EEEC10008(515169) S23: Object-Oriented Programming
Inheritance (II)

What you will learn from Lab 8

In this laboratory, you will learn how to use multiple inheritances.

TASK 8-1 MULTIPLE INHERITANCE

✓ A class can be directly derived from two or more base classes. It's called multiple inheritances. The class `Circle_in_Triangle` is derived from classes `Circle` and `Triangle`.

```
// lab8-1.cpp
#include <iostream>
using std::cout;
using std::endl;

class Point2D
{
   private:
      int x;
      int y;
   public:
      Point2D(){x = 0;y=0;}
      void display() const;
      // ...
};

class Circle
{
   private:
      Point2D center;
      double radius;
   public:
      void show();
      //...
};

class Triangle
{
   private:
      Point2D *vertices;
   public:
      // ...
      ~Triangle(){delete [] vertices;}
      void show();
};
```

```
class Circle_in_Triangle: public Circle, public Triangle
{
   public:
      // ...
      void show()
      {
         Circle::show();
         Triangle::show();
      }
};

int main()
{
   Point2D p(2, 2);
   Point2D *vec = new Point2D [3];

   Circle_in_Triangle ct(p,1,vec);
   ct.show();

   return 0;
}
```

TASK 8-2 AMBIGUITY RESOLUTION

✓ When two base classes have members with the same name, they can be resolved by using the scope resolution operator.

```
// lab8-2.cpp

/*  add area() for class Circle    */
/*  add area() for class Triangle */

int main()
{
   Point2D p(2, 2);
   Point2D *vec = new Point2D [3];
   vec[0].SetX(2);
   vec[0].SetY(1);
   vec[1].SetX(8);
   vec[1].SetY(1);
   vec[2].SetX(5);
   vec[2].SetY(6);
   Circle_in_Triangle ct(p,1,vec,255);
   ct.show();

   cout << "Area of Circle: " << ct.Circle::area() << endl;
   cout << "Area of Triangle: " << ct.Triangle::area() << endl;
   cout << "Area of Circle_in_Triangle: " << ct.area() << endl;
   return 0;
}
```

- The compiler shows the error message "`request for member 'area' is ambiguous`" on the screen.
- A using-declaration can bring different functions from base classes to a derived class and then overload resolution can be applied. You can add "`using Triangle::area;`" in `Circle_in_Triangle` and compile the program again.

TASK 8-3 REPLICATED BASE CLASSES

✓ With the possibility of derivation from two bases, a class can be a base twice.

```
// lab8-3.cpp

class Shape
{
   protected:
      int color;
};
class Circle: public Shape
{
   // definition in lab8-1
}
class Triangle: public Shape
{
   // definition in lab8-1
}
class Circle_in_Triangle: public Circle, public Triangle
{
public:
   // ...
   void show()
   {
      cout << "Circle's color: " << Circle::color << endl;
      cout << "Triangle's color: " << Triangle::color << endl;
      Circle::show();
      Triangle::show();
   }
};
```

- In this example, the colors of a `Circle` and a `Triangle` for an object of `Circle_in_Triangle` can be different.

TASK 8-4 VIRTUAL BASE CLASSES

✓ Often a base class need not be replicated. That is, only one copy of a replicated class needs to be inherited for a derived class object.

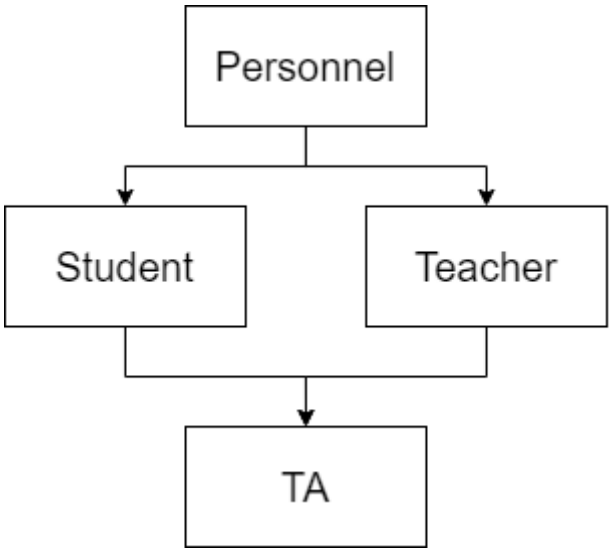
```
// lab8-4.cpp

class Shape
{
   protected:
      int color;
};
class Circle: public virtual Shape
{
   // definition in lab8-1
}
class Triangle: public virtual Shape
{
   // definition in lab8-1
}
class Circle_in_Triangle: public Circle, public Triangle
{
   public:
      // ...
      void show()
      {
         cout << "Circle's color: " << Circle::color << endl;
         cout << "Triangle's color: " << Triangle::color << endl;
         Circle:: show();
         Triangle:: show();
      }
};
```

