****

面向对象程序设计与实践

实验报告书

|  |  |
| --- | --- |
| 实验名称： | 类和对象 |
| 院系： | 计算机 |
| 专业： | 计算机科学与技术 |
| 姓名： | 肖文韬 |
| 学号： | 160800224 |

日期： 2018 年 4 月 11 日

# 实验2：类与对象

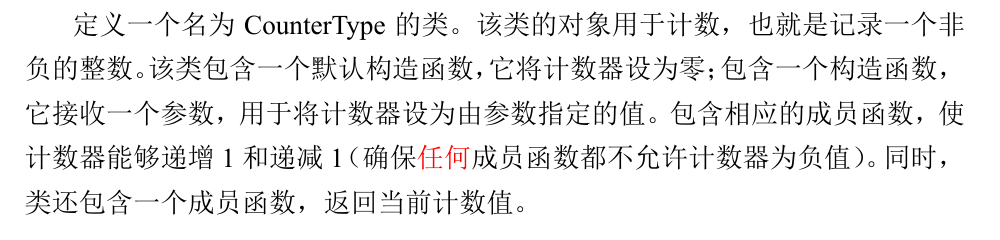
## 1. 实验目的

* 掌握用类和对象编制基于对象的程序
* 掌握对象的各种成员的使用方法
* 学习检查和调试基于对象的程序
* 掌握类的构造器和析构器
* 掌握继承机制及派生方法
* 学时数：3学时

## 2. 实验内容和步骤

2.1 编程题1

2.1.1 题目内容

2.1.2 程序设计

/\*\*

\* 获得计数器当前的值

\*/

size\_t getCount() const;

/\*\*

\* 计数器 +1

\* ! be careful overflow

\*/

size\_t addOne();

/\*\*

\* 计数器 -1

\*/

size\_t minusOne();

2.1.3 程序源码

/\*\*

\* 编程题1

\* 计数器

\*/

class CounterType {

private:

// C++ 11

size\_t count = 0;

public:

CounterType(): count(0) {

}

CounterType(int cnt) {

this->count = cnt < 0 ? 0 : cnt;

}

/\*\*

\* 获得计数器当前的值

\*/

size\_t getCount() const {

return count;

}

/\*\*

\* 计数器 +1

\* ! be careful overflow

\*/

size\_t addOne() {

return ++count;

}

/\*\*

\* 计数器 -1

\*/

size\_t minusOne() {

return count < 1 ? 0 : --count;

}

};

2.1.4程序使用说明

注意不要 Overflow 了.

2.1.5程序测试结果

CounterType cnt;

cnt.addOne();

cnt.minusOne();

cnt.minusOne();

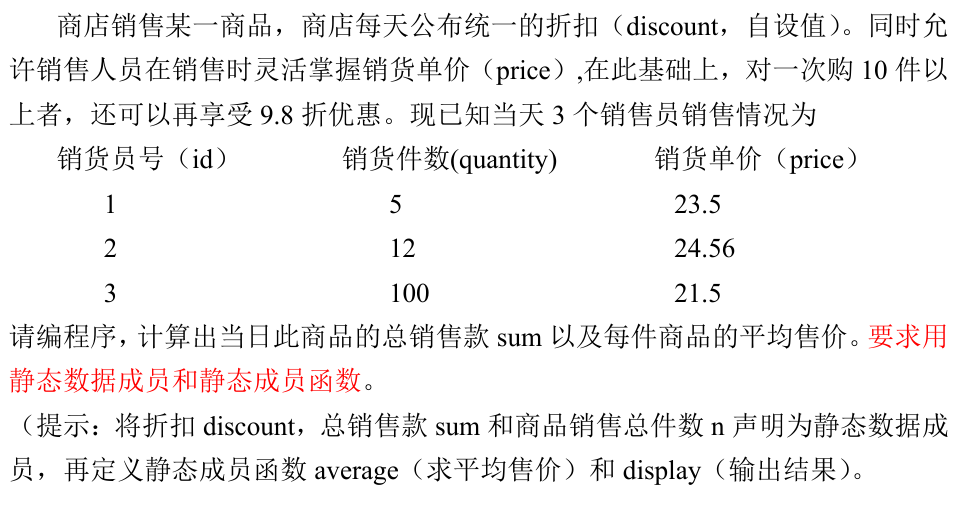
std::cout << "cnt: " << cnt << std::endl;

输出:

cnt: 0

2.2 编程题2

2.2.1 题目内容



2.2.2 程序设计

/\*\*

\* 计算平均价格

\* @return 平均价格

\*/

static double average();

/\*\*

\* 输出信息

\*/

static void display();

2.2.3 程序源码

/\*\*

\* 编程题2

\*/

class Store {

private:

