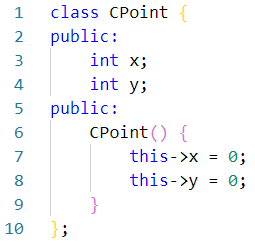
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên học phần:** | **Phương pháp lập trình hướng đối tượng** | **Mã HP:** |  |  |
| Thời gian làm bài: | **120 phút** | Ngày thi: | **27/12/2018** |  |
| *Sinh viên* ***không*** *được phép sử dụng tài liệu khi làm bài.* | | | |  |
|  | | | | |

**Câu 1**: (1 điểm)

Cho trước sơ đồ lớp Điểm trong không gian hai chiều và mã nguồn cài đặt như sau:

****

a) Hai thành phần: x và y có thể được gọi là gì của lớp?

b) Hàm CPoint() ở dòng 6 là hàm gì?

c) Từ khóa public có tác dụng gì?

d) Có thành phần nào trong cài đặt lớp có thể bỏ đi mà không ảnh hưởng đến cách hoạt động của lớp không?

**Câu 2**: (1 điểm)

Theo nguyên tắc bao bọc (encapsulation) của hướng đối tượng, các thuộc tính chứa dữ liệu của lớp không nên được truy cập trực tiếp mà phải thông qua cặp hàm getter và setter.

Giả sử ở công ty bạn có qui ước lập trình (coding convention) cho dự án C++ như sau:

1. Các hàm getter sẽ bắt đầu với tiền tố Get và theo sau là viết hoa theo PascalCase tên thuộc tính.

Ví dụ thuộc tính **name** sẽ có getter là **GetName**

1. Các hàm setter sẽ bắt đầu với tiền tố Set và theo sau là viết hoa theo PascalCase tên thuộc tính.

Ví dụ thuộc tính **name** sẽ có setter là **SetName**(string value)

Hãy **sửa lại sơ đồ lớp** và **cài đặt lớp** tương ứng.

**Câu 3**: (0.5 điểm)

Nghiệp vụ thay đổi, bạn muốn bổ sung thêm hàm tính khoảng cách tới một điểm khác có tên gọi là DistanceTo, đối số nhận vào là một điểm, kết quả trả ra số thực.

Hàm này có nguyên mẫu như sau:

double DistanceTo(const CPoint& other);

Biết lớp đã khai báo sẵn thư viện math.h để bạn có thể sử dụng hàm sqrt để lấy căn bậc hai một số.

Hãy **vẽ sơ đồ lớp mới** và **cài đặt hàm này** (chỉ cần cài đặt hàm không phải viết lại toàn bộ cài đặt của lớp).

**Câu 4:** (1 điểm)

Ở công ty cũ lương quá thấp, bạn quyết định chuyển qua công ty **BeyondSDK**, team mới của bạn có sở thích lập trình khá dị và đã đề ra các qui chuẩn lập trình như sau:

1. Tên lớp viết hoa theo kiểu PascalCase, không bắt đầu bằng chữ C.
2. Nếu là hàm thuộc nhóm Getter thì sẽ bỏ đi chữ Get, viết hoa theo kiểu PascalCase.
3. Hàm thuộc nhóm Setter thì bắt đầu với chữ Set, viết hoa theo kiểu PascalCase.
4. Tên hàm bắt đầu bằng động từ (động từ ***to be*** cũng tính là động từ)
5. Các thuộc tính nếu là private sẽ bắt đầu với m, theo sau là viết hoa theo kiểu PascalCase, ví dụ: **mCount**.

Hãy **sửa lại sơ đồ lớp của lớp Điểm** thỏa các qui chuẩn lập trình mới.

**Câu 5**: (1 điểm)

**Ở công ty BeyondSDK**, bạn được yêu cầu thiết kế ứng dụng vẽ các hình đơn giản gồm: Đường thẳng (Line), Hình chữ nhật (Rectangle), Hình Ê líp (Ellipse).

Biết rằng Đường thẳng được kiểm soát bởi 2 điểm: Bắt đầu (start) và kết thúc (end), Hình chữ nhật được kiểm soát bởi 2 điểm: Trái trên (leftTop) và Phải dưới (rightBottom), Hình Ê líp cũng được kiểm soát bởi 2 điểm: Trái trên (leftTop) và Phải dưới (rightBottom).

