CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Bài Tập Lớn 1

THE LORD OF THE RINGS: THE FELLOWSHIP OF THE RING

Phiên bản 1.0

1. Giới thiệu

Chyện xưa kể rằng, Chúa tể bóng tối Sauron đã tạo ra chiếc nhẫn quyền lực nhằm thống trị vùng Middle-earth. Để chống lại, liên minh giữa người và tiên đã cùng hợp sức đánh lại quân đội của Sauron; nhưng cuối cùng Elendil, vua của loài người, đã bị Sauron giết chết. Hoàng tử Isildur lúc đó đã kịp nhặt thanh kiếm gãy của người cha để chặt đứt ngón tay Sauron, chiếc nhẫn tách khỏi hắn và toàn bộ đội quân ma quỉ bị chế ngự. Dù vậy linh hồn của Sauron đã nhập vào chiếc nhẫn và hắn chỉ bị đánh bại hoàn toàn khi chiếc nhẫn bị phá huỷ. Isildur cầm lấy chiếc nhẫn và không kháng cự được ham muốn giữ nó cho riêng mình, ông ta đã không phá huỷ nó. Sau đó ông ta đã bị bọn quái vật phục kích và giết chết, còn chiếc nhẫn rơi xuống sông. Hàng ngàn năm sau, chiếc nhẫn được một kẻ tên là Gollum tìm thấy và giữ trong 500 năm, nó giúp hắn ta có được một cuộc sống "dài một cách quái dị". Nhưng cuối cùng chiếc nhẫn cũng rời xa hắn và rơi vào tay một người Hobbit tên là Bilbo Baggins.

Vào sinh nhật thứ 111, Bilbo giao lại chiếc nhẫn cho người cháu Frodo Baggins. Pháp sư Gandalf nhanh chóng nhận ra đó chính là Chiếc nhẫn quyền lực. Theo lời khuyên của Gandalf, Frodo mang chiếc nhẫn đi đến nơi an toàn. Trải qua bao nhiêu gian khó, Frodo cùng 2 người bạn Hobbit của mình với sự giúp sức của hiệp sĩ Strider và tiên nữ Arwen, chiếc nhẫn đã được mang đến nơi ở của loài tiên tại Rivendell, nơi được cai quản bởi Elrond. Tại đây, các cuộc thảo luận đã nổ ra gay gắt xung quanh số phận của chiếc nhẫn và tương lai của thế giới. Cuối cùng, mọi người đều nhất chí chiếc nhẫn cần bị (và chỉ có thể) phá huỷ trong ngọn lửa của núi Doom, nơi nó được rèn ra. Núi Doom là ngọn núi lửa ở Mordor và nằm sát bên pháo đài Barad-dur của Sauron; do đó cuộc hành trình phá huỷ chiếc nhẫn sẽ rất khó khăn và nguy hiểm. Frodo đã lên tiếng nhận nhiệm vụ mang chiếc nhẫn tới Mordor. Một Hiệp hội bảo vệ nhẫn cần phải được thành lập gồm các chiến binh để hộ tống Frodo trên hành trình này.

Thể theo một lời tiên tri ghi trong một cuốn sách cuả mình, Elrond ra lệnh cho các hiệp sĩ thi tài với nhau để được chọn tham gia hiệp hội. Trong cuộc thi tài, mỗi hiệp sĩ sẽ gặp các hiệp sĩ, tiên, phù thuỷ khác hoặc các chiến binh bóng tối, tay sai do chúa tể bóng tối Sauron cài vào. Tất cả các nhân vật đều được cấp một thẻ bài đặc biệt (bởi chúa nhẫn) có chứa một chữ số, gọi là ringsign. Các hiệp sĩ sẽ thu thập các ringsign từ các nhân vật mà mình thi tài. Số các ringsign mà các hiệp sĩ thu thập ở cuộc thi tài của mình sẽ tuỳ thuộc vào chiến tích mà họ đạt được trong cuộc thi. Nếu dãy số tạo bởi các ringsign của một hiệp sĩ có được sau hành trình tạo thành một con số đặc biệt được nhắc đến trong lời tiên tri, hiệp sĩ đó sẽ được chọn vào hiệp hội.