// 'static' is only use to \*declaration\*(not definition) static memberm, i.e., incomplete type

// static members are not associated with any object

// if static is const/constexpr, it can be initialized with constant expression, (const static

// can be initialized outside the class while constexpr must be initialized when declared.)

static constexpr double discount = 0.8, DiscountMoreThan10 = 0.98;

static std::pair<int, double> sales[];

static double sum;

static size\_t n;

public:

/\*\*

\* 计算平均价格

\* @return 平均价格

\*/

static double average();

/\*\*

\* 输出信息

\*/

static void display();

};

// initialize them outside the class (without static keyword)

std::pair<int, double> Store::sales[] = {{5, 23.5}, {12, 24.56}, {100, 21.5}};

size\_t Store::n = 0;

double Store::sum = 0.0;

double Store::average() {

// C++ 11

for (auto i : sales) {

Store::n += i.first;

sum += (i.first > 10 ? i.first \* i.second \* discount \* DiscountMoreThan10

: i.first \* i.second \* discount);

}

return sum / n;

}

void Store::display() {

std::cout << "Average price is: " << average() << std::endl;

}

2.2.4程序使用说明

无

2.2.5程序测试结果

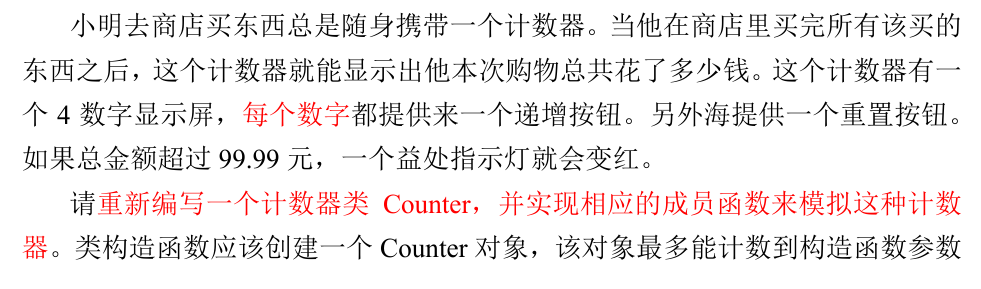
Store::display();

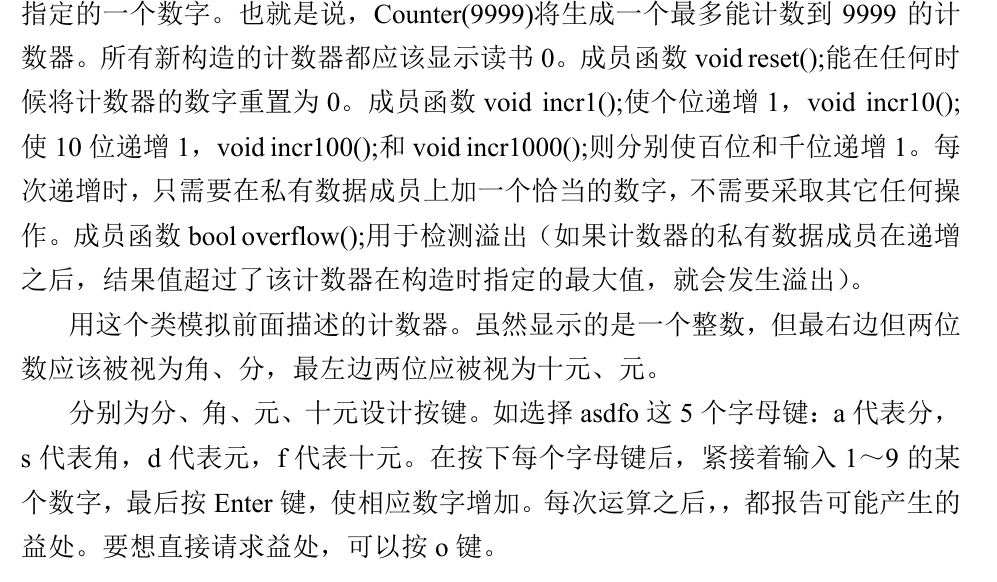
输出:

Average price is: 17.1851

2.3 编程题3

2.3.1 题目内容





2.3.2 程序设计

/\*\*

\* 判断是否溢出

\*/

bool overflow();

/\*\*

\* 个位加1

\*/

void incr1();

/\*\*

\* 十位加1

\*/

void incr10();

/\*\*

\* 百位加1

\*/

void incr100();/\*\*

\* 千位加1

\*/

void incr1000();

/\*\*

\* 重置为 0

\*/

void reset();

2.3.3 程序源码

/\*\*

\* 3

\*/

class Counter {

private:

unsigned max;

// c++ 11

unsigned cnt = 0;

public:

