ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

Kỹ thuật Lập trình - CO1027

Bài tập lớn 1

VUA HẢI TẶC - ONE PIECE
Arc Water 7 - Enies Lobby (phần 1)

Phiên bản 1.0

TP. HỒ CHÍ MINH, THÁNG 02/2026

ĐẶC TẢ BÀI TẬP LỚN

Phiên bản 1.0

1 Chuẩn đầu ra

Sau khi hoàn thành bài tập lớn này, sinh viên ôn lại và sử dụng thành thục:

- Các cấu trúc rẽ nhánh
- Các cấu trúc lặp
- Mảng 1 chiều và mảng 2 chiều
- Xử lý chuỗi ký tự
- Hàm và lời gọi hàm
- Các thao tác đọc/ghi tập tin

2 Dẫn nhập

One Piece là bộ truyện nổi tiếng xoay quanh hành trình của Monkey D. Luffy và các cộng sự trên con đường chinh phục kho báu huyền thoại One Piece, hướng đến mục tiêu trở thành Vua Hải Tặc. Xuyên suốt câu chuyện, tác phẩm không chỉ khai thác các chuyến phiêu lưu và xung đột trên đại dương, mà còn nhấn mạnh những giá trị cốt lõi như tinh thần đồng đội, lòng trung thành, khát vọng cá nhân và sự lựa chọn giữa công lý và tự do. Mỗi chặng hành trình (arc) là một tập hợp các thử thách mới, nơi các nhân vật phải đưa ra quyết định, chấp nhận hệ quả và trưởng thành qua từng biến cố.

Trong arc Water 7 – Enies Lobby, khi đặt chân đến Water 7 – vùng đất nổi tiếng với ngành đóng tàu – nhóm nhân vật chính phải đối mặt với nhiều biến cố liên tiếp. Con tàu Going Merry bị kết luận hư hỏng nghiêm trọng và không thể sửa chữa, những bất đồng nội bộ dần xuất hiện, đồng thời các âm mưu liên quan đến tổ chức CP9 và Robin dần được hé lộ. Chuỗi sự kiện này dẫn nhóm nhân vật đến Enies Lobby, nơi họ buộc phải vượt qua nhiều thử thách nguy hiểm để bảo vệ đồng đội và tiếp tục hành trình phía trước.

Cảm tác từ mạch truyện trên, Bài tập lớn (BTL) được thiết kế nhằm mô phỏng lại các sự kiện theo hướng logic và có tính chiến thuật, qua đó giúp sinh viên vận dụng và rèn luyện các kỹ thuật lập trình C++ cơ bản thông qua một kịch bản có tính liên kết và độ khó tăng dần. BTL sẽ không giữ nguyên toàn bộ cốt truyện mà sẽ chỉ lấy sự kiện làm mốc để xây dựng bài toán lập trình dựa trên đó cho phù hợp với yêu cầu đề bài.

Trong BTL 1, nội dung đề bài tập trung vào giai đoạn đầu của chặng hành trình này, khi các sự kiện diễn ra liên tiếp và dẫn đến những rạn nứt trong nội bộ nhóm, tạo tiền đề cho các thử thách và quyết định quan trọng về sau.

3 Dữ liệu nhập

3.1 Mô tả dữ liệu nhập

Dữ liệu nhập của chương trình được chứa trong một file, tên file sẽ được lưu trong biến **opw_input**. Mỗi dòng có định dạng chung:

```
<TÊN>␣<CÁC␣CHỈ␣SỐ>
```

Cấu trúc file như sau:

```
<NAME_1>␣<HP_1>␣<SKILL_1>
<NAME_2>␣<HP_2>␣<SKILL_2>
...
<NAME_7>␣<HP_7>␣<SKILL_7>
GOING_MERRY␣<SHIPHP>␣<REPAIRCOST>
```

Trong đó:

- `NAME_i`: Tên nhân vật thứ i (chuỗi không có khoảng trắng, có thể dùng chữ cái và dấu _). Các nhân vật có thể là: LUFFY, ZORO, SANJI, NAMI, CHOPPER, USOPP, ROBIN
- `HP_i`: Chỉ số bền bỉ của nhân vật thứ i (số nguyên).
- `SKILL_i`: Chỉ số năng lực tổng hợp của nhân vật thứ i (số nguyên).
- `GOING_MERRY`: Dòng mô tả tàu (từ khóa cố định).
- `SHIPHP`: Mức độ nguyên vẹn của tàu (số nguyên).
- `REPAIRCOST`: Chi phí sửa chữa tàu (số nguyên).

