

Đề thi thực hành cuối kỳ

CSC10003 – Phương pháp lập trình hướng đối tượng

Quy định nộp bài

- Nộp toàn bộ project, file báo cáo, hình vẽ UML (chỉ nộp file hình .png hay .jpg), nén lại thành tập tin MSSV.rar/zip và nộp trên Moodle. Nếu mở project nhưng chạy lỗi thì sẽ bị 0đ.
- Nếu chương trình bị lỗi **Memory Leak** sẽ bị trừ **50% tổng điểm bài làm**
- Nghiêm cấm sao chép mã nguồn, nếu phát hiện sẽ bị **0đ tất cả các bài có liên quan**
- Tổng điểm bài làm là **12 điểm**, tối đa là **10 điểm**, sinh viên có thể lựa chọn yêu cầu sao cho phù hợp để lấy 10 điểm
- **Báo cáo và sơ đồ UML bắt buộc phải có**, nếu thiếu sẽ bị trừ 1 điểm mỗi phần
- Thời gian làm bài: từ **0h00 ngày 24/12/2021 đến 23h55 ngày 25/12/2021**.
- **Không nhận bất kỳ bài làm nào nộp sau khi quá thời gian nộp.**

Bài 1: Tài khoản ngân hàng (4 điểm)

Để phục vụ cho việc đóng học phí, nhận học bổng, thanh toán các khoản chi tiêu cần thiết, bạn cần mở một tài khoản ngân hàng. Hiện việc mở tài khoản ngân hàng thông qua các hệ thống online rất tiện lợi thông qua hệ thống xác thực điện tử eKYC. Trong bài này bạn cần thiết kế các lớp để lưu trữ thông tin tài khoản ngân hàng.



Ảnh: Ngân hàng kỹ thuật số, nguồn: Google Image.


Ghi chú: Để đơn giản hóa bài này, những mô tả dưới đây đã lược bớt một số yêu cầu trong nghiệp vụ thực tế của ngân hàng.

Quy trình đăng ký một tài khoản ngân hàng mới gồm các bước nhập thông tin số điện thoại và email sau đó tiến hành nhận dạng gương mặt và các giấy tờ tùy thân. Xác nhận thông tin cá nhân và các điều khoản của ngân hàng. Ngân hàng tiến hành tạo cho bạn một **Tài khoản thanh toán (PaymentAccount)** gồm thông tin như sau: **Số tài khoản, Số dư, Ngày phát hành, danh sách Lịch sử giao dịch.**

Tài khoản này cho phép người dùng nhận và chuyển khoản thông qua cách dịch vụ internet banking thông qua web hoặc mobile banking thông qua ứng dụng của ngân hàng trên điện thoại di động. Khi sử dụng các dịch vụ này hệ thống cần phản hồi lại thông qua các phương thức:

- 🚦 **Double getBalance** trả về số dư hiện tại của tài khoản.
- 🚦 **Boolean transferTo (double amount)** Thực hiện thanh toán với số tiền amount đồng thời lưu lại thông tin vào lịch sử giao dịch. Phương thức trả về True nếu giao dịch thành công, ngược lại trả về False. Nếu balance lớn hơn amount thì giảm balance đi một lượng

bằng amount, lấy ngày hiện tại lưu vào lịch sử giao dịch. Khi đó giao dịch được tính là một giao dịch thành công.

 **showHistory** Hiện danh sách các lịch sử giao dịch, nếu giao dịch nhận tiền thì số tiền chuyển khoản là một số dương, ngược lại nếu chuyển khoản thanh toán thì là một số âm.

Lịch sử giao dịch lưu lại một thông tin danh sách chuyển khoản của người dùng gồm: **Số tài khoản, Số tiền chuyển khoản, Nội dung chuyển khoản, Ngày giao dịch**. Ngày giao dịch được lưu theo định dạng dd/MM/yyyy.


Bên cạnh tài khoản thanh toán, để sử dụng thanh toán quốc tế và online dễ dàng, bạn quyết định đăng ký và mở một tài khoản thẻ tín dụng với hạn mức là 30.000.000đ.


