Nội dung tuần 08

Xây dựng các lớp đối tượng có quan hệ kế thừa.

Bài tập

❖ Yêu cầu

A: bài 1, bài 2

H: Làm hết 5 bài

Bài 1

Cài đặt lại ví dụ 1.

Bài 2

Cài đặt hoàn thiện ví dụ 2.

Bài 3

Cài đặt lại Bài 1 tuần 07 áp dụng virtual

Bài 4

Cài đặt lại Bài 2 tuần 07 áp dụng virtual

Bài 5

Viết chương trình quản lý các loại hình cơ bản như sau:

- ***Hình tròn***, ***Hình tam giác đều***, ***Hình chữ nhật***, ***Hình vuông***: cần có chu vi **P** và diện tích **S**.
- ***Hình cầu***, ***Hình lập phương***, ***Hình hộp***, ***Hình trụ***, ***Hình nón***: cần có diện tích xung quanh **Sxp** và thể tích **V**.

Áp dụng kế thừa và đa hình cài đặt các lớp đối tượng phù hợp để thực thi hàm main sau:

```
int main()
{
	vector<Hinh*> dsHinh;
	dsHinh.push_back(new HinhTron(3.5));
	dsHinh.push_back(new HinhTamGiacDeu(8.5));
	dsHinh.push_back(new HinhChuNhat(3.7, 1.9));
	dsHinh.push_back(new HinhVuong(4.1));
	dsHinh.push_back(new HinhCau(13.5));
	dsHinh.push_back(new HinhLapPhuong(2.9));
	dsHinh.push_back(new HinhHop(1.2, 3.7, 6.3));
	dsHinh.push_back(new HinhTru(6.1, 4.9));
	dsHinh.push_back(new HinhNon(8.7, 3.2));

	/// xuất thông tin các hình
	/// định dạng: <tên hình> [P hoặc Sxq = xx, S hoặc V = xx]
```

```
	/// ví dụ: "Hinh Vuong [P = 12.5, S = 45.7]"
	/// hoặc: "Hinh Cau [Sxp = 45.7, V = 67.4]"
	for (int i = 0; i < dsHinh.size(); i++)
	{
		cout << *dsHinh[i] << endl;
	}

	cout << endl << endl;
	system("pause");
	return 0;
}
```

Hướng dẫn

Ví dụ 1: Cách gọi phương thức của lớp đối tượng cơ sở từ lớp đối tượng con (trùng tên phương thức)

Xét ví dụ sau với lớp đối tượng cơ sở là **NhanVien** và lớp đối tượng dẫn xuất là **NhanVienKyThuat**.

```
class NhanVien
{
private:
	string hoTen, diaChi;

public:
	NhanVien();
	NhanVien(const string& ht, const string& dc);
	~NhanVien();

	friend ostream& operator<<(ostream& os, const NhanVien& nv);
};

NhanVien::NhanVien()
{
	hoTen = "";
	diaChi = "";
}

NhanVien::NhanVien(const string& ht, const string& dc)
{
	hoTen = ht;
	diaChi = dc;
}

NhanVien::~NhanVien()
{
}

ostream& operator<<(ostream& os, const NhanVien& nv)
{
	os << nv.hoTen << " (" << nv.diaChi << ")";
	return os;
}
```

Cài đặt lớp đối tượng **NhanVienKyThuat**

```
class NhanVienKyThuat : public NhanVien
{
private:
	string chungChiNganh;
public:
	NhanVienKyThuat();
	NhanVienKyThuat(const string& ht, const string& dc, const string& cc);
	~NhanVienKyThuat();

	friend ostream& operator<<(ostream& os, const NhanVienKyThuat& nvkt);
};

NhanVienKyThuat::NhanVienKyThuat() : NhanVien()
{
	chungChiNganh = "";
}

NhanVienKyThuat::NhanVienKyThuat(const string& ht, const string& dc, const
string& cc) : NhanVien(ht, dc)
{
	chungChiNganh = cc;
}

NhanVienKyThuat::~NhanVienKyThuat()
{
}

ostream& operator<<(ostream& os, const NhanVienKyThuat& nvkt)
{
	/// làm sao để xuất thông tin private ở trong class NhanVien?
	/// dòng lệnh sau bị lỗi vì KHÔNG được phép truy suất vào thành
	/// phần private của lớp đối tượng NhanVien
	os << nvkt.hoTen << " (" << nvkt.diaChi << ")";

	os << " [" << nvkt.chungChiNganh << "]";
	return os;
}
```