➢ In this example, the colors of a `Circle` and a `Triangle` for an object of `Circle_in_Triangle` are the **same** since they are inherited for the same base.

EXERCISE 8-1: E3

- ✓ In this exercise, you need to design a simple E3-like course managing system.
- ✓ Please design six classes in this exercise, six classes `E3`, `Personnel`, `Teacher`, `Student`, `TA`, and `Course` to read and print information using multiple inheritances.
- ✓ Please implement all your codes in files named `E3.cpp`, `Personnel.cpp`, `Teacher.cpp`, `Student.cpp`, `TA.cpp` and `Course.cpp`.
- ✓ Do not modify `ex8-1.cpp`.
- ✓ The inheritance graph for `Personnel`, `Teacher`, `Student`, and `TA` is shown below.



- ✓ You need to parse two files, `personals.csv` and `courses.csv`. The format of the input files is shown below. The first line of the csv file indicates the column labels and the following rows are entries. Each column is separated by a comma “,”. In `courses.csv`, for `TA_IDs` and `Student_IDs` , the IDs are separated by “|”.

personals.csv
ID,Name,Email,Password,Position,Degree,Tuition,Salary 111511000,Alex,111511000@nycu.edu.tw,1234,Student,bachelor,40000, 111511001,Zack,111511001@nycu.edu.tw,1234,Student,bachelor,40000, 111511002,Jhonny,111511002@nycu.edu.tw,1234,Student,bachelor,40000, 111511003,Bob,111511003@nycu.edu.tw,1234,Student,bachelor,40000, 111511004,Jessica,111511004@nycu.edu.tw,1234,Student,bachelor,40000, 311511000,Samuel,311511000@nycu.edu.tw,tasamuel,TA,Master,30000,24000 311511001,Amy,311511001@nycu.edu.tw,taamy,TA,Master,30000,24000 311511002,Nicholas,311511002@nycu.edu.tw,tanicholas,TA,PHD,30000,40000 311511003,Brian,311511003@nycu.edu.tw,tabrian,TA,Master,30000,24000 311511004,Jimmy,311511004@nycu.edu.tw,tajimmy,TA,Master,30000,24000 311511005,Lowry,311511005@nycu.edu.tw,talowry,TA,PHD,30000,40000 opwen,Wen Hung-Pin,opwen@nycu.edu.tw,opwen,Teacher,,,2000000 tingyoyo,Lin Ting-Yu,tingyoyo@nctu.edu.tw,tingyoyo,Teacher,,,2000000

courses.csv
ID,Name,Professor_ID,TA_IDs,Student_IDs 515169,Object-Oriented Programming,opwen,311511000|311511001|311511002|311511003|311511004|311511005,111511000|111511001|111511002|111511003 535374,Wireless Ad Hoc Networks,tingyoyo,311511001|311511002,311511000|311511003|311511004|111511000|111511001 515102,Introduction to Computers and Programming,opwen,311511000|311511001|311511002|311511003|311511004|311511005,111511000|111511001|111511003

