# Debug & Refactoring

## 4 - Sửa lỗi và cấu trúc lại
## https://github.com/tqlong/advprogram

# Nội dung

- **Lỗi logic & Sửa lỗi (debugging)**
- Cấu trúc lại (refactoring)
- Tiếp tục cấu trúc và tối ưu (optimization)
- Kỹ thuật
  - Thư viện **string**
  - Truyền tham số bằng giá trị, tham chiếu, tham chiếu hằng
  - Từ khóa **static, const**

# Lỗi logic

Chạy thử Hangman: Khi người chơi liên tiếp chọn 1 kí tự đã đoán đúng

- Biến ***correctGuess*** tăng dần
- Khi biến ***correctGuess*** bằng độ dài của từ → người chơi thắng cuộc
  - Không cần đoán các ký tự *khác* của từ

*Lỗi logic*: chương trình hoạt động không như mong muốn

# Lỗi logic

- Thường khó phát hiện
  - Chạy thử
  - Chuyên nghiệp: unit tests, test cases (input - output) cho từng hàm
- Tránh lỗi logic
  - Tìm hiểu kỹ các yêu cầu của chương trình
  - Tận dụng các thư viện nổi tiếng đã được kiểm thử kỹ càng
  - Càng lập trình nhiều, càng có kinh nghiệm tránh lỗi

# Sửa lỗi

*Chỉ tăng biến **correctGuess** khi **ch** không nằm trong các ký tự đã đoán đúng trước đó*

`!isCharInWord(ch, correctChars)`

```
if (isCharInWord(ch, word)) {
    if (!isCharInWord(ch, correctChars)) {
        correctChars += ch;
        correctGuess ++;
        secretWord = updateSecretWord(ch, secretWord, word);
    }
} else {
    incorrectChars += ch;
    incorrectGuess ++;
}
```

# string: sử dụng tiện ích tìm kiếm

- Hàm **isCharInWord()** đã hoạt động tốt
- Thao tác tìm kiếm là thao tác phổ biến
- Lớp **string** đã cài đặt sẵn nhiều tiện ích
  - http://www.cplusplus.com/reference/string/string/
  - http://www.cplusplus.com/reference/string/string/find_first_of/

```
std::string str ("Please, replace the vowels in this sentence by asterisks.");
std::size_t found = str.find_first_of("aeiou");
while (found!=std::string::npos)
{
  str[found]='*';
  found=str.find_first_of("aeiou",found+1);
}

std::cout << str << '\n';
```

# Hàm isCharInWord() mới

```
bool isCharInWord(char ch, string word)
{
    return (word.find_first_of(ch) != string::npos);
}
```

- Mã lệnh “sạch” hơn, rõ nghĩa hơn
- Tận dụng thư viện <string>
  - Sử dụng lại công sức của rất nhiều người đi trước
  - Thư viện đã được kiểm thử, chạy thử, sửa lỗi rất kỹ
  - *Isaac Newton: “Tôi nhìn thấy được xa hơn bởi vì tôi đứng trên vai những người khổng lồ”*

# Sửa phần hiển thị

- Phần hiển thị cũng chưa được như mong muốn: Người chơi vẫn nhìn thấy các hình giá treo cũ
- Mẹo:
  - *In nhiều dòng trắng* để xoá màn hình, đẩy hình vẽ xuống dưới màn hình
  - Tạo hiệu ứng giao diện “cố định”, thân người “mọc” dần ra

```
cout << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl;
cout << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl;
cout << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl << endl;

cout << getDrawing(incorrectGuess);
```

# Vẫn còn lỗi :-(

- Nếu từ có nhiều ký tự giống nhau (ví dụ: trousers), ***correctGuess*** chỉ tăng lên 1
  - Chương trình hoạt động không như mong muốn
- Sửa lỗi: ***correctGuess*** cần tăng lên bằng số lượng ký tự đoán đúng trong từ
  - Cần tìm cách đếm số ký tự đoán đúng mỗi lượt chơi (lượt đoán)

# Đếm số ký tự đoán đúng

Nhận xét: Hàm **updateSecretWord()** duyệt qua các ký tự đoán đúng

- Thêm chức năng đếm số ký tự đoán đúng vào hàm
- Bản chất là *thêm đầu ra* cho hàm
  - **secretWord**
  - số đếm ký tự đoán đúng (**count**)