Tài khoản thẻ tín dụng (CreditCardAccount) hỗ trợ bạn thanh toán chậm (thực hiện chi tiêu trước) đối với bất kỳ giao dịch nào sau đó bạn sẽ trả tiền cho ngân hàng khi đến hạn thanh toán. Bạn sẽ có thời gian 45 ngày để thanh toán mà không áp dụng lãi suất. Tức là bạn có thể chi tiêu từ kỳ đến cuối kỳ (chu kỳ 1 tháng) và có thêm 15 ngày để thanh toán cho tất cả giao dịch trong kỳ đó. Lúc này bạn có hai lựa chọn là thanh toán hết hoặc thanh toán số tiền tối thiểu bằng 5% tổng số tiền đã chi tiêu trong kỳ đó. Nếu bạn không thanh toán bất kỳ khoản tiền nào (tối thiểu hoặc trả hết) thì bạn phải chịu phạt 2.000.000đ phí trả chậm và đồng thời chịu lãi trên số tiền trong kỳ sao kê đó. Nếu bạn chỉ thanh toán tối thiểu thì bạn không phải chịu phí phạt trả chậm nhưng vẫn chịu lãi trên số tiền còn chưa thanh toán hết trong kỳ. Số tiền còn phải thanh toán trong kỳ, tiền lãi và tiền phí trả chậm sẽ được cộng vào kỳ thanh toán kế tiếp.


Thẻ tín dụng có những thuộc tính và phương thức sau:


Thuộc tính: **Hạn mức tín dụng (creditLimit), Lãi suất (interestRate), Thanh toán tối thiểu (minPayment), Phí trả chậm (latePenalty), Số dư hiện tại (balance)**

Cách hoạt động của tài khoản thẻ tín dụng được hỗ trợ bởi các phương thức như sau:

 **Double getBalance.** Lấy thông tin tổng số tiền chi tiêu từ thẻ tín dụng

 **Boolean charge (double amount)** Kiểm tra tài khoản hiện tại có thể thanh toán số tiền amount hay không. Tài khoản thực hiện được thanh toán khi $amount + balance \leq creditLimit$. Nếu thanh toán được thì thực hiện cộng amount vào balance của tài khoản thẻ và trả về True. Nếu không thanh toán được thì trả về False.

 **Void payment(double amount)** Thanh toán thẻ tín dụng, thực hiện trừ số tiền amount vào balance (lúc này balance có thể đạt giá trị âm).

 **showHistory** Hiện danh sách các lịch sử giao dịch.

Thẻ tín dụng của ngân hàng tùy vào mục đích sử dụng có thể có một số tiện ích đi kèm thẻ như tích điểm hoặc hoàn tiền.

Tài khoản thẻ tích điểm (RewardCardAccount) là một CreditCardAccount cho phép người dùng tích điểm (là số nguyên dương) trên mỗi giao dịch. Do đó RewardCardAccount có những khác biệt sau:

- ✚ Có thêm thuộc tính **rewardRate** là một số thực biểu thị tỉ lệ hoàn điểm, **currentPoints** là số điểm hiện tại trong tài khoản.
- ✚ Hàm **getCurrentPoints** trả về một số nguyên là số điểm tích lũy được.
- ✚ Phương thức **charge** xử lý tương tự CreditCardAccount, thêm vào đó là khi phương thức trả về True cần phải tính số điểm được nhận là *amount* nhân với *rewardRate*, chuyển số điểm về kiểu int (làm tròn xuống) và thêm số điểm này vào *currentPoints*.
- ✚ Có thêm phương thức **payWithPoints(int pAmount)** để thanh toán giao dịch với số điểm là *pAmount*. Nếu *pAmount* lớn hơn *currentPoints*, phương thức không cần thực hiện thao tác nào. Ngược lại, giảm *currentPoints* đi một lượng *pAmount*.