- ✓ Execute command: `>./ex8-1 ./inputs/personals.csv ./inputs/courses.csv`
- ✓ To pass the test, your program cannot contain memory leaks. You can use `valgrind` to test for memory leaks.
- ✓ Please write every class in separate files and write a "`makefile`" in your exercise directory to compile your code. Your code can be compiled by typing "`make`", and the name of your program should be `ex8-1`. You can reference "`makefile_template`" and https://makefiletutorial.com/
- ✓ Here are the output and the class template.

```
TA_Samuel@ICP:~/workspace/spring/lab8$ ./ex8-1 ./inputs/personals.csv ./inputs/courses.csv
All Personnels:
id: 111511000   Name: Alex      email: 111511000@nycu.edu.tw
id: 111511001   Name: Zack      email: 111511001@nycu.edu.tw
id: 111511002   Name: Jhonny    email: 111511002@nycu.edu.tw
id: 111511003   Name: Bob       email: 111511003@nycu.edu.tw
id: 111511004   Name: Jessica   email: 111511004@nycu.edu.tw
id: 311511000   Name: Samuel    email: 311511000@nycu.edu.tw
id: 311511001   Name: Amy       email: 311511001@nycu.edu.tw
id: 311511002   Name: Nicholas  email: 311511002@nycu.edu.tw
id: 311511003   Name: Brian     email: 311511003@nycu.edu.tw
id: 311511004   Name: Jimmy     email: 311511004@nycu.edu.tw
id: 311511005   Name: Lowry     email: 311511005@nycu.edu.tw
id: opwen       Name: Wen Hung-Pin      email: opwen@nycu.edu.tw
id: tingyoyo    Name: Lin Ting-Yu       email: tingyoyo@nctu.edu.tw

All Course:
id: 515169      Name: Object-Oriented Programming
Teacher:
        id: opwen       Name: Wen Hung-Pin      email: opwen@nycu.edu.tw        Salary: 2000000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
TAs:
        id: 311511000   Name: Samuel    email: 311511000@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511001   Name: Amy       email: 311511001@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

        id: 311511002   Name: Nicholas  email: 311511002@nycu.edu.tw    Salary: 40000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

        id: 311511003   Name: Brian     email: 311511003@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511004   Name: Jimmy     email: 311511004@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511005   Name: Lowry     email: 311511005@nycu.edu.tw    Salary: 40000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses:

Students:
        id: 111511000   Name: Alex      email: 111511000@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
```

```
Students:
        id: 111511000   Name: Alex      email: 111511000@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"

        id: 111511001   Name: Zack      email: 111511001@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"

        id: 111511002   Name: Jhonny    email: 111511002@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming"

        id: 111511003   Name: Bob       email: 111511003@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"


id: 535374      Name: Wireless Ad Hoc Networks
Teacher:
        id: tingyoyo    Name: Lin Ting-Yu       email: tingyoyo@nctu.edu.tw     Salary: 2000000
        Taught Courses: "Wireless Ad Hoc Networks"
TAs:
        id: 311511001   Name: Amy       email: 311511001@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

        id: 311511002   Name: Nicholas  email: 311511002@nycu.edu.tw    Salary: 40000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

Students:
        id: 311511000   Name: Samuel    email: 311511000@nycu.edu.tw    Degree: Master  Tuition: 30000
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511003   Name: Brian     email: 311511003@nycu.edu.tw    Degree: Master  Tuition: 30000
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511004   Name: Jimmy     email: 311511004@nycu.edu.tw    Degree: Master  Tuition: 30000
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 111511000   Name: Alex      email: 111511000@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"

        id: 111511001   Name: Zack      email: 111511001@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"


id: 515102      Name: Introduction to Computers and Programming
Teacher:
        id: opwen       Name: Wen Hung-Pin      email: opwen@nycu.edu.tw        Salary: 2000000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
```

```
id: 515102      Name: Introduction to Computers and Programming
Teacher:
        id: opwen       Name: Wen Hung-Pin      email: opwen@nycu.edu.tw        Salary: 2000000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
TAs:
        id: 311511000   Name: Samuel    email: 311511000@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511001   Name: Amy       email: 311511001@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

        id: 311511002   Name: Nicholas  email: 311511002@nycu.edu.tw    Salary: 40000
        Taught Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"
        Attend Courses:

        id: 311511003   Name: Brian     email: 311511003@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511004   Name: Jimmy     email: 311511004@nycu.edu.tw    Salary: 24000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses: "Wireless Ad Hoc Networks"

        id: 311511005   Name: Lowry     email: 311511005@nycu.edu.tw    Salary: 40000
        Taught Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
        Attend Courses:

Students:
        id: 111511000   Name: Alex      email: 111511000@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"

        id: 111511001   Name: Zack      email: 111511001@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Wireless Ad Hoc Networks" "Introduction to Computers and Programming"

        id: 111511003   Name: Bob       email: 111511003@nycu.edu.tw    Degree: bachelor        Tuition: 40000
        Attend Courses: "Object-Oriented Programming" "Introduction to Computers and Programming"
```

