

Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: Inheritance in C++

Overview of This Lecture



- Inheritance
 - Motivation
 - Compositional vs. inheritance-based approaches
 - Hierarchy of classes

Acknowledgment

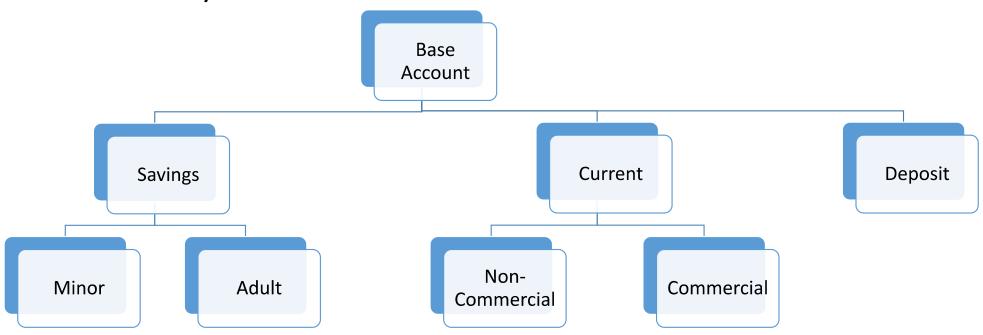


 Much of this lecture is motivated by the treatment in An Introduction to Programming Through C++ by Abhiram G. Ranade
 McGraw Hill Education 2014

A Bank Account Example



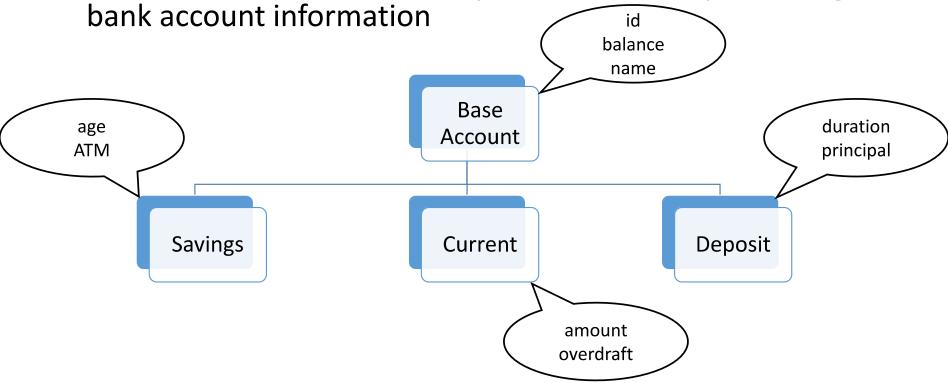
Hierarchy of Accounts



A Bank Account Example



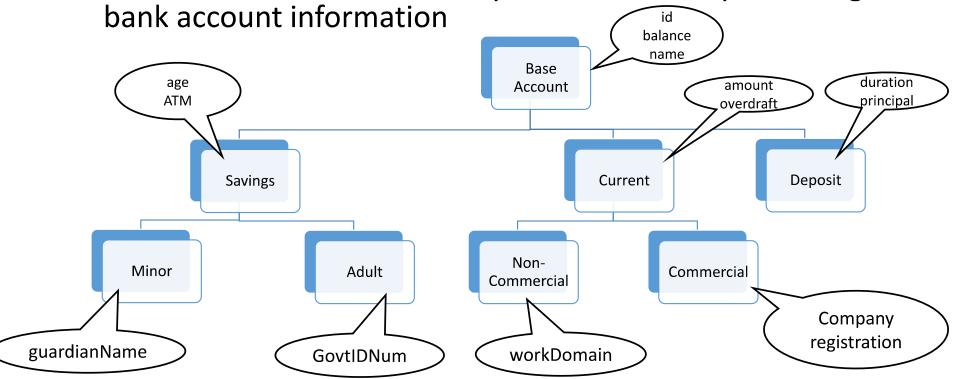
• We want to define a hierarchy of classes for representing bank account information



A Bank Account Example



• We want to define a hierarchy of classes for representing



How to define these Classes?



Compositional Way

```
class savings {
    public:
        base b;
        int age;
        long int ATM;
    };
```

```
class base {
    public:
    int id;
    float balance;
    char name[];
};
```

```
int main() {
    savings s;
    s.b.id = 1024;
    s.age = 20;
    ...
    current c;
    c.b.id = 1001;
    c.amount = 15000;
    return 0;
}
```

```
class current {
  public:
  base b;
  int amount;
  int overdraft;
};
```

Dr. Deepak B. Phatak & Dr. Supratik Chakraborty, IIT Bombay

How to define these Classes?