Lưu ý: Các dòng không có thứ tự cố định.

Ràng buộc dữ liệu

- Với mỗi nhân vật:
  - $0 \leq$ `HP_i` ≤ 1000

  - $0 \leq$ SKILL_i ≤ 100

- Với tàu:
  - $0 \leq$ SHIPHP ≤ 1000
  - $0 \leq$ REPAIRCOST ≤ 3000

Nếu giá trị đọc được nhỏ hơn 0 thì gán bằng 0; nếu vượt quá giới hạn thì gán bằng giới hạn trên tương ứng. Các phép tính cần làm tròn về số nguyên thì làm tròn lên số nguyên lớn hơn gần nhất.

Ví dụ minh họa:

```
LUFFY␣120␣95
ZORO␣110␣90
SANJI␣105␣88
NAMI␣80␣60
CHOPPER␣90␣55
USOPP␣70␣50
ROBIN␣85␣65
GOING_MERRY␣320␣1500
```

3.2 Nhiệm vụ 0: Đọc dữ liệu đầu vào

3.2.1 Mô tả hàm

- **Tên hàm:** `readInput`
- **Tham số đầu vào:**
  - `filename`: Tên tập tin chứa dữ liệu đầu vào.
  - `character`: Mảng 2 chiều lưu tên các nhân vật theo thứ tự xuất hiện.
  - `hp`: Mảng lưu chỉ số bền bỉ tương ứng với từng nhân vật.
  - `skill`: Mảng lưu chỉ số năng lực tương ứng với từng nhân vật.
  - `shipHP`: Chỉ số mức độ nguyên vẹn của tàu.
  - `repairCost`: Chi phí sửa chữa tàu.
- **Kết quả trả về:**
  - `true`: Nếu tập tin được mở và dữ liệu được đọc thành công.
  - `false`: Nếu không mở được tập tin hoặc dữ liệu không hợp lệ.

- **Chức năng:** Sinh viên cần hiện thực hàm `readInput` để đọc dữ liệu từ tập tin đầu vào và lưu trữ thông tin vào các mảng tương ứng.
  - Đọc lần lượt từng dòng trong tập tin đầu vào.
  - Mỗi dòng có dạng `<TÊN> <CHỈ_SỐ_1> <CHỈ_SỐ_2>`.
  - Nếu `<TÊN>` là tên nhân vật:
    * Lưu tên vào mảng `character` theo thứ tự xuất hiện (không lặp).
    * Lưu hai chỉ số tương ứng vào mảng `hp` và `skill`.
    * Nếu tên nhân vật đã xuất hiện trước đó, cập nhật lại chỉ số sau cùng.
  - Nếu `<TÊN>` là `GOING_MERRY`:
    * Lưu giá trị vào `shipHP` và `repairCost`.

  Sau khi đọc xong, dữ liệu trong các mảng và biến được **đảm bảo hợp lệ và đúng theo các quy ước của đề bài**, sẵn sàng cho các nhiệm vụ tiếp theo.

3.2.2 Ví dụ minh họa

Giả sử tập tin đầu vào có nội dung:

```
ZORO 110 90
GOING_MERRY 320 1500
LUFFY 120 95
USOPP 70 50
CHOPPER 90 55
NAMI 80 60
SANJI 105 88
ROBIN 85 65
```

Sau khi đọc tập tin và xử lý, các mảng/biến nhận giá trị như sau:

- `character`: { `ZORO`, `LUFFY`, `USOPP`, `CHOPPER`, `NAMI`, `SANJI`, `ROBIN` }
- `hp`: {110, 120, 70, 90, 80, 105, 85}
- `skill`: {90, 95, 50, 55, 60, 88, 65}
- `shipHP`: 320
- `repairCost`: 1500

4 Nhiệm vụ

4.1 Nhiệm vụ 1: Đánh giá tình trạng tàu Going Merry

Water 7 được biết đến là khu vực có các xưởng sửa chữa và đóng tàu nổi tiếng. Nhóm Mũ Rơm đến đây nhằm kiểm tra tình trạng của tàu Going Merry và ước tính chi phí sửa chữa. Nhiệm vụ của hàm trong phần này là xác định liệu tình trạng hiện tại của tàu có khiến chi phí sửa chữa dự kiến ban đầu cần được điều chỉnh hay không.