2. Yêu cầu

Trong bài tập lớn này, sinh viên sẽ được cung cấp một file chứa dữ liệu nhập, bao gồm thông số của một hiệp sĩ tham dự cuộc thi tài, cùng với các sự kiện xảy ra với hiệp sĩ này trong suốt hành trình thi thố tài năng. Chương trình sẽ xuất ra màn hình dãy số tạo bởi các ringsign mà hiệp sĩ này có được sau cuộc thi tài. Các dữ liệu nhập xuất mà sinh viên phải xử lý đều được biểu diễn dưới dạng *danh sách liên kết* (linked list). Chi tiết cụ thể công việc sinh viên phải làm sẽ mô tả trong phần 4.

3. Dữ liệu nhập

Dữ liệu nhập của chương trình được chứa trong file mang tên `input.txt`. File này sẽ chứa các thông tin như sau:

```
172 4 0
1
```

thông tin về hiệp sĩ dự thi tài

thông tin về các sự kiện mà hiệp sĩ trải qua ở cuộc thi tài

Như vậy file nhập sẽ bao gồm ít nhất hai dòng. Dòng đầu tiên sẽ mô tả thông tin về hiệp sĩ dự hành trình, có định dạng như sau:

HP level ringsign

Trong đó:

- *HP*: Chỉ số sức khoẻ của hiệp sĩ là một số nguyên có giá trị 1 đến 999. Chỉ số này cũng sẽ là chỉ số *MaxHP* biểu diễn sức mạnh tối đa của hiệp sĩ.
- *level*: đẳng cấp của hiệp sĩ, là một số nguyên có giá trị từ 1 đến 10.
- *ringsign*: chữ số trên ringsign ban đầu hiệp sĩ được phát, là một số nguyên có giá trị từ 0 đến 9.

Kể từ dòng thứ hai của file nhập sẽ mô tả các sự kiện mà hiệp sĩ gặp phải khi tham gia thi tài. Mỗi sự kiện sẽ được mô tả bằng một giá trị số, gọi là *mã sự kiện*. Ý nghĩa tương ứng của từng sự kiện được mô tả trong Bảng 1. Số sự kiện là không cố định, có thể thay đổi tuỳ theo test case. Một sự kiện có thể xảy ra nhiều lần. Sẽ có tối đa 2^{100} sự kiện xảy ra. Nếu số sự kiện nhiều, các sự kiện có thể trình bày thành nhiều dòng.

Ví dụ 1: Với dữ liệu nhập là

```
172 4 0
```

23 42

thì hiệp sĩ sẽ có *HP* là 172, *level* là 4, ringsign ban đầu hiệp sĩ có in chữ số 0. Mặt khác MaxHP của hiệp sĩ cũng là 172 (nghĩa là *HP* của hiệp sĩ sẽ không bao giờ vượt quá 172 trong suốt quá trình thi tài).

Trong quá trình dự thi tài, hiệp sĩ sẽ gặp các sự kiện sau:

Sự kiện **1**: gặp Ma nhẫn.

Sự kiện **2**: gặp Gollum.

Bảng 1 – Các sự kiện xảy ra trong hành trình

Mã sự kiện	Ý nghĩa
0	Thời gian thi thố chấm dứt đột ngột
1X	Gặp quái vật Uruk-hai
2X	Gặp Ma nhẫn (Ringrwaiths)
3X	Gặp Strider
4X	Gặp Gollum
5X	Gặp chỉ huy bóng tối Lurtz
6X	Gặp người lùn Gimli
7	Gặp tiên nữ Arwen
8	Gặp nữ vương Galadriel
9X	Gặp giáo chủ Saruman

4. Hiện thực chương trình

Sinh viên sẽ hiện thực một hàm *combat* có prototype như sau:

ringsignList combat (knight& theKnight, eventList* pEvent)*

Trong đó *ringsignList* sẽ là một danh sách liên kết chứa các ringsign mà hiệp sĩ có được sau khi kết thúc hành trình, được khai báo như sau:

```
struct ringsignList {
        int nRingsign;                  //ringsign
        ringsignList* pNext;
}
```

Danh sách này gọi là Danh sách **Mật Mã Elrond (Elrond's Code - EC)** hay **Danh sách EC** của hiệp sĩ. Cách xây dựng danh sách EC sẽ được mô tả trong phần 5. **Mã (Code)** của một danh sách EC là một số nguyên được tạo ra bằng cách ghép các chữ số trên các ringsign của danh sách từ phần tử đầu tiên đến phần tử cuối cùng.