Tài khoản thẻ hoàn tiền (CashbackCardAccount) là một CreditCardAccount cho phép người dùng nhận được số tiền hoàn lại trên mỗi giao dịch. Do đó CashbackCardAccount có những khác biệt sau:

- ✚ Có thêm thuộc tính **cashbackRate** là một số thực biểu thị tỉ lệ hoàn tiền, **currentCashBack** là số tiền có thể hoàn lại hiện tại trong tài khoản.
- ✚ Hàm **getCurrentCashBack** trả về số tiền có thể hoàn lại hiện tại.
- ✚ Phương thức **charge** xử lý tương tự CreditCard, thêm vào đó là khi phương thức trả về True cần phải tính số tiền hoàn lại được nhận là *amount* nhân với *cashBackRate* và thêm số tiền này vào **currentCashBack**.
- ✚ Có thêm phương thức **redeemCashBack** để tiến hành hoàn tiền, phương thức sẽ giảm số tiền *currentCashBack* trực tiếp vào trong *balance* và đặt *currentCashBack* lại mặc định bằng 0.

Yêu cầu thực hiện:

- ✚ Xây dựng các lớp **PaymentAccount**, **PaymentHistory**, **CreditCardAccount**, **RewardCardAccount**, **CashBackCardAccount** với các thuộc tính và phương thức mô tả ở trên. **(1 điểm)**
- ✚ Thiết kế đúng các mối quan hệ giữa các lớp và viết đúng các phương thức được mô tả. **(1 điểm)**
- ✚ Hoàn thiện phần nhập xuất cho chương trình chính và từng lớp. **(0.5 điểm)**
- ✚ Loại tài khoản tín dụng có chức năng phát hành thẻ để sử dụng trực tiếp. Mỗi tài khoản có thể phát hành 1 thẻ chính và tối đa 5 thẻ phụ. Mỗi thẻ có các thông tin về số thẻ, ngày phát hành và tài khoản thẻ liên kết đến. Hãy thiết kế và viết thêm các thành phần để thực hiện chức năng này. **(1 điểm)**
- ✚ Thiết kế chương trình chính và các lớp để hỗ trợ cho việc thực hiện thanh toán thẻ tín dụng và sao kê thẻ tín dụng vào cuối kì (vào ngày 15 hằng tháng). **(0.5 điểm)**

Bài 2: Trò chơi chiến thuật Đại chiến hai thế giới (6 điểm + 2 điểm)

Trò chơi chiến thuật là một cách gọi đơn giản của thể loại trò chơi chiến lược theo thời gian thực (real-time strategy) – là một trong những thể loại trò chơi phổ biến quen thuộc với nhiều tựa game nổi tiếng từ xưa đến nay: Age of Empire, Warcraft, Starcraft... Khác với thể loại chiến thuật theo lượt, trò chơi chiến lược theo thời gian thực yêu cầu đòi hỏi người chơi cần phải có tầm nhìn chiến thuật và cách quản lý cũng như có sự tính toán chính xác, tốc độ, quyết đoán mang tầm vĩ mô và vĩ mô.