4.1.1 Mô tả hàm

- **Tên hàm**: `damageEvaluation`
- **Tham số đầu vào**:
  - `shipHP`: Chỉ số mức độ nguyên vẹn của tàu.
  - `repairCost`: Chi phí sửa chữa tàu.
- **Kết quả trả về**: Giá trị `repairCost`
- **Chức năng**:

  Ta định nghĩa khái niệm "Số hoàn hảo": Số hoàn hảo là số nguyên dương mà tổng các ước số dương của nó (không tính chính nó) bằng chính giá trị của số đó. Ví dụ $6 = 1 + 2 + 3$ là một số hoàn hảo Hàm thực hiện đánh giá dựa trên hai điều kiện sau:

  1. `shipHP` nhỏ hơn 455.
  2. Tổng các chữ số của `shipHP` là một **số hoàn hảo**.

  Nếu cả hai điều kiện đều thỏa mãn, tàu được xem là đang trong tình trạng hư hỏng nghiêm trọng và cần sửa chữa khẩn cấp; khi đó, `repairCost` được điều chỉnh tăng thêm 50%. Trong các trường hợp còn lại, chi phí sửa chữa không thay đổi.

4.1.2 Ví dụ minh họa

Ví dụ 4.1

Với dữ liệu nhập:

```
shipHP␣=␣411
repairCost␣=␣253
```

Xét 2 điều kiện:

1. $shipHP = 411 < 455$
2. $4 + 1 + 1 = 6$. 6 là một số hoàn hảo

Do cả 2 điều kiện đều thoả mãn:
$repairCost = 253 + 253 * 0.5 = 379,5 \rightarrow 380$

4.2 Nhiệm vụ 2: Mâu thuẫn giữa Luffy và Usopp

Sau khi đánh giá, các thợ đóng tàu kết luận tình trạng của tàu Going Merry là không thể sửa. Cả nhóm đều rất buồn. Luffy quyết định từ bỏ con tàu và tiếp tục hành trình với một con tàu mới trong khi Usopp cho rằng đó là một hành động phản bội và cảm thấy đồng cảm với Going Merry vì đều bị xem là "không có giá trị". Sự khác biệt trong quan điểm giữa Luffy và Usopp bắt đầu hình thành và gia tăng theo thời gian. Nhiệm vụ này mô phỏng quá trình tích tụ mâu thuẫn nội bộ thông qua việc xác định các sự kiện xảy ra và cập nhật chỉ số mâu thuẫn tương ứng.

4.2.1 Mô tả hàm

- **Tên hàm**: `conflictSimulation`
- **Tham số đầu vào**:
  - `character[][MAX_NAME]`: Mảng chứa tên của 7 nhân vật.
  - `hp[]`: Mảng chứa chỉ số bền bỉ tương ứng với từng nhân vật.
  - `skill[]`: Mảng chứa chỉ số năng lực tương ứng với từng nhân vật.
  - `shipHP`: Chỉ số mức độ nguyên vẹn của tàu.
  - `repairCost`: Chi phí sửa chữa tàu
- **Kết quả trả về**: Giá trị chỉ số mâu thuẫn sau khi quá trình mô phỏng kết thúc.

- **Chức năng**:
  Hàm thực hiện mô phỏng sự gia tăng mâu thuẫn theo từng bước thời gian. Tại mỗi bước, một sự kiện được xác định dựa trên mức độ áp lực hiện tại thông qua việc gia tăng chỉ số mâu thuẫn `conflictIndex`. Giá trị này là một số nguyên và được tính bằng công thức:

$$conflictIndex = skill\_Luffy - skill\_Usopp + \frac{repairCost}{100} + \frac{500 - shipHP}{50}$$

  Sự kiện sẽ xảy ra có ID sự kiện là:

$$id = conflictIndex \mod 6$$

  Các sự kiện có thể xảy ra được mô tả trong Bảng 1.

Bảng 1: Các sự kiện làm gia tăng chỉ số mâu thuẫn

ID sự kiện	Nội dung sự kiện	Gia tăng conflictIndex
0	Toàn bộ sự kiện sẽ xảy ra	Tăng thêm 255
1	Nhóm nhận được thông tin không khả quan về tình trạng của tàu Going Merry.	Tăng thêm 20
2	Luffy đưa ra quyết định từ bỏ Going Merry để tiếp tục hành trình.	Tăng thêm 50
3	Usopp phản đối quyết định của Luffy và cho rằng đó là hành vi phản bội.	Tăng thêm 70
4	Sự chênh lệch về năng lực và vai trò giữa Luffy và Usopp ngày càng thể hiện rõ.	Tăng thêm 90
5	Căng thẳng nội bộ đạt mức cao, mâu thuẫn tích tụ qua nhiều sự kiện liên tiếp.	Tăng thêm 100

Sau khi sự kiện kết thúc, quá trình này sẽ tiếp tục được xét cho đến khi `conflictIndex` đạt `255` trở lên hoặc trải qua đủ 10 sự kiện.