# Truyền tham số bằng tham chiếu

- Tham số **secretWord** thay đổi trong hàm
- Sử dụng tham chiếu khi truyền **secretWord**
  - Mọi thay đổi bên trong hàm sẽ làm thay đổi biến được truyền vào hàm
- Hàm **updateSecretWord()** trả về số nguyên

```
int updateSecretWord(string& secretWord, char ch, string word)
{
    int len = word.length();
    int count = 0;
    for (int i = 0; i < len; ++i) {
        if (word[i] == ch) {
            secretWord[i] = ch;
            count++;
        }
    }
    return count;
}
```

```
if (isCharInWord(ch, word)) {
    if (!isCharInWord(ch, correctChars)) {
        int numberOfOccurences =
         updateSecretWord(secretWord, ch, word);
        correctChars += ch;
        correctGuess += numberOfOccurences;
    }
} else {
    incorrectChars += ch;
    incorrectGuess ++;
}
```

# Hangman 2.0

```
int main()
{
    srand(time(0));

    char ch;
    string word = chooseWord();
    string secretWord = string(word.length(), '-');
    int incorrectGuess = 0, correctGuess = 0;
    string incorrectChars = "", correctChars = "";
    const int MAX_GUESSES = 7;

    while (true) {
        cout << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl;

        cout << getDrawing(incorrectGuess)
             << endl << "Current word: " << secretWord
             << endl << "Correct guesses: " << correctChars
             << endl << "Incorrect guesses: " << incorrectChars
             << endl << "Choose a character: ";


        cin >> ch;
```

```
        if (isCharInWord(ch, word)) {
            if (!isCharInWord(ch, correctChars)) {
                int numberOfOccurences =
                  updateSecretWord(secretWord, ch, word);
                correctChars += ch;
                correctGuess += numberOfOccurences;
            }
        } else {
            incorrectChars += ch;
            incorrectGuess ++;
        }



        if (correctGuess == (int)word.length()) {
            cout << endl << "Well done :D. The word is: " << word << endl;
            break;
        }

        if (incorrectGuess == MAX_GUESSES) {
            cout << endl << "You lose :(. The word is: " << word << endl;
            break;
        }
    } // while
    return 0;
}
```

Toàn bộ code chương trình
https://raw.githubusercontent.com/tqlong/advprogram/master/lec3-hangman/HangMan_2_0.cpp

# Nội dung

- Lỗi logic & Sửa lỗi (debugging)
- **Cấu trúc lại (refactoring)**
- Tiếp tục cấu trúc và tối ưu (optimization)
- Kỹ thuật
  - Thư viện **string**
  - Truyền tham số bằng giá trị, tham chiếu, tham chiếu hằng
  - Từ khóa **static, const**

# Hangman 2.0

```
int main()
{
    srand(time(0));

    char ch;
```

Khởi tạo

```
    string word = chooseWord();
    string secretWord = string(word.length(), '-');
    int incorrectGuess = 0, correctGuess = 0;
    string incorrectChars = "", correctChars = "";
    const int MAX_GUESSES = 7;
```

```
    while (true) {
```

Hiển thị

```
        cout << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl << endl << endl
             << endl << endl << endl << endl << endl;

        cout << getDrawing(incorrectGuess)
             << endl << "Current word: " << secretWord
             << endl << "Correct guesses: " << correctChars
             << endl << "Incorrect guesses: " << incorrectChars
             << endl << "Choose a character: ";
```

Nhập liệu

```
        cin >> ch;
```

Cập nhật

```
        if (isCharInWord(ch, word)) {
            if (!isCharInWord(ch, correctChars)) {
                int numberOfOccurences =
                  updateSecretWord(secretWord, ch, word);
                correctChars += ch;
                correctGuess += numberOfOccurences;
            }
        } else {
            incorrectChars += ch;
            incorrectGuess ++;
        }
```

Kiểm tra

```
        if (correctGuess == (int)word.length()) {
            cout << endl << "Well done :D. The word is: " << word << endl;
            break;
        }

        if (incorrectGuess == MAX_GUESSES) {
            cout << endl << "You lose :(. The word is: " << word << endl;
            break;
        }
    } // while
    return 0;
}
```

Toàn bộ code chương trình
https://raw.githubusercontent.com/tqlong/advprogram/master/lec3-hangman/HangMan_2_0.cpp