Ví dụ 2: Nếu một danh sách EC có 3 ringsign lần lượt chứa các chữ số 0,7 và 9, danh sách đó sẽ được ký hiệu là **'079'**, Mã của danh sách là số nguyên 79.

Thông tin về hiệp sĩ được truyền qua biến tham khảo *theKnight* thuộc cấu trúc *knight* được khai báo như sau:

```
struct knight {
    int HP;
    int level;
    int nInitRingsign;
}
```

Các thành phần *HP*, *level*, *nInitRingsign* của *struct knight* tương ứng với các thông số *HP*, *level* và *ringsign* của hiệp sĩ.

Thông số *pEvent* là một con trỏ trỏ đến danh sách liên kết của các sự kiện được đọc từ file input, được định nghĩa như sau:

```
struct evenList {
    int nEvenCode;
    evenList* pNext;
}
```

5. Xây dựng danh sách EC

Khởi tạo, danh sách EC sẽ chỉ có đúng một phần tử, giá trị của phần tử này chính là giá trị của ringsign ban đầu của hiệp sĩ.

Ví dụ 3: Với dữ liệu nhập là

```
172 4 5
0
```

(Hiệp sĩ kết thúc cuộc thi tài của mình ngay sự kiện đầu tiên.) Kết quả của danh sách EC trả về là **'5'** (Danh sách chỉ có một phần tử có giá trị là 5).

Trong quá trình tham dự cuộc thi, danh sách EC của hiệp sĩ có thể thay đổi dựa theo các sự kiện mà hiệp sĩ này gặp phải. Nếu trong bất kỳ trường hợp nào mà danh sách EC của hiệp sĩ rỗng, hàm *combat* sẽ kết thúc và trả về NULL.

Các sự kiện cụ thể như sau:

S1) Nếu gặp sự kiện có mã là 0, hành trình kết thúc, hàm *combat* sẽ trả về danh sách EC hiện hành của hiệp sĩ. (xem ví dụ 3).

S2) Nếu gặp sự kiện có mã dạng 1X,2X,3X,4X,5X,6X,9X hiệp sĩ phải giao tranh với đối thủ tương ứng. Mỗi đối thủ cũng sẽ có đẳng cấp *levelO* và ringsign *RingsignO* riêng. Chữ số in trên RingsignO của đối thủ chính là chỉ số X trên mã sự kiện tương ứng. Nếu gặp đối thủ ở sự kiện thứ **i** (Sự kiện đầu tiên i=1), *levelO* tương ứng của đối thủ sẽ được tính như sau:

$$b = \mathbf{i}\%10 \quad (1)$$

$$levelO = \mathbf{i} > 6 ? (b > 5 ? b : 5) : b \quad (2)$$

Nếu *level* của hiệp sĩ cao hơn *levelO* của đối thủ, hiệp sĩ chiến thắng. Khi chiến thắng, hiệp sĩ sẽ đoạt ringsign của của đối thủ và **thêm vào cuối danh sách EC của mình**, ngoại trừ một số trường hợp đặc biệt sẽ có mô tả riêng bên dưới.

Nếu *level* của hiệp sĩ bằng *levelO*, trận đấu hoà.

Nếu *level* của hiệp sĩ nhỏ hơn *levelO* của đối thủ, hiệp sĩ sẽ thua. *HP* của hiệp sĩ sẽ được tính lại như sau:

$$HP = HP - damage \quad (3)$$

Trong đó *damage* sẽ được tính như sau:

$$damage = baseDamage * levelO * 10 \quad (4)$$

với *baseDamage* sẽ tuỳ thuộc vào đối thủ, được mô tả ở Bảng 2.

Bảng 2 – Chỉ số baseDamage của các đối thủ

Đối thủ	baseDamage
Quái vật Uruk-hai	1
Ma nhẫn	1.8
Strider	4.5
Gollum	8.2
Lurtz	7.5
Gimli	9
Saruman	0.1

Lưu ý rằng *HP* sẽ luôn là số nguyên khi tính bằng công thức (3). Nếu *HP* nhỏ hơn hoặc bằng 0 sau khi tính bằng công thức (3), hiệp sĩ bị thương quá nặng và không thể tiếp tục dự hành trình, khi đó kết quả của hàm *combat* sẽ là NULL.