4.2.2 Ví dụ

Ví dụ 4.2

Giả sử:

$$skill_{\texttt{Luffy}} = 95, \quad skill_{\texttt{Usopp}} = 50, \quad repairCost = 1500, \quad shipHP = 350, \quad maxSteps = 10$$

Ở bước mô phỏng đầu tiên, ta có:

$$conflictIndex = 95 - 50 + \frac{1500}{100} + \frac{500 - 350}{50} = 63$$

Suy ra:

$$id = 63 \bmod 6 = 3$$

Theo Bảng 1, sự kiện có ID 3 xảy ra và chỉ số mâu thuẫn tăng thêm 70.
Do đó, sau khi áp dụng hệ quả của sự kiện ở bước này:

$$conflictIndex_{\text{mới}} = 63 + 70 = 133$$

Quá trình tiếp tục xét các bước tiếp theo cho đến khi `conflictIndex` đạt từ 255 trở lên hoặc đã trải qua đủ `10` sự kiện.

Điều này đồng nghĩa với chuyện chắc chắn mâu thuẫn giữa Usopp và Luffy đạt đỉnh điểm và dẫn đến một cuộc chiến (mô tả ở nhiệm vụ tiếp theo).

4.3 Nhiệm vụ 3: Cuộc đối đầu giữa Luffy và Usopp

Khi chỉ số mâu thuẫn giữa Luffy và Usopp đạt đến ngưỡng cao, Usopp quyết định rời nhóm và thách đấu Luffy để giành quyền sở hữu tàu Going Merry. Cuộc đối đầu này không nhằm phân thắng bại về sức mạnh, mà là sự va chạm giữa niềm tin, lòng tự trọng và tình cảm dành cho con tàu. Dù buộc phải chấp nhận trận đấu, Luffy vẫn giới hạn sự can thiệp của các thành viên khác nhằm tránh làm tổn hại đến sự gắn kết của cả nhóm. Nhiệm vụ này mô phỏng quá trình lựa chọn các thành viên phù hợp để giữ vai trò ổn định tinh thần và hạn chế ảnh hưởng tiêu cực của cuộc đối đầu đối với tập thể.

4.3.1 Mô tả hàm

- **Tên hàm**: `resolveDuel`

- **Tham số đầu vào**:
  - `character[][]`: Mảng chứa tên của 7 nhân vật.
  - `hp[]`: Mảng chứa chỉ số bền bỉ tương ứng với từng nhân vật.
  - `skill[]`: Mảng chứa chỉ số năng lực tương ứng với từng nhân vật.
  - `conflictIndex`: Chỉ số mâu thuẫn
  - `repairCost`: Chi phí sửa chữa tàu Going Merry.
  - `duel`: Mảng kết quả các nhân vật được chọn
- **Kết quả trả về**: Gán kết quả tập các nhân vật được chọn hỗ trợ gián tiếp cho Luffy vào tham số `duel`
- **Chức năng**:

  Trong nhiệm vụ này, các nhân vật được chia thành ba nhóm:
  - **Luffy**: Nhân vật chính của cuộc đối đầu.
  - **Usopp**: Nhân vật đối đầu với Luffy.
  - **Thành viên**: Các nhân vật còn lại trong nhóm (không bao gồm Luffy và Usopp).

  Với mỗi thành viên còn lại i, xác định:
  - Mức hỗ trợ: $support_i = skill_i$
  - Chi phí can thiệp: $cost_i = (hp_i \bmod 10) + 1$

  Sức chống chịu của Usopp trong cuộc đối đầu được xác định bởi:

$$U = skill_{\texttt{USOPP}} + \frac{conflictIndex}{20} + \frac{repairCost}{500}$$

  Luffy giành chiến thắng nếu tồn tại một tập các thành viên sao cho:

$$skill_{\texttt{LUFFY}} + \sum support_i \geq U$$

  Trong tất cả các tập thành viên thỏa điều kiện trên, hàm cần lựa chọn tập có **tổng chi phí can thiệp nhỏ nhất** tức là chỉ vì bất đắc dĩ mới phải chiến đấu và làm ít gây sát thương tinh thần nhất cho cả nhóm. Nếu có nhiều tập thỏa mãn, ưu tiên tập có số lượng ít hơn.