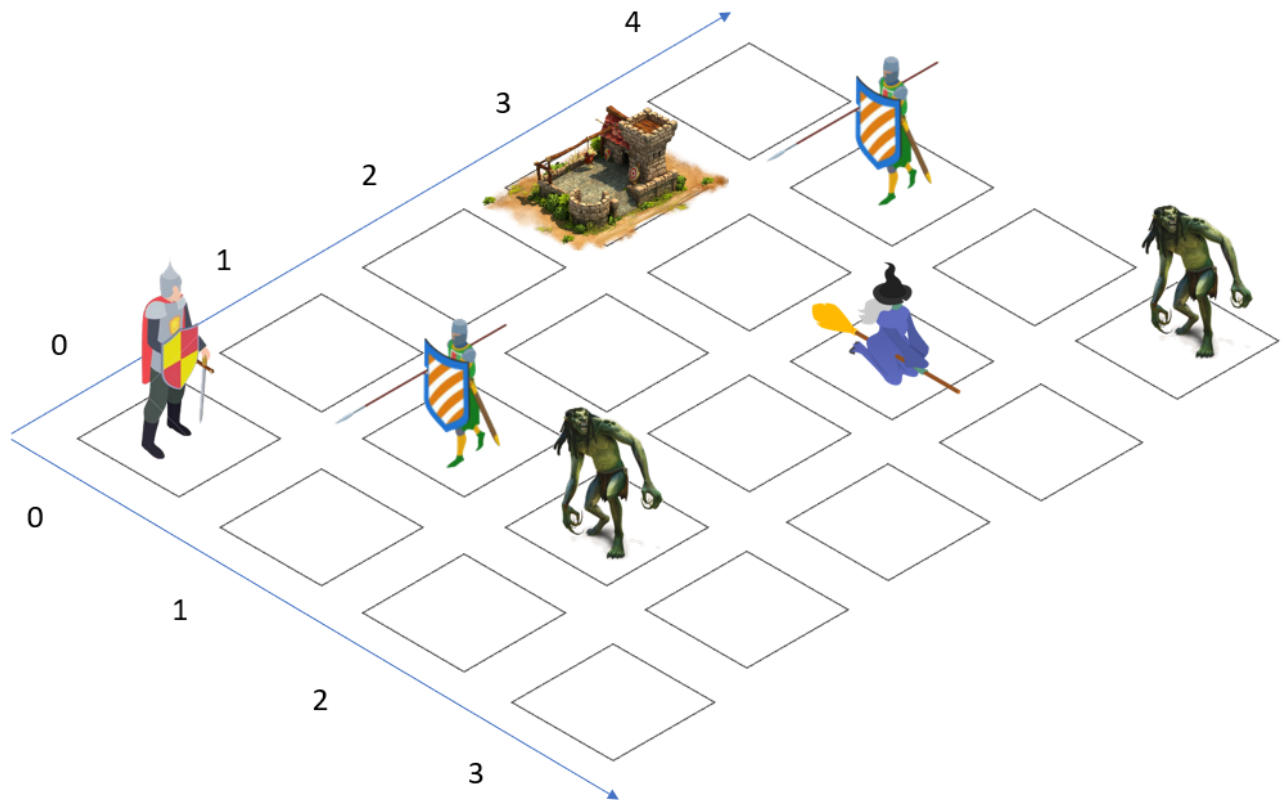


Ảnh: trò chơi Warcraft III (nguồn: sưu tầm từ internet)

Trò chơi chiến lược theo thời gian thực đa phần hướng đến việc sử dụng và khai thác hiệu quả các đơn vị lính, công trình, thu thập tài nguyên để xây dựng, phát triển đội ngũ theo chiến thuật tối ưu nhất nhằm mục đích giành chiến thắng theo mục tiêu nhiệm vụ hoặc giành chiến thắng khi tranh đấu với người chơi khác. Theo đó, các trò chơi điển hình của thể loại này thường có tính năng xây dựng và nâng cấp căn cứ, chiêu mộ và phát triển binh lính đồng thời cho phép người chơi được điều khiển di chuyển, tấn công và phòng thủ dựa trên đội quân của mình.

Trong đề án lần này, các bạn sẽ viết chương trình C++ xây dựng trò chơi thể loại chiến lược theo thời gian thực, được mang tên là **“Đại chiến hai thế giới”**. Cốt truyện lấy bối cảnh thần thoại, khi thế giới con người từ thuở sơ khai đã bị xâm chiếm bởi một thế lực quỷ dữ đen tối muốn thống

trị thế giới loại người. Bạn trong vai trò là một thủ lĩnh sẽ chiêu mộ và dẫn dắt các đội quân tinh nhuệ tiêu diệt toàn bộ thế lực tàn ác.



Ở phiên bản đầu tiên của trò chơi, bạn sẽ tiến hành xây dựng nên những thành phần cơ bản cần thiết để tạo nên môi trường. Tất cả mọi hoạt động của người chơi cũng như các sự kiện xảy ra đều ở trên **bản đồ** (toàn bộ trò chơi chỉ có 1 bản đồ duy nhất). Trên **bản đồ** sẽ được đặt *các loại công trình cơ bản, các loại binh lính cũng như các quái vật* sẽ xuất hiện. Bản đồ này có hình chữ nhật với kích thước được tính toán dựa theo *chiều dài, chiều rộng*. Mỗi vị trí trên bản đồ được mô tả thông qua *hệ thống tọa độ XY* (vị trí *X* tương ứng với đơn vị chiều rộng, vị trí *Y* tương ứng với đơn vị chiều dài). Khi trò chơi bắt đầu, bản đồ sẽ được *khởi tạo* bằng cách quy định vị trí các công trình. Thông tin quy định được nhập trực tiếp từ bàn phím qua console.