Ở đây vấn đề khó khăn là ở cả 2 lớp đối tượng **NhanVien** và **NhanVienKyThuat** đều có operator <<. Giải quyết vấn đề như sau

```
ostream& operator<<(ostream& os, const NhanVienKyThuat& nvkt)
{
	/// làm sao để xuất thông tin private ở trong class NhanVien?
	/// vì quan hệ IS-A nên hoàn toàn có thể coi NhanVienKyThuat
	/// là một loại NhanVien
	os << (NhanVien)nvkt;

	os << " [" << nvkt.chungChiNganh << "]";
	return os;
}
```

Thử với hàm main sau

```
int main()
{
```

```
	NhanVienKyThuat nvkt("Nguyen Van A", "Nguyen Van Cu, Q.5", "CNTT");
	cout << nvkt << endl << endl;
	system("pause");
	return 0;
}
```

```
Nguyen Van A (Nguyen Van Cu, Q.5) [CNTT]

Press any key to continue . . .
```

Ví dụ 2: Vấn đề khi cần gọi đúng phương thức của lớp đối tượng dẫn xuất

Xét ví dụ về cài đặt phương thức tính diện tích cho **HinhChuNhat** và **HinhTron** là dẫn xuất của **HinhHocPhang**.

```
class HinhHocPhang
{
public:
	HinhHocPhang() {}

	float TinhDienTich();
};

float HinhHocPhang::TinhDienTich()
{
	return 0;
}
```

```
class HinhChuNhat : public HinhHocPhang
{
private:
	float chieuDai, chieuRong;

public:
	HinhChuNhat();
	HinhChuNhat(const float& cd, const float& cr);

	float TinhDienTich();
};

HinhChuNhat::HinhChuNhat()
{
	chieuDai = chieuRong = 0;
}

HinhChuNhat::HinhChuNhat(const float& cd, const float& cr)
{
	chieuDai = (cd == 0) ? 1 : abs(cd);
	chieuRong = (cr == 0) ? 1 : abs(cr);
}

float HinhChuNhat::TinhDienTich()
{
	return chieuDai * chieuRong;
```

```
}
```

```
#define PI 3.14159
class HinhTron : public HinhHocPhang
{
private:
	float banKinh;

public:
	HinhTron();
	HinhTron(const float& bk);

	float TinhDienTich();
};

HinhTron::HinhTron()
{
	banKinh = 0;
}

HinhTron::HinhTron(const float& bk)
{
	banKinh = (bk == 0) ? 1 : abs(bk);
}

float HinhTron::TinhDienTich()
{
	return PI * banKinh * banKinh;
}
```

Cài đặt hàm main

```
int main()
{
	HinhHocPhang hhp;
	HinhChuNhat hcn(34, 56.8);
	HinhTron ht(73.9);
	cout << hhp.TinhDienTich() << "\t" << hcn.TinhDienTich() << "\t" <<
ht.TinhDienTich();
	cout << endl;
	/// vì HinhChuNhat là một loại HinhHocPhang nên có thể gán
	hhp = hcn;
	/// lúc này phương thức tính diện tích là của lớp đối tượng nào?
	cout << hhp.TinhDienTich() << endl;
	/// Giải pháp???
	cout << endl << endl;
	system("pause");
	return 0;
}
```

```
0	1931.2	17156.9
0
```

Sử dụng phương thức với từ khóa virtual (ảo hóa)

Giải pháp cho vấn đề kể trên (mong muốn gọi đúng phương thức của đối tượng thật sự khi gọi) là sử dụng từ khóa ***virtual*** cho phương thức cần **override.**

Thay đổi trong lớp đối tượng cơ sở

```
class HinhHocPhang
{
public:
	HinhHocPhang() {}

	virtual float TinhDienTich();
};
```

Ngoài ra cần phải sử dụng pointer cho các đối tượng của lớp đối tượng.

Thay đổi ở hàm main

```
int main()
{
	/// sử dụng pointer
	HinhHocPhang* hhp = new HinhHocPhang();
	HinhChuNhat* hcn = new HinhChuNhat(34, 56.8);
	HinhTron* ht = new HinhTron(73.9);
	cout << hhp->TinhDienTich() << "\t" << hcn->TinhDienTich() << "\t" << ht-
>TinhDienTich();
	cout << endl;
	/// vì HinhChuNhat là một loại HinhHocPhang nên có thể gán
	delete hhp;
	hhp = hcn;
	cout << hhp->TinhDienTich() << endl;
	cout << endl << endl;
	delete hcn;
	delete ht;

	cout << endl << endl;
	system("pause");
	return 0;
}
```

Kết quả đã phù hợp

```
0	1931.2	17156.9
1931.2
```