Ví dụ 4: Với dữ liệu nhập là

```
172 4 0
32 31
```

Như vậy ban đầu danh sách EC khởi tạo là **'0'**. Sau đó hiệp sĩ gặp Strider (mã sự kiện 3X) ở sự kiện 1 (có *levelO* là 1 và *RingsignO* là 2) và chiến thắng (có *level* cao hơn *levelO*), danh sách EC trở thành **'02'** do hiệp sĩ đoạt ringsign của đối thủ là thêm vào cuối danh sách EC. Sau đó, hiệp sĩ tiếp tục đánh thắng hiệp sĩ Strider ở sự kiện thứ hai (có *levelO* là 2 và *RingsignO* là 1), kết quả của danh sách EC trả về là **'021'**.

S3) Tiên nữ Arwen được bầu chọn là Nữ thần Tình yêu của toàn nhân loại, vì vậy nàng có quyền ban thưởng và sai khiến các hiệp sĩ. Nếu hiệp sĩ gặp Arwen (mã sự kiện là 7), nàng sẽ thu lại bộ ringsign của hiệp sĩ và cấp cho hiệp sĩ một bộ ringsign khác để tạo thành một danh sách EC mới sao cho Mã tạo ra từ danh sách mới (xem ví dụ 2) sẽ **lớn hơn Mã từ danh sách EC cũ đúng 1 đơn vị**.

Ví dụ 5: Với dữ liệu nhập là

```
812 10 0
11 12 13 14 15 16 17 18 19 7
```

Sau khi đánh thắng 9 quái vật Uruk-hai đầu tiên, danh sách EC của hiệp sĩ sẽ là **'0123456789'** (tức là danh sách 0,1,2,3,4,5,6,7,8,9 và mã là (số có nghĩa) 123456789), hiệp sĩ sẽ gặp tiên nữ Arwen. Tiên nữ sẽ trao cho hiệp sĩ một bộ ringsign mới tạo thành dãy số **'123456790'** (tức là danh sách 1,2,3,4,5,6,7,9,0 có mã là 123456790 lớn hơn mã 123456789 một đơn vị).

S4) Galadriel là một nữ vương có quyền phép ghê gớm. Hiệp sĩ khi gặp Galadriel sẽ được phục hồi *HP* trở về *MaxHP* ban đầu, tuy nhiên sẽ mất đi ringsign ở cuối danh sách EC. Nếu *HP* của hiệp sĩ đang ở mức *MaxHP*, Galadriel sẽ không thực hiện giao dịch với hiệp sĩ.

Ví dụ 6: Với dữ liệu nhập là

```
200 2 0
28 20 29 8
```

Sau khi đánh thắng Ma nhẫn đầu tiên và đánh hoà với ma nhẫn thứ hai, danh sách EC của hiệp sĩ bây giờ sẽ là **'08'**. Ở trận thứ ba hiệp sĩ thua trận và HP bị giảm còn 200 – 1.8*3*10 = 146. Khi gặp Galadriel, HP của hiệp sĩ cũng sẽ được phục hồi thành 200, tuy nhiên Galadriel sẽ lấy đi ringsign cuối, nên danh sách EC cuối cùng của hiệp sĩ sẽ là **'0'**.

Các điều kiện đặc biệt thêm vào:

Trong sự kiện S2, ngoài các ảnh hưởng của sự kiện như đã mô tả trong sự kiện S2, các điều kiện đặc biệt sau đây có thể xảy ra thêm:

A1) Nếu hiệp sĩ đánh thua Gollum, Gollum sẽ lấy đi **ringsign trùng cuối cùng tính từ đầu danh sách EC** của hiệp sĩ. Nếu không có ringsign nào như vậy tồn tại trong danh sách EC của hiệp sĩ, Gollum sẽ không làm gì cả.

