TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN TP.HCM
KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM
HỆ CHÍNH QUY
MÔN: **LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG**

HƯỚNG DẪN THỰC HÀNH TUẦN 6

KẾ THỪA, ĐA XẠ

- NGUYỄN LÊ HOÀNG DŨNG

TP.HCM, ngày 21 tháng 10 năm 2018

MỤC LỤC

1 Bài tập 1

- Trong một ngôi nhà nọ có nuôi 3 loại thú cưng: chó, mèo và chuột hamster. Các con thú cưng này đều có các thuộc tính: *tên, tuổi, cân nặng* (kg). Ngoài ra, mỗi loại con còn có thêm một số thuộc tính riêng biệt sau:

  + Chó: *chiều cao, chiều dài* (để làm chuồng cho phù hợp)

  + Mèo: *loại mèo* (là một chuỗi)

  + Chuột hamster: *màu lông* (là một số nguyên: 0 – màu xám tro, 1 – trắng sọc đen, 2 – màu trà sữa, 3 – màu khác)

- Số tiền thức ăn (số thực) mà mỗi con tiêu thụ trong một ngày được tính như sau:

  + Chó: cân nặng * 0.05

  + Mèo: (cân nặng – 1) * 0.04

  + Chuột: loại màu lông xám tro và màu trà sữa: (cân nặng + tuổi) *0.02, các loại còn lại: câng nặng * 0.025

Yêu cầu:

- Hãy cài đặt các lớp đại diện cho các loại thú cưng trên để:

  + Quản lý thông tin thú cưng.

  + Tính tiền ăn của thú cưng.

- Hãy cài đặt lớp **CNgoiNha** để *quản lý danh sách các con thú cưng* trong ngôi nhà, với các chức năng sau:

  + Nhập, xuất danh sách các con thú cưng trong nhà.

  + Tính tổng tiền ăn của các con thú cưng trong nhà.

Lưu ý: Cần khai báo, cài đặt đầy đủ các hàm sau:

- Các loại hàm khởi tạo và hàm hủy.
- Các hàm getter/setter.
- Các hàm để thực hiện yêu cầu bài toán.

2 Bài tập 2

Cho trước các lớp **BaseSort** như sau:

```
#include <iostream>
#include <vector>
#include <algorithm>
#include "PhanSo.h"
using namespace std;


class BaseSort{
public:
    //Hàm hủy
    virtual ~BaseSort() {};
    //Hàm thuần ảo, so sánh 2 số nguyên
    virtual bool operator()(const int& a, const int& b) = 0;
    //Hàm thuần ảo, so sánh 2 phân số
    virtual bool operator()(const CPhanSo& a, const CPhanSo& b) = 0;
};

//Lớp hỗ trợ việc in giá trị của vector số nguyên, vector phân số ra màn hình
class MyPrint
{
public:
	//Phương thức tĩnh in vector số nguyên ra màn hình
	static void printIntVector(const vector<int>& values) {
		for (unsigned i = 0; i < values.size(); ++i) {
			cout << values[i] << ' ';
		}
		cout << endl;
	}
	//Phương thức tĩnh in vector phân số ra màn hình
	static void printPhanSoVector(const vector<CPhanSo>& values) {
		for (unsigned i = 0; i < values.size(); ++i) {
			values[i].Xuat();
		}
		cout << endl;
	}
};
```

Hãy khai báo và cài đặt tất cả các lớp (CPhanSo, Asc, Desc) với phương thức cần thiết để thực hiện hàm main sau:

```
int main() {
	//Khởi tạo vector với mảng 4 phần tử số nguyên
	static const int arr[] = {16,2,77,29};
	vector<int> vInt(arr, arr + sizeof(arr) / sizeof(arr[0]) );

	//Sắp xếp tăng dần cho vector trên
	sort(vInt.begin(), vInt.end(), Asc());
	MyPrint::printIntVector(vInt);// {2,16,29,77}

	//Sắp xếp giảm dần cho vector trên
```

```
        Desc *desc = new Desc();
        sort(vInt.begin(), vInt.end(), *desc); //?
        MyPrint::printIntVector(vInt);// {77,29,16,2}

        //Khởi tạo vector với mảng 4 phần tử phân số
        static const CPhanSo arr2[] = {CPhanSo(1,2),CPhanSo(1,4),CPhanSo(3,3)};
        vector<CPhanSo> vPhanSo(arr2, arr2 + sizeof(arr2) / sizeof(arr2[0]) );

        //Sắp xếp tăng dần cho vector trên
        sort(vPhanSo.begin(), vPhanSo.end(), Asc());
        MyPrint::printPhanSoVector(vPhanSo);// {(1/4),(1/2),(3/3)}

        //Sắp xếp giảm dần cho vector trên
        Desc *desc2 = new Desc();
        sort(vPhanSo.begin(), vPhanSo.end(), *desc); //?
        MyPrint::printPhanSoVector(vPhanSo);// {(3/3),(1/2),(1/4)}

        return 0;
}
```