4.3.2 Ví dụ minh hoạ

Ví dụ 4.3

Giả sử:

$$skill_{\texttt{LUFFY}} = 95, \quad skill_{\texttt{USOPP}} = 50, \quad conflictIndex = 120, \quad repairCost = 1500$$

Ta có:

$$U = 50 + \frac{120}{20} + \frac{1500}{500} = 59$$

Giả sử hai thành viên được chọn có mức hỗ trợ lần lượt là $support_1 = 20$ và $support_2 = 10$, khi đó:

$$skill_{\texttt{LUFFY}} + support_1 + support_2 = 95 + 30 = 125 \geq 59$$

Do đó, tập này đảm bảo Luffy giành chiến thắng. Trong số các tập thỏa mãn điều kiện, tập có tổng chi phí can thiệp nhỏ nhất sẽ được chọn làm kết quả của hàm.

Sau khi Usopp thua trong cuộc chiến và quyết định rời đi, Luffy vẫn tặng Merry lại cho Usopp như một lời xin lỗi và công nhận tình cảm của cậu với con tàu, nhưng sau đó Merry đã "chết" trong sự tiếc nuối của cả băng, đặc biệt là Usopp.

4.4 Nhiệm vụ 4: Sự rời đi của Nico Robin

Khi vết rạn nứt nội bộ còn chưa kịp khép lại, Nico Robin bất ngờ rời nhóm trong im lặng. Chỉ sau đó, nhóm Mũ Rơm mới dần nhận ra rằng những biến cố tại Water 7 đều dẫn về một cái tên duy nhất: CP9 – lực lượng mật vụ bí mật của Chính phủ Thế giới.

Trước khi rời đi, Robin có bí mật để lại 1 thông điệp, khi giải mã thông điệp này, nhóm sẽ biết sự việc này liên quan đến CP9 và Chính phủ Thế giới. Để tránh bị phát hiện, Robin đã để thông điệp đúng trong hàng loạt các thông điệp vô nghĩa gây nhiễu khác. Một thông điệp chỉ được coi là đúng và hợp lệ khi trong thông điệp tồn tại cụm từ "CP9" hoặc "ENIESLOBBY".

4.4.1 Mô tả hàm

- **Tên hàm**: `decodeCP9Message`
- **Tham số đầu vào**:
  - `character[][MAX_NAME]`: Mảng chứa tên của 7 nhân vật.
  - `hp[]`: Mảng chứa chỉ số bền bỉ tương ứng với từng nhân vật.

- `skill[]`: Mảng chứa chỉ số năng lực tương ứng với từng nhân vật.
- `conflictIndex`: Chỉ số mâu thuẫn
- `repairCost`: Chi phí sửa chữa tàu.
- `cipherText`: Chuỗi ký tự biểu diễn thông điệp đã được mã hoá.
- `resultText`: Chuỗi kết quả sau giải mã

- **Kết quả trả về**: Chuỗi `plainText_TRUE` nếu là thông điệp hợp lệ hoặc chuỗi `plainText_FALSE` nếu kết quả không hợp lệ. Trong trường hợp không hợp lệ từ bước kiểm tra checksum, hàm trả về chuỗi rỗng.
- **Chức năng**:
  Thông điệp mã hoá có dạng `message#XY`, trong đó `XY` là hai chữ số checksum.

  1. **Kiểm tra checksum:** tính

  $$checksum = \left(\sum ASCII(message[i])\right) \bmod 100$$

  Nếu `checksum` khác `XY` thì thông điệp không hợp lệ.

  2. **Đảo theo block:** xác định

  $$key = (conflictIndex + repairCost) \bmod 26, \quad B = (key \bmod 5) + 4$$

  Chia `message` thành các block độ dài B (block cuối có thể ngắn hơn) và đảo ngược từng block.

  3. **Giải mã dịch vòng:** với từng ký tự:
     - `A-Z`: dịch ngược `key` theo cơ chế vòng trong bảng chữ cái.
     - `a-z`: dịch ngược `key` theo cơ chế vòng trong bảng chữ cái.
     - `0-9`: dịch ngược `key` mod `10` theo cơ chế vòng trong tập chữ số.
     - ký tự khác giữ nguyên.