# Cấu trúc lại code (refactoring)

- Tiếp tục mô-đun hóa chương trình bằng hàm
  - Khởi tạo, hiển thị, nhập liệu, cập nhật, kiểm tra
  - Giữ hàm **main()** "sạch", dễ hiểu, dễ bảo trì và phát triển tiếp
  - Sử dụng tham chiếu để thay đổi tham số

```
void initialize(string& word, string& secretWord,
                int& incorrectGuess, int& correctGuess,
                string& incorrectChars, string& correctChars)
{
    word = chooseWord();
    secretWord = string(word.length(), '-');
    incorrectGuess = 0;
    correctGuess = 0;
    incorrectChars = "";
    correctChars = "";
}
```

```
char ch;
string word, secretWord;
int incorrectGuess, correctGuess;
string incorrectChars, correctChars;
const int MAX_GUESSES = 7;

initialize(word, secretWord, incorrectGuess, correctGuess,
           incorrectChars, correctChars);
```

# Truyền tham trị

Hiển thị

- Chỉ cần giá trị các tham số
- Không cần thay đổi các tham số

```
void render(string word, string secretWord,
            int incorrectGuess, int correctGuess,
            string incorrectChars, string correctChars,
            int MAX_GUESSES)
{
    cout << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl;

    cout << getDrawing(incorrectGuess)
         << endl << "Current word: " << secretWord
         << endl << "Correct guesses: " << correctChars
         << endl << "Incorrect guesses: " << incorrectChars
         << endl << "Choose a character: ";

    if (correctGuess == (int)word.length())
        cout << endl << "Well done :D. The word is: " << word << endl;

    if (incorrectGuess == MAX_GUESSES)
        cout << endl << "You lose :(. The word is: " << word << endl;
}
```

# Truyền tham chiếu hằng

Khi không cần thay đổi giá trị tham số, *truyền tham chiếu hằng* nhanh hơn do *không phải sao chép dữ liệu (mảng)*

- Thêm từ khóa **const**

```
void render(const string& word, const string& secretWord,
            int incorrectGuess, int correctGuess,
            const string& incorrectChars, const string& correctChars,
            int MAX_GUESSES)
{
    cout << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl;

    cout << getDrawing(incorrectGuess)
         << endl << "Current word: " << secretWord
         << endl << "Correct guesses: " << correctChars
         << endl << "Incorrect guesses: " << incorrectChars
         << endl << "Choose a character: ";

    if (correctGuess == (int)word.length())
        cout << endl << "Well done :D. The word is: " << word << endl;

    if (incorrectGuess == MAX_GUESSES)
        cout << endl << "You lose :(. The word is: " << word << endl;
}
```

# Truyền nhiều loại tham số

- Sử dụng tham trị, tham chiếu, tham chiếu hằng để phân biệt các loại tham số

Cập nhật

```
void update(char ch, const string& word,
            string& secretWord,
            int& incorrectGuess, int& correctGuess,
            string& incorrectChars, string& correctChars)
{
    if (isCharInWord(ch, word)) {
        if (!isCharInWord(ch, correctChars)) {
            int numberOfOccurences = updateSecretWord(secretWord, ch, word);
            correctChars += ch;
            correctGuess += numberOfOccurences;
        }
    } else {
        incorrectChars += ch;
        incorrectGuess ++;
    }
}
```

# Từ khóa static

- Mỗi lần gọi hàm **getDrawing()**, chương trình khởi tạo lại biến **figure** chứa các hình vẽ
  - Do biến này bị xóa khỏi bộ nhớ sau lần gọi hàm trước
- Dùng từ khóa **static**: giữ biến tồn tại trong bộ nhớ suốt quá trình chạy chương trình
  - Kết hợp với **const** để giữ biến không thay đổi

```
static string figure[] = { ... };
```

```
static const string figure[] = { ... };
```

