KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện Thoại: (08) 38.354.266 - Fax:(08) 38.350.096

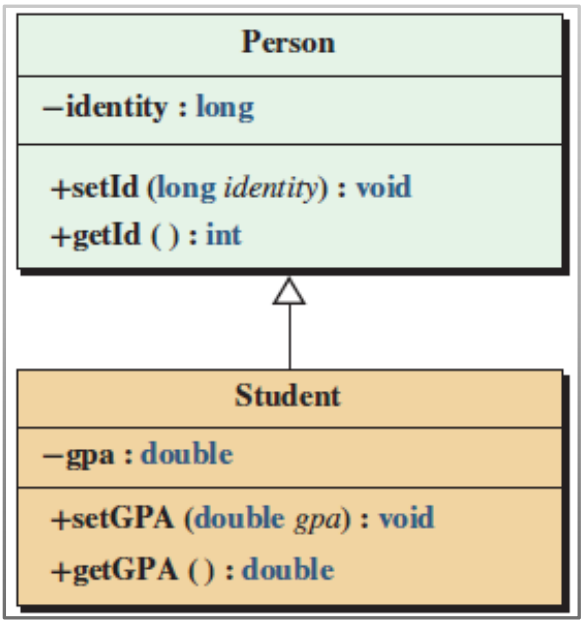
LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

BÀI TẬP TUẦN 4

2022 – 2023

Bài tập thực hành Kế thừa trong OOP

Bài 1 Thiết kế 2 lớp, Person và Student, với lớp Student thừa kế từ lớp Person. Hình 1.1 trình bày sơ đồ thừa kế với thông tin định nghĩa 2 lớp là thành phần dữ liệu và các hàm thành phần của 2 lớp.



Hình 1.1 Các lớp với thành phần dữ liệu và các hàm thành phần

Biết rằng: Lớp Person và lớp Student định nghĩa như sau:

Định nghĩa lớp Person:

Định nghĩa lớp Person

```
/*****************************************************
* Tệp giao diện của lớp Person – person.h
*****************************************************/
#ifndef PERSON_H
#define PERSON_H
#include <iostream>
#include <cassert>
#include <iomanip>
using namespace std;

class Person {
private:
	long identity;  // Ex. 211276111
public:
	//Constructors
	Person();
	Person(long identity);
	~Person();
	Person(const Person& person);

	//getter, setter
	long getId();
	void setId();

	//other methods
	void print();
};
#endif
----------------------------------------
```

Định nghĩa lớp Student

```
/*****************************************************
* Tệp giao diện của lớp Student – student.h
* Lớp Student thừa kế lớp Person
*****************************************************/
#ifndef STUDENT_H
#define STUDENT_H
#include "person.h"

class Student :public Person {
private:
	double gpa;   //Ex. 6.5
public:
	//Constructors
	Student();
	Student(long identity, double gpa);
	~Student();
	Student(const Student& student);
	//getter(); setter()
	double getGpa();
	void setGpa();
	//Other methods
	void print();
};
#endif

---------------------------------
```

Hiện thực lớp Person

```
/*****************************************************
 * File hiện thực lớp Person – person.cpp
 *****************************************************/
#include "person.h"
// Parameter constructor
Person::Person(long id)
: identity(id)
{
      assert(identity >= 100000000 && identity <= 999999999);
}
// … Other member functions
```

Hiện thực lớp Student

```
/*****************************************************
 * File hiện thực của lớp Student - student.cpp
 *****************************************************/
#include "student.h"
// Parameter constructor
// Person(id), thua ke tu lop Person
Student::Student(long id, double gp)
: Person(id), //call constructor of Person class
gpa(gp)
{
      assert(gpa >= 0.0 && gpa <= 10.0);
 }
```

Tip: '*Constructors*, *destructor*, *and* *assignment operators* *are not inherited, they must be redefined in derived class*.'

- Compound assignment operators (=, +=, -=, *=, /=, and %=).