4.4.2 Ví dụ minh hoạ

Ví dụ 4.4

Giả sử:

$$conflictIndex = 120, \quad repairCost = 1500$$

suy ra:

$$key = (120 + 1500) \bmod 26 = 8, \quad B = (8 \bmod 5) + 4 = 7$$

Chuỗi thông điệp mã hoá nhận được là:

$$cipherText = \text{``AQ 7XK\#96''}$$

- **Bước 1 (Checksum):** tách `message = ‘‘AQ 7XK’’` và checksum đọc được là `96`. Tính lại:

  $$(65 + 81 + 32 + 55 + 88 + 75) \bmod 100 = 396 \bmod 100 = 96$$

  Do đó thông điệp hợp lệ.
- **Bước 2 (Đảo theo block):** với $B = 7$, do độ dài `message` là 6 nên chỉ có một block. Đảo ngược block này:

  $$\text{``AQ 7XK''} \rightarrow \text{``KX7 QA''}$$

- **Bước 3 (Dịch vòng ngược):** với $key = 8$:
  - `K` dịch ngược $8 \rightarrow$ `C`
  - `X` dịch ngược $8 \rightarrow$ `P`
  - `7` dịch ngược ($key \bmod 10$) $= 8 \rightarrow$ `9`
  - ký tự khoảng trắng giữ nguyên
  - `Q` dịch ngược $8 \rightarrow$ `I`
  - `A` dịch ngược 8 (cơ chế vòng) $\rightarrow$ `S`

  Vì vậy:

  $$plainText = \text{``CP9 IS''}$$

Do thông điệp giải mã chứa chuỗi ‘‘CP9’’, đây là thông điệp hợp lệ, hàm trả về `"CP9 IS_TRUE"`

4.5 Nhiệm vụ 5: Phân tích bản đồ Enies Lobby

Sau khi xác nhận sự can thiệp của CP9, nhóm Mũ Rơm quyết định tiến vào Enies Lobby để giải cứu Robin. Enies Lobby là khu vực tư pháp được bảo vệ nghiêm ngặt, với nhiều khu vực nguy hiểm và chướng ngại khác nhau. Để chuẩn bị cho cuộc đột nhập, nhóm cần phân tích bản đồ khu vực này, được mô hình hoá dưới dạng một ma trận hai chiều.

Mỗi ô trong ma trận biểu diễn một khu vực của Enies Lobby, với ý nghĩa:

- `-1`: Khu vực không thể đi qua.
- `0`: Khu vực trống, an toàn.
- `1`: Khu vực có lính canh thường.
- `2`: Khu vực có tháp canh.
- `3`: Khu vực do CP9 kiểm soát.

4.5.1 Nhiệm vụ 5.1: Xác định ngưỡng nguy hiểm

- **Tên hàm**: `analyzeDangerLimit`
- **Tham số đầu vào**:
  - `grid[][]`: Ma trận hai chiều biểu diễn bản đồ Enies Lobby.
  - `rows`, `cols`: Số hàng và số cột của ma trận.
- **Kết quả trả về**: Giá trị nguyên `dangerLimit` (ngưỡng mức độ nguy hiểm cho phép).
- **Chức năng**:
  Hàm duyệt toàn bộ bản đồ để tính hai đại lượng sau:
  1. **maxRowSum**: Tổng mức độ nguy hiểm lớn nhất theo hàng. Với mỗi hàng, tính tổng các ô có giá trị không âm (không tính các ô `-1`), sau đó lấy giá trị lớn nhất trong các tổng hàng.
  2. **maxCell**: Giá trị lớn nhất của một ô trong toàn bộ ma trận (đại diện cho khu vực nguy hiểm nhất).

  Sau đó xác định ngưỡng nguy hiểm:

$$dangerLimit = maxRowSum + maxCell$$

4.5.2 Nhiệm vụ 5.2: Đánh giá lộ trình tiếp cận

- **Tên hàm**: `evaluateRoute`
- **Tham số đầu vào**:

  - `grid[][]`: Ma trận hai chiều biểu diễn bản đồ Enies Lobby.
  - `rows`, `cols`: Số hàng và số cột của ma trận.
  - `dangerLimit`: Ngưỡng mức độ nguy hiểm cho phép.

- **Kết quả trả về**: Giá trị logic (`true` hoặc `false`).
- **Chức năng**:

  Nhóm Mũ Rơm bắt đầu từ ô `(0,0)` và cần tiếp cận ô `(n-1,m-1)`. Nhóm chỉ được di chuyển sang phải hoặc xuống dưới.