Ví dụ 7: Với dữ liệu nhập là

```
400 3 3
```

17 13 10 43

Sau khi đánh thắng hai quái vật Uruk-hai đầu tiên, danh sách EC của hiệp sĩ sẽ là **'373'**. Sau khi đánh hoà ở sự kiện thứ ba, danh sách EC không đổi. Đến sự kiện thứ tư hiệp sĩ đánh thua Gollum – lúc này đang có level 4 và chữ số in trên ringsign là **3**. Gollum sẽ lấy đi ringsign số **3** cuối cùng trên danh sách EC, danh sách EC còn lại (cũng chính là của kết quả trả về) sẽ là **'37'**.

A2) Nếu hiệp sĩ đánh thua Lurtz, hiệp sĩ sẽ bị cướp đi **3 ringsign đầu danh sách EC của mình**.

Ví dụ 8: Với dữ liệu nhập là

```
812 4 0
```

13 14 19 18 55

Sau khi đánh thắng 3 quái vật Uruk-hai đầu tiên, danh sách EC của hiệp sĩ sẽ là **'0349'**, tiếp đó hiệp sĩ đánh hoà với quái vật Uruk-hai thứ tư. Đến sự kiện thứ năm, hiệp sĩ đánh thua Lurtz và bị mất 3 ringsign đầu danh sách, do vậy danh sách EC kết quả trả về sẽ là **'9'**.

A3) Nếu hiệp sĩ đánh thắng Saruman, Saruman sẽ đảo ngược thứ tự danh sách EC của hiệp sĩ. Ngược lại nếu hiệp sĩ đánh thua Saruman, Saruman sẽ cuỗm của hiệp sĩ **tất cả** các ringsign có chữ số trùng với số in trên ringsign của Saruman.

Ví dụ 9: Với dữ liệu nhập là

```
200 8 0
```

27 24 93

Sau khi đánh thắng 2 Ma nhẫn đầu tiên, danh sách EC của hiệp sĩ bây giờ sẽ là **'074'**. Ở trận thứ ba hiệp sĩ đánh thắng Saruman nên Saruman sẽ đảo ngược danh sách EC của hiệp sĩ thành **'470'**.

Ví dụ 10: Với dữ liệu nhập là

```
200 2 3
```

13 13 93 24 18 45

Sau khi đánh thắng quái vật Uruk-hai đầu tiên và hoà với quái vật Uruk-hai thứ hai, danh sách EC của hiệp sĩ bây giờ sẽ là **'33'**. Ở trận thứ ba hiệp sĩ đánh thua Saruman nên bị Saruman cuỗm tất cả ringsign có giá trị là **3**. Danh sách EC của hiệp sĩ lúc này là rỗng nên hàm *combat* sẽ kết thúc và trả về NULL.

A4) Nếu *HP* ban đầu của hiệp sĩ là 777, hiệp sĩ này là hiện thân của Isildur, một người tham lam; ai lấy của Isildur một thứ gì sẽ bị mất gấp mười. Chính vì vậy không ai dám thu ringsign của Isildur. Isildur sẽ không bao giờ bị mất ringsign nào kể cả khi đánh thua Lurtz và Saruman và Gollum và khi giao dịch với Galadriel.

A5) Nếu *HP* ban đầu của hiệp sĩ là 888, hiệp sĩ chính là vị tiên Legolas, một nhân vật xuất chúng. Legolas không giao đấu với các Strider và Gimli. Galadriel không bao giờ thu ringsign của Legolas khi tăng HP cho chàng.

Ví dụ 11: Với dữ liệu nhập là

```
888 3 0
```

37 34 38 63

Legolas không giao đấu với ba hiệp sĩ đầu tiên vì họ chính là hiệp sĩ Strider. Đến sự kiện thứ tư người lùn Gimli cũng không giao đấu với Legolas. Kết quả danh sách EC cuối cùng của Legolas vẫn là **'0'**.

A6) Nếu *HP* ban đầu của hiệp sĩ là 999, hiệp sĩ đó chính là Pháp sư Gandalf. Chỉ có các Strider, Gollum, Lurtz và Saruman là dám giao chiến với Gandalf, các đối thủ khác sẽ bỏ qua không giao chiến với ông. Arwen và Galadriel vẫn giao dịch với ông bình thường.

Ví dụ 12: Với dữ liệu nhập là

```
999 8 0
```

7 27 14

Gandalf gặp Arwen ở sự kiện đầu tiên nên sẽ có được danh sách EC mới **'1'**. Sau đó các Ma nhẫn và quái vật Uruk-hai ở sự kiện thứ hai và thứ ba đều không giao chiến với ông. Danh sách EC sau cùng trả về là **'1'**.