Inheritance Way

```
class savings : public base {
   public:
     int age;
     long int ATM;
};
```

```
class base {
    public:
    int id;
    float balance;
    char name[];
};
```

```
int main() {
    savings s;
    s.id = 1024;
    s.age = 20;
    ...
    current c;
    c.id = 1001;
    c.amount = 15000;
    return 0;
}
```

```
class current : public base {
   public:
    int amount;
   int overdraft;
};
```

Difference in Access Style



Compositional Way

```
int main() {
    savings s;
    s.b.id = 1024;
    s.age = 20;
    ...
    current c;
    c.b.id = 1001;
    c.amount = 15000;
    return 0;
}
```

Inheritance Way

```
int main() {
    savings s;
    s.id = 1024;
    s.age = 20;
...
    current c;
    c.id = 1001;
    c.amount = 15000;
    return 0;
}
```

Complex Access Control



Requirement:

Members 'id' & 'balance' of class 'base' to be accessible only from class 'savings', and not from other classes

How do we achieve this?

Compositional: Private Member & Friend Class



```
class base {
    private:
    int id;
    int balance;
    friend of base

};
```

```
class savings {
  base b; ...
  public:
  void createAccount() {
    b.id = 1;
    b.balance = 0;
  }
};
```

Compositional: Private Member & Friend Class



To extend this privilege to 'current' class

```
class current{
                       class base {
                                                           base b; ...
                        private:
                                                                                        access due
                                                          public:
                         int id;
                                                                                       to friendship
class 'savings' is a
                                                            void createAccount() {
                         int balance;
 friend of base
                                                               b.id = 1;
                         friend class savings;
                                                               b.balance = 0:
                         friend class current;
                      };
```

Can we do without inserting "friend class" declarations every time?

Towards Derived Classes



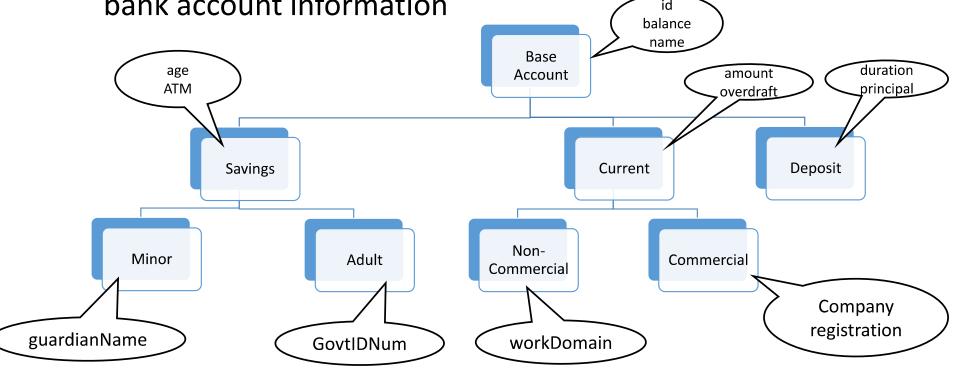
Can we explicitly say that a class is **derived** from a **base class** and **inherits attributes** of base class with special privileges to access those attributes?

Inheritance in C++

How do we specify Inherited Classes



• We want to define a hierarchy of classes for representing bank account information



Class Hierarchy



```
class base {
                      Super/Base
                                                    public:
                                                      int id; float balance;
                            Class
                                                      char name[];
  base
savings
                            class savings : public base {
                                                              class current : public base {
                                                                                                  class deposit: public base {
 minor
                             publig:
                                                               public:
                                                                                                    public:
         Derived
                               int age; long int ATM;
                                                                 int amount; int overdraft;
                                                                                                     int duration; int principal;
         Classes
                                                            class noncom: public current{
class minor: public savings{
                              class adult: public savings{
                                                                                             class com: public current{
 public:
                               public:
                                                              public:
                                                                                               public:
   char guardianName[];
                                 long int GovtIDNum
                                                               char workDomain[];
                                                                                                char companyRegistration[]
```

Complete Program



A complete program which uses all public members and shows how to access members of base class from derived class

```
#include<iostream>
using namespace std;

class base {
  public:
  int id;
  float balance;
  char name[];
};
```

```
class savings : public base {
  public:
  int age;
  long int ATM;
};
class current : public base {
  public:
  int amount;
  int overdraft;
};
```

```
int main() {
    savings s;
    s.id = 1;
    s.age = 20;
    cout << s.id << s.age;
    current c;
    c.id = 2;
    c.amount = 15000;
    cout << c.id << c.amount;
    return 0;
}</pre>
```

Summary



- Compositional vs. inheritance-based approaches
- Defining hierarchy of classes