  Hàm cần thực hiện:

  1. Kiểm tra xem có tồn tại ít nhất một lộ trình hợp lệ từ điểm bắt đầu đến điểm kết thúc mà không đi qua các ô có giá trị `-1` hay không.
  2. Nếu tồn tại lộ trình hợp lệ, tính tổng mức độ nguy hiểm trên lộ trình đó.
  3. So sánh tổng mức độ nguy hiểm với `dangerLimit` để đưa ra quyết định có nên tiến vào Enies Lobby hay không.

  Hàm trả về `true` nếu tồn tại lộ trình hợp lệ và tổng mức độ nguy hiểm không vượt quá `dangerLimit`; ngược lại trả về `false`.

4.6 Tạm kết

Thông qua các nhiệm vụ trong Bài tập lớn 1, sinh viên đã từng bước mô phỏng lại chuỗi sự kiện quan trọng trong arc Water 7, từ việc đánh giá tình trạng tàu Going Merry, sự tích tụ mâu thuẫn nội bộ, cho đến quá trình điều tra CP9 và phân tích bản đồ Enies Lobby. Các bài toán được thiết kế theo hướng tăng dần độ khó, yêu cầu sinh viên vận dụng linh hoạt các kiến thức lập trình C++ cơ bản như cấu trúc rẽ nhánh, vòng lặp, mảng một chiều, mảng hai chiều, xử lý chuỗi và thiết kế hàm.

Kết thúc BTL 1, nhóm Mũ Rơm đã đưa ra quyết định tiến vào Enies Lobby — một bước ngoặt mang tính đối đầu trực diện với Chính phủ Thế giới. Tuy nhiên, những gì chờ đợi phía trước không còn là các quyết định chiến thuật, mà là những trận chiến thực sự.

Chúc các bạn làm Bài tập lớn vui vẻ!!!

5 Yêu cầu

Để hoàn thành bài tập lớn này, sinh viên phải:

1. Đọc toàn bộ tập tin mô tả này.

2. Tải xuống tập tin initial.zip và giải nén nó. Sau khi giải nén, sinh viên sẽ nhận được các tập tin: main.cpp, main.h, water_seven.h, water_seven.cpp, và các file dữ liệu đọc mẫu. Sinh viên phải nộp 2 tập tin là water_seven.h và water_seven.cpp nên không được sửa đổi tập tin main.h khi chạy thử chương trình.
3. Sinh viên sử dụng câu lệnh sau để biên dịch:
   g++ -o main main.cpp water_seven.cpp -I . -std=c++11
   Sinh viên sử dụng câu lệnh sau để chạy chương trình:
   ./main opw_tc_01_input
   Các câu lệnh trên được dùng trong command prompt/terminal để biên dịch và chạy chương trình. Nếu sinh viên dùng IDE để chạy chương trình, sinh viên cần chú ý: thêm đầy đủ các tập tin vào project/workspace của IDE; thay đổi lệnh biên dịch của IDE cho phù hợp. IDE thường cung cấp các nút (button) cho việc biên dịch (Build) và chạy chương trình (Run). Khi nhấn Build IDE sẽ chạy một câu lệnh biên dịch tương ứng, thông thường câu lệnh chỉ biên dịch file main.cpp. Sinh viên cần tìm cách cấu hình trên IDE để thay đổi lệnh biên dịch: thêm file water_seven.cpp, thêm option -std=c++11, -I .
4. Chương trình sẽ được chấm trên nền tảng Unix. Nền tảng chấm và trình biên dịch của sinh viên có thể khác với nơi chấm thực tế. Nơi nộp bài trên LMS được cài đặt để giống với nơi chấm thực tế. Sinh viên phải chạy thử chương trình trên nơi nộp bài và phải sửa tất cả các lỗi xảy ra ở nơi nộp bài LMS để có đúng kết quả khi chấm thực tế.
5. Sửa đổi các file water_seven.h, water_seven.cpp để hoàn thành bài tập lớn này và đảm bảo hai yêu cầu sau:
   - Chỉ có một lệnh **include** trong tập tin `water_seven.h` là:
     #include "main.h"
   - Trong tập tin `water_seven.cpp`, chỉ có một lệnh include:
     #include "water_seven.h"
   - Ngoài hai include trên, không cho phép có bất kỳ lệnh **#include** nào khác trong các tập tin này (bao gồm cả các lệnh **#include** đã được comment)
   - Hiện thực các hàm được mô tả ở các nhiệm vụ trong BTL này.
6. Sinh viên được khuyến khích viết thêm các hàm để hoàn thành BTL này.