Mỗi công trình trên bản đồ được phép tạo ra một loại quân lính khác nhau. Quân lính khi được tạo sẽ xuất hiện ngẫu nhiên tại một trong 8 vị trí xung quanh bản đồ (trừ các ô đã có chứa công trình hoặc quân lính trước đó). Khi nhập thông tin của công trình, hãy cho phép người dung nhập thông tin các quân lính.

Người chơi ban đầu được mặc định đặt tại vị trí $X = 0$ và $Y = 0$. Người chơi được *hỗ trợ điều khiển di chuyển* đến vị trí chỉ định thông qua tọa độ X và Y , vậy có tổng cộng 4 trường hợp khi người chơi đi đến vị trí mong muốn:

- 📌 Đến vị trí trống: không có sự kiện gì xảy ra
- 📌 Đến vị trí có quân lính: người chơi sẽ thu nhận quân lính đó vào trong binh đoàn của mình
- 📌 Đến vị trí có quái vật: người chơi sẽ dùng binh đoàn của mình tấn công quái vật đó

Trong bản đầu tiên của trò chơi, có một dạng **công trình đặc biệt** mang chức năng *nâng cấp lính*. Công trình này có thể xuất hiện nhiều nơi trên bản đồ. Mỗi công trình sẽ đều có *tên gọi riêng* và nằm trên một *vị trí* của bản đồ.

Công trình nâng cấp lính cho phép *nâng cấp toàn bộ lính* có trong đội quân của người chơi khi người chơi ghé thăm. Cấp độ lính sẽ được nâng cấp lên 1 đơn vị. Tuy nhiên, khi đó công trình cũng sẽ biến mất khỏi bản đồ.

Tất cả binh lính sẽ có 2 loại chỉ số là tấn công và phòng thủ. Tuy nhiên, tùy thuộc vào từng loại lính (tương ứng với từng loại công trình) sẽ có chỉ số tấn công và phòng thủ cơ bản (ban đầu) khác nhau. Sinh viên tự đề xuất thêm các chỉ số của từng loại binh lính. Khi các binh lính tăng cấp, các chỉ số sẽ được thay đổi công thức sau:

Chỉ số tăng theo cấp, mỗi cấp sẽ tăng 1 điểm tấn công và 5 điểm phòng thủ

Tất cả binh lính đều có *tên gọi* để phân biệt với nhau, có *lượng máu*, *vị trí* và *sức mạnh tấn công*.

Thế lực tàn ác bắt đầu quá trình thôn tóm thế giới loài người bằng cách thả quái vật chiếm đóng tại các vị trí tương ứng trên bản đồ. Do mới bắt đầu thời kỳ sơ khai, thế lực tàn ác thả quái vật thích nghi với môi trường rừng rậm để chiếm đóng các khu rừng và điều khiển các loài động vật khác. **Quái vật** cũng có *tên gọi*, có *lượng máu*, *vị trí* và *cấp độ*. **Quái vật rừng rậm** này có khả năng tấn công dựa trên *điểm nộ*, quái vật có *cấp độ* càng cao, *điểm nộ* tăng lên càng nhiều, cụ thể:

Điểm nộ của quái rừng tăng 3 điểm mỗi khi quái tăng 1 cấp

Những quái vật chỉ có 1 điểm nộ. Quái vật có *khả năng tấn công* người chơi khi người chơi dừng tại vị trí mà quái vật chiếm đóng. Mỗi quái vật rừng đều rất khỏe và có khả năng tấn công toàn bộ binh lính của người. Nếu toàn bộ binh lính đều hết máu, người chơi sẽ thua và thế giới sẽ

thuộc về thể lực tàn ác. Quái vật chỉ bị tiêu diệt khi quái không còn máu, đồng thời người chơi cần phải sống sót để tiếp tục hành trình. Khi quái vật *tấn công lính*, lính sẽ bị trừ máu tùy thuộc vào loại lính có khả năng phòng thủ hay không. Khả năng tấn công của quái vật rừng rậm được mô tả:

*Sức tấn công quái rừng được tính bằng sức mạnh của quái vật + (điểm nộ * 2)*

Việc nhập xuất thông tin của tất cả các đối tượng được hỗ trợ qua việc nhập xuất trên màn hình console. Ngoài ra, các mảng và danh sách được yêu cầu sử dụng vector, các dữ liệu liên quan đến chuỗi yêu cầu sử dụng kiểu string của thư viện string trên C++. Các dữ liệu khác các bạn tự đề xuất lựa chọn kiểu dữ liệu phù hợp.

