitly...

 Base-class constructor and base-class assignment operators are <u>not inherited</u> by derived-class... however derived-class can call base-class constructor and assignment operator...

```
Inheritance
C++
class ABase
 public:
  ABase(int a) { x = a; } void setX(int a) { x = a; }
   void operator =(int a) { x = a; }
   void operator -=(int i) {x -= i;}
void operator +=(int i) {x += i;}
 private:
  int x;
};
class ADerived : public ABase {
 public:
  ADerived(int a, int b) : ABase(b){y = a;}
   void operator +=(int i) {y += i;}
   void operator =(ABase &aB) {
      aB = y; // Called not inherited...pani
 private:
   int y;
};
void main() {
   ABase basObj(10);
   ADerived drvObj(20,30);
  basObj = 50;
drvObj = basObj;
   basObj +=10;
  basObj -=2;
drvObj +=10;
   drvObi -=2;
   drvObj.setX(10);
                          Syntax Error... base-class
   drvObj = 100;
                          assignment operators are not
```

inherited by derived-class

C++

Inheritano 3

Implicit Derived-Class Object to Base-Class Object Conversion

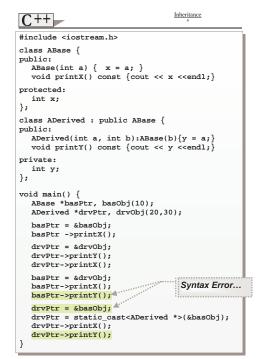
- Even though a derived-class object is\_a base-class object, the derived-class type and the base-class type are different...
- Derived-class object <u>can</u> be treated as a base-class object...
  - Derived class has members corresponding to all of the base class's members... Thus, base-class can be assigned a derived-class...
  - ➤ Needless to say, that derived-class may have more members than the base-class object...
- Base-class object <u>cannot</u> be treated as a derived-class object...
  - Would leave additional derived class members undefined...Thus derived-class cannot be assigned a base-class...
  - ➤ Obviously, *assignment operator* can be overloaded to allow such an assignment...

```
Inheritance
C++
#include <iostream.h>
class ABase {
 public:
  ABase(int a) { x = a; }
  int getX() const {return x;}
 protected:
  int x;
};
class ADerived : public ABase {
 public:
   ADerived(int a, int b) : ABase(b){
     y = a;
  void operator =(ABase &aB) {
     y = aB.getX();
private:
  int y;
};
void main() {
  ABase basObj(10);
  ADerived drvObj(20,30);
                              Syntax Error, without
  drvObi = basObi;
                              the assignment
  basObj = drvObj;
                             operator overloaded...
```



Inheritance

- Mixing base and derived class pointers and objects
   Referring to a base-class object with a base-class pointer
   allowed...
  - ▶ Referring to a derived-class object with a derivedclass pointer → allowed...
  - ➤ Referring to a derived-class object with a baseclass pointer
    - → allowed for base-class members... The derived-class object is an object of its base-class as well...
    - → Syntax error for derived class members...
  - ➤ Referring to a base-class object with a derivedclass pointer → Syntax error...The derived-class pointer must first be cast to a base-class pointer...





Inheritano 7

## **Problems with Multiple Inheritance**

```
class A {
                 public :
                     void f1(){
                       x=x;
                  protected:
              };
class B:public A {
  protected:
                            class C:public A {
   protected:
      int y;
                                   int z;
   public :
                                public :
      void f1(){
                                   void f1(){
      y=y;
                                     z=z;
};
        class ABC:public B, public C {
          protected:
              int xyz;
 void main () {
    abc.f1();
```