Để vẽ một hình ra màn hình, hàm Draw sẽ có nguyên mẫu như sau:

void Draw(HDC hdc);

Trong đó hdc là biến chứa handle tới ngữ cảnh thiết bị vẽ (device context).

Hãy **vẽ sơ đồ lớp** biểu diễn các đối tượng trong ứng dụng vẽ này.

(Chú ý ta đang ở công ty mới nên cần tuân thủ theo qui chuẩn lập trình mới)

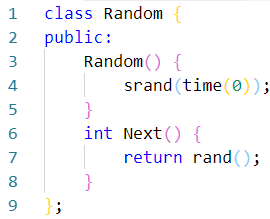
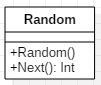
(Không yêu cầu cài đặt các lớp!)

**Câu 6**: (0.5 điểm)

Lớp đường thẳng có thể kế thừa từ lớp điểm không? Giải thích lựa chọn của bạn.

**Câu 7**: (1 điểm)

Cho lớp Random được thiết kế và cài đặt như sau:



Theo thời gian bạn thấy cần bổ sung thêm hai hàm sinh số ngẫu nhiên như sau:

int Next(int ceiling): sinh số ngẫu nhiên từ 0 đến ceiling - 1

int Next(int floor, int ceiling): sinh số ngẫu nhiên từ floor đến ceiling - 1

Hãy **vẽ sơ đồ lớp** và **cài đặt lớp** Random với hai hàm mới này.

**Câu 8**: (2.5 điểm)

**1. Viết hàm main** lần lượt làm các việc sau:

a) Phát sinh ra ngẫu nhiên số n

b) Tạo ra n hình ngẫu nhiên thuộc 1 trong 3 loại Đường thẳng, Hình chữ nhật, Hình ê líp.

(Có thể sử dụng lớp vector từ thư viện chuẩn std)

c) Lần lượt lặp qua n hình, in ra thông tin của các hình này ra màn hình (chính là 2 điểm điều khiển để vẽ ra mỗi hình) bằng hàm ToString.

Để làm được việc này, bạn có thể cần thay đổi thiết kế sơ đồ lớp ban đầu.

2. Hãy **vẽ sơ đồ lớp mới** với các thay đổi. (Chú ý không yêu cầu cài đặt hàm ToString, coi như đã có sẵn)

**Câu 9**: (1 điểm)

Xét lớp Random đã cho, với cách làm hiện tại ta luôn phải qua 2 bước để sinh số ngẫu nhiên:

1. Tạo mới thể hiện của lớp
2. Gọi hàm Next để sinh số ngẫu nhiên

Hãy sửa đổi lớp Random để chỉ có thể tạo ra một thể hiện duy nhất và không cần phải tạo thể hiện cụ thể của lớp rồi mới phát sinh số ngẫu nhiên.

Ví dụ chỉ cần viết:

int number = Random::Next();

Gợi ý: Sử dụng mẫu singleton

**Câu 10**: (0.25 điểm)

Đọc đoạn văn bên dưới và trả lời câu hỏi.

*Sometimes you want "something random", and you don't care about how that random value is arrived at. Having a static method for that could work.*

*However, sometimes you want to be able to repeatbly obtain the same random sequence. For that, you use the overload of the constructor that takes a seed value, and in that case, you don't want any other code that's using random numbers to consume one of the numbers from your sequence. In that case, you definitely need an instance of the class.*

*Having a repeatable 'random' sequence is useful in testing scenarios.*

*For example, you could use it in testing a game engine to ensure that an AI was correctly picking targets, or paths - even if it has a random path evaluation.*

Câu hỏi: Ngữ cảnh nào thì lớp Random không thích hợp cài đặt theo mẫu Singleton? Cho ví dụ?

**Câu 11:** (0.25 điểm)

Trong trường hợp của kế thừa, hàm override ở lớp con có thể trả về kiểu dữ liệu khác với kiểu dữ liệu của hàm gốc ở lớp cha hay không? Cho ví dụ giải thích.

-- **HẾT --**