Counter(unsigned max = 9999) {

this->max = max > 9999 ? 9999 : max;

}

void getCount() {

std::cout << "cnt: " << cnt << std::endl;

}

/\*\*

\* 判断是否溢出

\*/

bool overflow() {

return cnt > max;

}

/\*\*

\* 个位加1

\*/

void incr1() {

cnt++;

if (overflow()) {

reset();

std::cout << "溢出\n";

}

}

/\*\*

\* 十位加1

\*/

void incr10() {

cnt += 10;

if (overflow()) {

reset();

std::cout << "溢出\n";

}

}

/\*\*

\* 百位加1

\*/

void incr100() {

cnt += 100;

if (overflow()) {

reset();

std::cout << "溢出\n";

}

}

/\*\*

\* 千位加1

\*/

void incr1000() {

cnt += 1000;

if (overflow()) {

reset();

std::cout << "溢出\n";

}

}

/\*\*

\* 重置为 0

\*/

void reset() {

cnt = 0;

}

};

2.3.4程序使用说明

无.

2.3.5程序测试结果

Counter counter = Counter();

std::string line;

while (std::getline(std::cin, line) && !line.empty()) {

std::stringstream ss(line);

char c = 0;

ss >> c;

unsigned num = 0;

ss >> num;

if (c == 'o') {

c = 'f';

num = 10;

}

while (num-- > 0) {

switch(c) {

case 'a': counter.incr1();break;

case 's': counter.incr10();break;

case 'd': counter.incr100();break;

case 'f': counter.incr1000();break;

}

}

counter.getCount();

}

输出:

a1

cnt: 1

d5

cnt: 501

f9

cnt: 9501

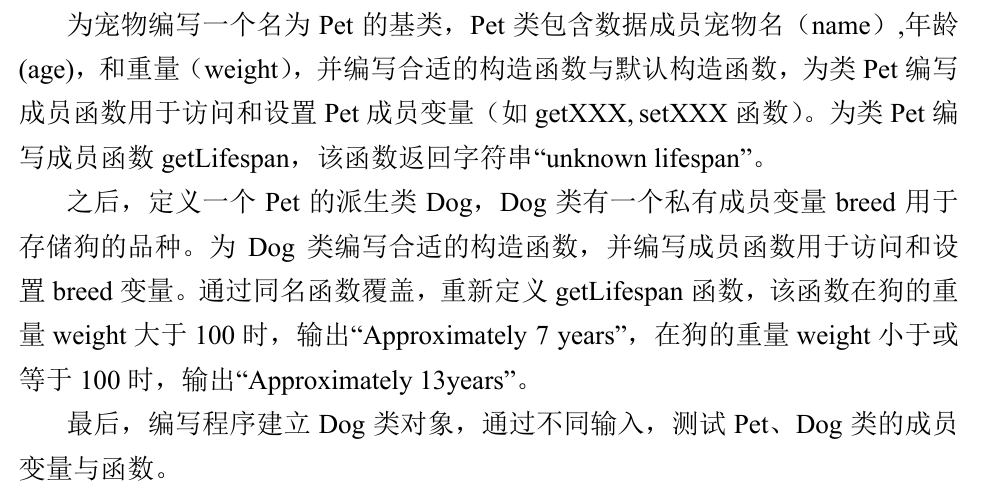
f1

溢出

cnt: 0

2.4 编程题4

2.4.1 题目内容



2.4.2 程序设计

2.4.3 程序源码

/\*\*

\* 4

\*/

class Pet {

protected:

std::string name;

size\_t age = 0;

double weight = 0.0;

public:

explicit Pet(): Pet("unknown", 0) {

}

explicit Pet(std::string name, double weight): name(name), weight(weight) {

}

std::string getName() const {

return name;

}

void setName(std::string name) {

this->name = name;

}

double getWeight() const {

return weight;

}

void setWeight(double weight) {

this->weight = weight;

}

void getLifespan() const {

std::cout << "unknown lifespan\n";

}

};

class Dog: public Pet{

private:

std::string bread;

public:

// 继承构造器

using Pet::Pet;

std::string getBread() const {

return bread;

}

void setBread(std::string bread) {

this->bread = bread;

}

void getLifespan() const {

std::cout << (weight > 100 ? "Approximately 7 years\n" : "Approximately 10 years\n");

}

};

2.4.4程序使用说明

2.4.5程序测试结果

Dog d = Dog("jerry", 101);

d.getLifespan();

std::cout << "weight: " << d.getWeight() << ", name: " << d.getName() << std::endl;

输出:

Approximately 7 years

weight: 101, name: jerry

1. 存在的问题以及解决方法；

继承构造器的问题, 查下 cppreference就好了 =,=

4. 实验心得体会