A7) (*Bonus*) Nếu danh sách EC sau cùng của hiệp sĩ là đối xứng, hiệp sĩ này sẽ được chọn tham gia hiệp hội bảo vệ nhẫn. Trong trường hợp này chương trình sẽ xuất một giá trị chuỗi đặc biệt ra màn hình (Xem phần 6 để biết thêm chi tiết).

Ví dụ 13: Nếu danh sách EC của hiệp sĩ là **'12345678909876543 21'** thì hiệp sĩ này được chọn.

6. Cách dịch và thực thi chương trình

Sinh viên download file *assigment1.zip* từ trang Web của môn học. Khi giải nén file này, sẽ có được các file sau:

`input.txt`	Một file input ví dụ.
`main.cpp`	Chương trình chính
`tournament.cpp`	Chương trình hiện thực bởi sinh viên
`defs.h`	File định nghĩa các cấu trúc và hàm dùng chung
`Assignmen1.pdf`	File mô tả nội dung bài tập lớn

File `input.txt` là một file nhập mẫu như được mô tả ở phần 3. File `main.cpp` là chương trình khởi tạo, bao gồm các hàm viết sẵn như sau:

- *main*(): chương trình chính sẽ thực thi
- *readFile*(): hàm đọc file input
- *display*() : hàm xuất dữ liệu ra màn hình.

Trong hàm này sẽ có lời gọi đến một hàm *checkPalindrome* để kiểm tra danh sách EC có đối xứng không. Hàm *checkPalindrome* được khai báo trong file `defs.h` và được hiện thực trong file tournament.cpp. Sinh viên cần viết lại hàm này nếu muốn có điểm bonus được mô tả trong phần S7.

Lưu ý rằng ***sinh viên không được phép thay đổi file `main.cpp` và `defs.h` khi hiện thực chương trình****. Ngoài ra, các hàm do sinh viên viết không được xuất bất kỳ dữ liệu nào ra màn hình khi thực thi.*

Để dịch và thực thi chương trình, sinh viên chứa cả 3 files `main.cpp`, `tournament.cpp` và `defs.h` trong cùng một thư mục; sau đó chỉ cần dịch và thực thi **duy nhất** file `main.cpp`. Mọi công việc cần phải làm sẽ được hiện thực trong file `tournament.cpp`, tuy nhiên không cần dịch và thực thi file này.

Ví dụ 14: Để dịch và thực thi chương trình trên môi trường linux, có thể sử dụng lệnh sau:

```
$ make
$ ./assignment input.txt
```

Lưu ý:

L1) Sinh viên phải ***sử dụng danh sách liên kết để lập trình bài tập, KHÔNG DÙNG MẢNG HOẶC CHUỖI***.

CÁC BÀI LÀM KHÔNG DÙNG DANH SÁCH LIÊN KẾT SẼ KHÔNG CÓ ĐIỂM.