✓ E3.h

```
// E3.h

class E3 {
   vector<Personal*> personals;
   vector<Student*> students;
   vector<Teacher*> teachers;
   vector<TA*> TAs;
   vector<Course*> courses;

  public:
   ~E3();
   void importPersonalsFromCsv(string csvFilename);
   void importCoursesFromCsv(string csvFilename);
   void printAllPersonal();
   void printAllCourse();
};
```

✓ Course.h

```
// Course.h

class Course {
  private:
   string id;
   string name;
   Teacher* teacher;
   vector<TA*> TAs;
   vector<Student*> students;

  public:
   Course(string id, string name, Teacher* teacher);
   void addTA(TA* ta);
   void addStudent(Student* student);
   void printInfo();
   string getName();
};
```

✓ Personnel.h

```
// Personnel.h

class Personnel {
  private:
   string id;
   string name;
   string email;
   string password;
  public:
   Personnel();
   Personnel(string id, string name, string email, string password);
```

```
    // Reference
    //
https://wiki.sei.cmu.edu/confluence/display/cplusplus/OOP52-CPP.+Do+not+de
lete+a+polymorphic+object+without+a+virtual+destructor
    virtual ~Personnel() = default;
    void printInfo();
    string getId();
};
```

✓ Teacher.h

```
// Teacher.h

class Teacher : public virtual Personnel {
   protected:
    int salary;
    vector<Course*> taughtCourses;

   public:
    Teacher(string id, string name, string email, string password, int salary);
    void printInfo();
    void addTaughtCourse(Course* course);
};
```

✓ Student.h

```
// Student.h

class Student : public virtual Personnel {
   protected:
    string degree;
    int tuition;
    vector<Course*> attendedCourses;

   public:
    Student(string id, string name, string email, string password, string
degree, int tuition);
    void printInfo();
    void addAttendCourse(Course* course);
};
```

✓ TA.h

```
// TA.h

class TA : public Student, public Teacher {
   private:
   public:
    TA(string id, string name, string email, string password, string degree,
int tuition, int salary);
    void printInfo();
};
```

✓ ex8-1.cpp

```
// ex8-1.cpp

int main(int argc, char** argv) {
    string personnelsFilename = string(argv[1]);
    string coursesFilename = string(argv[2]);

    E3 e3;
    e3.importPersonnelsFromCsv(personnelsFilename);
    e3.importCoursesFromCsv(coursesFilename);

    cout << "All Personnels:" << endl;
    e3.printAllPersonnel();
    cout << endl;

    cout << "All Course:" << endl;
    e3.printAllCourse();
    cout << endl;

}
```

EXERCISE 8-2 (VERY EASY): GNU DEBUGGER PRACTICE, GDB

✓ GDB is a debugger for several languages, including C and C++. It allows you to inspect what the program is doing at a certain point during execution. Errors like segmentation faults may be easier to find with the help of gdb.

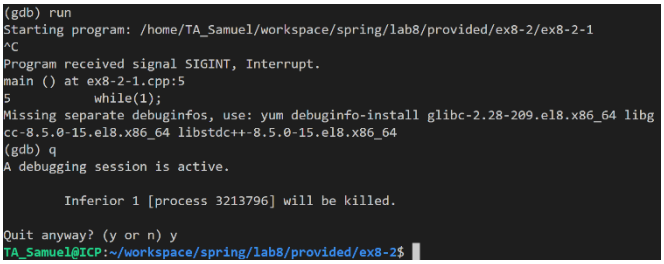
✓ You can reach detail at https://www.sourceware.org/gdb/documentation/.

✓ Here is an example for you, the result is shown in right image below.

✓ ex8-2-1.cpp

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

int main(){
    while(1);
    return 0;
}
```



✓ Commands

```
g++ <filename> -o <executable_file> -g

gdb <executable_file>

run

<ctrl + c>

q
```

- ✓ Then you can find the problem in code `ex8-2-1.cpp:7`
- ✓ Hence, please find the bug in `ex8-2-2.cpp` using `gdb` and fix it in file to get the right answer of `*t1`.
- ✓ `ex8-2-2.cpp`

```
#include <iostream>
using namespace std;
int main() {
   int* arr;
   arr[0] = 0;
   arr[1] = 1;
   arr[2] = 2;
   cout << arr[0] << " " << arr[1] << " " << arr[2] << endl;
}
```