Yêu cầu

1. Cài đặt các Constructors và assignment operator (toán tử gán bằng) và các phương thức ở lớp cơ sở 'Person' và lớp dẫn xuất 'Student'.
2. Viết chương trình kiểm thử các lớp đã viết.

Bài 2 Công ty ABC cần xây dựng ứng dụng quản lý thông tin và tính lương cho nhân viên. Thông tin mỗi nhân viên bao gồm: mã nhân viên, họ tên, ngày sinh, địa chỉ.

Công ty có 2 loại nhân viên với cách tính lương như sau: Thuộc tính: private

- Lương(Nhân viên sản xuất): số sản phẩm * 20.000 vnđ

- Lương(Nhân viên công nhật): số ngày * 300.000 vnđ

Yêu cầu

I. Vẽ sơ đồ quan hệ kế thừa giữa các lớp, với mỗi lớp có thông tin chi tiết gồm thành phần dữ liệu và hàm thành phần.

Note: *Tương tự như Hình 1.1, Bài 1*.

II. Áp dụng tính kế thừa, khai báo class NVSanXuat và NVCongNhat kế thừa class NhanVien và cài đặt các hàm sau:

1. Cài đặt các Constructors cho mỗi Class
2. Nhập thông tin nhân viên
3. Xuất thông tin nhân viên ra màn hình
4. Tính lương nhân viên

Được xài vector chỉ tuần này

Cuối kỳ phải xài con trỏ

III. Cài đặt class CongTy sử dụng mảng động hoặc danh sách liên kết. Viết các Constructors và Assignment Operator (toán tử gán bằng), cài đặt các phương thức để thực hiện các chức năng sau.

1. Nhập, xuất danh sách các nhân viên.
2. Tính tổng tiền lương của tất cả nhân viên.
3. Tìm nhân viên có lương cao nhất
4. Tính lương trung bình trong công ty
5. Nhập vào mã, tìm nhân viên tương ứng
6. Nhập vào tên, tìm nhân viên tương ứng
7. Có bao nhiêu nhân viên sinh trong tháng 5
8. Thêm một nhân viên vào danh sách.
9. Xóa một nhân viên khỏi danh sách.
10. Ghi tất cả các nhân viên có lương nhỏ hơn lương trung bình của công ty lên file ‘emp_LowerAvgSalary.dat’.

Hướng dẫn

Đoc/Ghi một đối tượng từ/lên file

Demo

```
/*****************************************************
* Guide to Read/Write an object from/to file
*****************************************************/
#include <iostream>
#include <fstream>
using namespace std;

// Class to define the properties
class Employee {
public:
	string Name;
	int Employee_ID;
	int Salary;

	Employee();
```

```
};



// Implement Employee class
Employee::Employee() {
	Name = "abc";
	Employee_ID = 123456;
	Salary = 2000;
}

int main(){

	Employee *Emp_1 = new Employee();
	Emp_1->Name = "Bingo";
	Emp_1->Employee_ID = 212121;
	Emp_1->Salary = 11000;

	//Writing this data 'Emp_1 object' to file 'Employee.txt'
	ofstream file1;
	file1.open("Employee.txt", ios::out | ios::trunc);
	file1.write((char*)&Emp_1, sizeof(Emp_1));
	cout << "\nWrite obj_Employee to file successfully!\n";
	file1.close();

	//Reading data 'Emp_1 object' from file 'Employee.txt'
	ifstream file2;
	file2.open("Employee.txt", ios::in);
	file2.seekg(0);
	file2.read((char*)&Emp_1, sizeof(Emp_1));
	cout << "\nRead obj_Employee from file successfully!\n";
	printf("\nName :%s", Emp_1->Name.c_str());
	printf("\nEmployee ID :%d", Emp_1->Employee_ID);
	printf("\nSalary :%d", Emp_1->Salary);
	file2.close();
	printf("\n---------------------- End --------------------\n");

	system("pause");
	return 0;
}

/*Write obj_Employee to file successfully!

Read obj_Employee from file successfully!

Name :Bingo
Employee ID :212121
Salary :11000
---------------------- End --------------------*/
```