Yêu cầu thực hiện: (4 điểm)



Bạn hãy xây dựng chương trình C++ để xuất phát triển hệ thống mã nguồn cho trò chơi “Đại chiến hai thế giới” sử dụng kiến thức lập trình hướng đối tượng với các kỹ thuật kế thừa, đa hình và áp dụng các mẫu thiết kế hướng đối tượng phù hợp dựa vào mô tả nêu trên. (2 điểm)

Ngoài ra, bạn hãy viết báo cáo để mô tả lại hệ thống mã nguồn mà bạn đề xuất xây dựng, đồng thời đưa ra những nhận định, nhận xét và phân tích về các đoạn mã nguồn mà bạn tâm đắc trong đồ án (khả năng mở rộng, tùy biến linh hoạt, tính tổng quát hóa, ...) (1 điểm)

Bạn hãy vẽ sơ đồ UML tương ứng cho hệ thống bạn đề xuất xây dựng (1 điểm)

Lưu ý: Phần nào không có trong mô tả / yêu cầu, sinh viên có thể tự đề xuất thiết kế.

Phản điểm cộng: (2 điểm)

-  Báo cáo có sự phân tích rõ ràng, chi tiết, hợp lý, trình bày báo cáo đẹp (1 điểm)
-  Mã nguồn có tổ chức hợp lý, có khả năng mở rộng và tính linh hoạt, mã nguồn có chú thích đầy đủ (1 điểm)

Phần yêu cầu nâng cao (2 điểm)

Bạn được yêu cầu phát triển thêm bản cập nhật cho trò chơi. Trong bản cập nhật tiếp theo, bạn sẽ cập nhật thêm một loại công trình và một loại quái vật khác để đa dạng hóa lối chơi cũng như có thêm nhiều chiến thuật cho người chơi lựa chọn. Công trình mới được thêm vào cho phép khả năng nhân bản binh lính. Công trình nhân bản binh mã đều có những thông tin như tên gọi và vị trí tọa độ trên bản đồ. Khi người chơi viếng thăm, công trình sẽ hỗ trợ nhân bản lính dựa trên

tên lính. Lính nhân bản tất cả thông tin hiện tại của lính đó. Ngoài ra, công trình này quy định số lượng lính nhân bản tương ứng. Khi công trình tiến hành nhân bản sẽ lựa chọn ngẫu nhiên số lượng lính dựa trên số lượng nhân bản cho phép của công trình và thêm tất cả lính nhân bản vào cuối danh sách binh mã của người chơi. Khi nâng cấp công trình, số lượng lính nhân bản cũng được tăng tương ứng theo tỉ lệ: mỗi cấp độ tăng cho phép nhân bản thêm 1 lính.

Một loại quái vật khác rất mạnh được thêm vào để tăng độ khó và thử thách cho người chơi. Quái vật phù thủy có khả năng triệu hồi thêm quái vật rừng rậm và quái vật phù thủy khác. Do đó quái vật phù thủy nắm trong tay danh sách các quái vật triệu hồi. Danh sách này bị giới hạn số lượng dựa trên cấp độ của quái vật phù thủy và được tăng lên 1 quái vật triệu hồi thêm đối với mỗi cấp tăng thêm. Sức mạnh tấn công của quái vật phù thủy bao gồm sức mạnh của bản thân phù thủy đó cộng với sức mạnh của tất cả quái vật trong danh sách triệu hồi. Phù thủy có lượng máu bằng lượng máu của quái vật cộng với lượng máu của tất cả quái vật trong danh sách triệu hồi. Do đó khi lính tấn công quái phù thủy cần phải cân nhắc thật kỹ lưỡng.