6 Nộp bài

Sinh viên chỉ nộp 2 tập tin: water_seven.h và water_seven.cpp, trước thời hạn được đưa ra trong đường dẫn "Assignment 1 - Submission". Có một số testcase đơn giản được sử dụng để kiểm tra bài làm của sinh viên nhằm đảm bảo rằng kết quả của sinh viên có thể biên dịch và chạy được. Sinh viên có thể nộp bài bao nhiêu lần tùy ý nhưng chỉ có bài nộp cuối cùng được tính điểm. Vì hệ thống không thể chịu tải khi quá nhiều sinh viên nộp bài cùng một lúc, vì vậy sinh viên nên nộp bài càng sớm càng tốt. Sinh viên sẽ tự chịu rủi ro nếu nộp bài sát hạn chót (trong vòng 1 tiếng cho đến hạn chót). Khi quá thời hạn nộp bài, hệ thống sẽ đóng nên sinh viên sẽ không thể nộp nữa. Bài nộp qua các phương thức khác đều không được chấp nhận.

7 Một số quy định khác

1. Sinh viên phải tự mình hoàn thành bài tập lớn này và phải ngăn không cho người khác đánh cắp kết quả của mình. Nếu không, sinh viên sẽ bị xử lý theo quy định của trường vì gian lận.
2. Mọi quyết định của giảng viên phụ trách bài tập lớn là quyết định cuối cùng.
3. Sinh viên không được cung cấp testcase sau khi chấm bài.
4. Nội dung Bài tập lớn sẽ được Harmony với một số câu hỏi trong bài Kiểm tra Cuối kỳ với nội dung tương tự.
5. Nếu sinh viên sử dụng mã nguồn từ các chương trình AI hoặc công cụ tự động tạo mã mà không hiểu rõ về cách hoạt động và ý nghĩa của mã đó, BTL sẽ bị chấm 0 điểm.

8 Harmony cho Bài tập lớn

Bài kiểm tra cuối kì của môn học sẽ có một số câu hỏi Harmony với nội dung của BTL.

Sinh viên phải giải quyết BTL bằng khả năng của chính mình. Nếu sinh viên gian lận trong BTL, sinh viên sẽ không thể trả lời câu hỏi Harmony và nhận điểm 0 cho BTL.

Sinh viên **phải** chú ý làm câu hỏi Harmony trong bài kiểm tra cuối kỳ. Các trường hợp không làm sẽ tính là 0 điểm cho BTL, và bị không đạt cho môn học. **Không chấp nhận giải thích và không có ngoại lệ.**

9 Gian lận

Bài tập lớn phải được sinh viên TỰ LÀM. Sinh viên sẽ bị coi là gian lận nếu:

- Có sự giống nhau bất thường giữa mã nguồn của các bài nộp. Trong trường hợp này, TẤT CẢ các bài nộp đều bị coi là gian lận. Do vậy sinh viên phải bảo vệ mã nguồn bài tập lớn của mình.
- Bài của sinh viên bị sinh viên khác nộp lên.
- Sinh viên không hiểu mã nguồn do chính mình viết, trừ những phần mã được cung cấp sẵn trong chương trình khởi tạo. Sinh viên có thể tham khảo từ bất kỳ nguồn tài liệu nào, tuy nhiên phải đảm bảo rằng mình hiểu rõ ý nghĩa của tất cả những dòng lệnh mà mình viết. Trong trường hợp không hiểu rõ mã nguồn của nơi mình tham khảo, sinh viên được đặc biệt cảnh báo là KHÔNG ĐƯỢC sử dụng mã nguồn này; thay vào đó nên sử dụng những gì đã được học để viết chương trình.
- Nộp nhầm bài của sinh viên khác trên tài khoản cá nhân của mình.
- Sử dụng mã nguồn từ các công cụ có khả năng tạo ra mã nguồn mà không hiểu ý nghĩa.

Trong trường hợp bị kết luận là gian lận, sinh viên sẽ bị điểm 0 cho toàn bộ môn học (không chỉ bài tập lớn).

KHÔNG CHẤP NHẬN BẤT KỲ GIẢI THÍCH NÀO VÀ KHÔNG CÓ BẤT KỲ NGOẠI LỆ NÀO!

Sau mỗi bài tập lớn được nộp, sẽ có một số sinh viên được gọi phỏng vấn ngẫu nhiên để chứng minh rằng bài tập lớn vừa được nộp là do chính mình làm.

10 Thay đổi so với phiên bản trước

Tài liệu