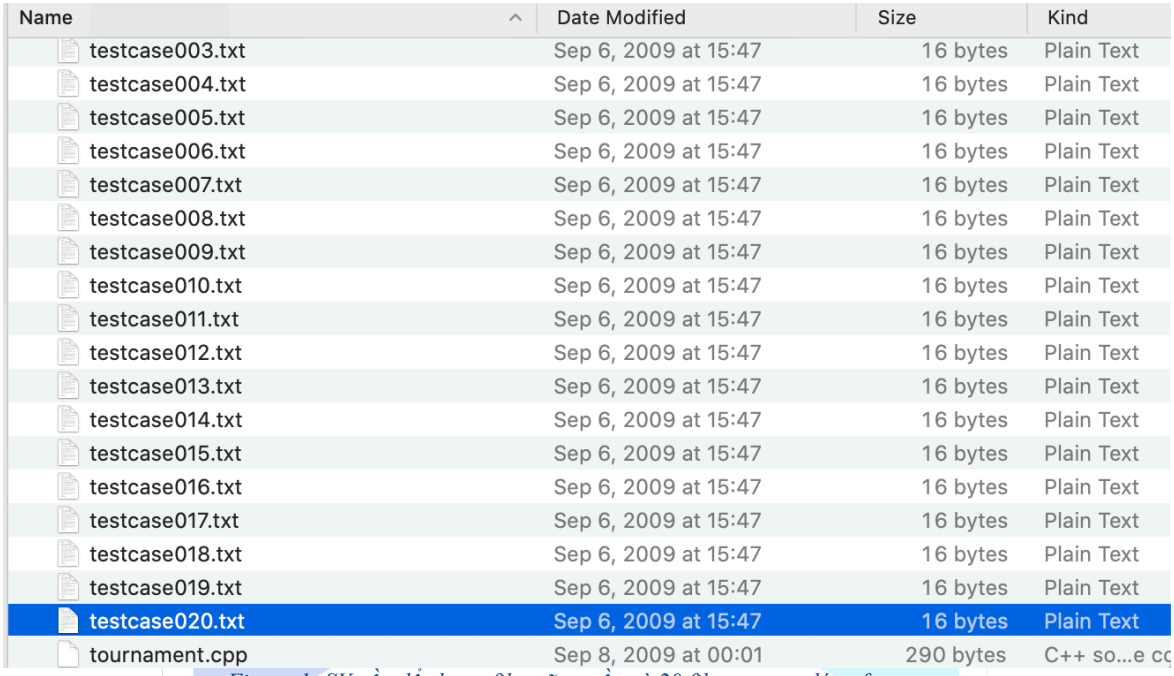
L2) Danh sách các sự kiện là một con trỏ *pEvent* chỉ đến phần tử đầu tiên của danh sách. Sinh viên phải tự viết đoạn mã để truy xuất trực tiếp lần lượt các phần tử của danh sách này khi xử lý các sự kiện.

L3) Danh sách Mật mã Elrond (EC list) kết quả của hàm *combat* là một con trỏ chỉ đến phần tử đầu tiên của danh sách ringsign (có kiểu con trỏ của *ringsignList*). Sinh viên phải viết **các thao tác trực tiếp** trên danh sách này trong quá trình cập nhật danh sách. Sinh viên có thể viết các đoạn mã dùng chung thành các hàm để khi cần là gọi lại. Không dùng List ADT như trên lớp lý thuyết.

6. Nộp bài

Sinh viên cần hiện thực và kiểm thử chương trình của mình kỹ lưỡng trước khi nộp bài. Để kiểm thử chương trình, Sinh viên cần soạn và chạy thử với tối thiểu 20 testcase, cover toàn bộ các trường hợp xảy ra của dữ liệu đầu vào.

Khi nộp bài, sinh viên sử dụng account đã được cấp phát trên hệ thống BKCSELAB.NET để kiểm tra chương trình có chạy được không và nộp bài. Sinh viên zip file mã nguồn đã hiện thực bao gồm file **storeman.cpp và 20 testcase với định dạng testcase001.txt thành 1 file duy nhất có tên với định dạng MSSV.zip (Tất cả các file trong file zip khác file tournament.cpp sẽ bị tự động xoá khi chấm bài).** File được zip phải là file chương trình gốc, sinh viên không được nén file khi nộp bài. Sau khi kiểm tra và nộp bài trên BKCSELAB.NET, sinh viên phải nộp 1 bản trên BKEL, 2 phiên bản nộp trên 2 hệ thống này này phải được đảm bảo cùng là một phiên bản.



Name	Date Modified	Size	Kind
testcase003.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase004.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase005.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase006.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase007.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase008.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase009.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase010.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase011.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase012.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase013.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase014.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase015.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase016.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase017.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase018.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase019.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
testcase020.txt	Sep 6, 2009 at 15:47	16 bytes	Plain Text
tournament.cpp	Sep 8, 2009 at 00:01	290 bytes	C++ so...e cc

Figure 1: SV cần để chung file mã nguồn và 20 file testcase đúng format

7. Thời hạn nộp bài

Thời hạn chót để nộp bài là **7h00 ngày thứ Hai 24/08/2020**. Sinh viên phải dùng account trên BKCSELAB và BKEL để nộp bài. KHÔNG nhận bài được gửi qua mail hoặc bất kỳ hình thức nào khác. Bài nộp trễ sẽ KHÔNG được nhận.

8. Xử lý 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.*
- 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.*

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 - 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.