# Hàm main() sau khi refactoring

**Hangman 2.1**
https://raw.githubusercontent.com/tqlong/advprogram/master/lec3-hangman/HangMan_2_1.cpp

```
int main()
{
    srand(time(0));

    char ch;
    string word, secretWord;
    int incorrectGuess, correctGuess;
    string incorrectChars, correctChars;
    const int MAX_GUESSES = 7;

    initialize(word, secretWord, incorrectGuess, correctGuess, incorrectChars, correctChars);

    do {
        render(word, secretWord, incorrectGuess, correctGuess, incorrectChars, correctChars, MAX_GUESSES);
        cin >> ch;
        update(ch, word, secretWord, incorrectGuess, correctGuess, incorrectChars, correctChars);
    } while (correctGuess < (int)word.length() && incorrectGuess < MAX_GUESSES);
    render(word, secretWord, incorrectGuess, correctGuess, incorrectChars, correctChars, MAX_GUESSES);
    return 0;
}
```

# Nội dung

- Lỗi logic & Sửa lỗi (debugging)
- Cấu trúc lại (refactoring)
- **Tiếp tục cấu trúc và tối ưu (optimization)**
- Kỹ thuật
  - Thư viện **string**
  - Truyền tham số bằng giá trị, tham chiếu, tham chiếu hằng
  - Từ khóa **static, const**

# Cấu trúc lại và tối ưu

Làm sau khi có một chương trình/hàm chạy để:

- Giúp dễ hiểu, dễ bảo trì, dễ phát triển tiếp hơn
- Giúp ít lỗi hơn (lỗi logic, lỗi xử lý thiếu nhập liệu người dùng, lỗi bộ nhớ - memory leak, …)
- Giúp chạy nhanh hơn, tốn ít bộ nhớ và tài nguyên hơn (làm khi thực sự cần)

# Dễ hiểu, bảo trì, phát triển tiếp

- Phần mềm thường phức tạp, phát triển lâu bởi cả nhóm người (đến và đi)
- Thời gian phát triển ban đầu thường ít hơn nâng cấp, bảo trì sau này
- Đọc code nhiều chẳng kém viết code:
  - Đọc lại code của mình khi viết, khi sửa, ...
  - Đọc code người khác để dùng, để làm tiếp, …
- Ví dụ: Linux kernel (1990 tới giờ, nhiều phiên bản, hầu như không còn code của phiên bản đầu tiên, người phát triển khắp thế giới, ~90 MB nén...)

# Tiếp tục cấu trúc và tối ưu code

- *Nhận xét:* khi thắng cuộc, **correctGuess == word.length()** thì **word == secretWord**
- Có thể bỏ **correctGuess** khỏi mã chương trình
  - Không cần đếm số ký tự đoán đúng nữa (đơn giản hơn)
  - Các lỗi trước đây liên quan đến biến **correctGuess** biến mất
  - Tận dụng khả năng so sánh chuỗi ký tự của **string** (tránh lỗi)

# Tối ưu code

```
void initialize(string& word,
                string& secretWord,
                int& incorrectGuess,
                int& correctGuess,
                string& incorrectChars,
                string& correctChars)
{
    word = chooseWord();
    secretWord =
        string(word.length(), '-');
    incorrectGuess = 0;
    correctGuess = 0;
    incorrectChars = "";
    correctChars = "";
}
```

```
void render(const string& word, const string& secretWord,
            int incorrectGuess, int correctGuess,
            const string& incorrectChars, const string& correctChars,
            int MAX_GUESSES)
{
    cout << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl << endl << endl
         << endl << endl << endl << endl << endl;

    cout << getDrawing(incorrectGuess)
         << endl << "Current word: " << secretWord
         << endl << "Correct guesses: " << correctChars
         << endl << "Incorrect guesses: " << incorrectChars
         << endl << "Choose a character: ";

    if (word == secretWord)
        cout << endl << "Well done :D. The word is: " << word << endl;

    if (incorrectGuess == MAX_GUESSES)
        cout << endl << "You lose :(. The word is: " << word << endl;
}
```

# Tối ưu code

```
void update(char ch, const string& word,
            string& secretWord,
            int& incorrectGuess,
            int& correctGuess,
            string& incorrectChars,
            string& correctChars)
{
    if (isCharInWord(ch, word)) {
        if (!isCharInWord(ch, correctChars)) {
            int numberOfOccurences =
            updateSecretWord(secretWord, ch, word);
            correctChars += ch;
        }
    } else {
        incorrectChars += ch;
        incorrectGuess ++;
    }
}
```

```
void updateSecretWord(string& secretWord,
                      char ch, string word)
{
    int len = word.length();
    for (int i = 0; i < len; ++i) {
        if (word[i] == ch) {
            secretWord[i] = ch;
        }
    }
}
```

# Hangman 2.2

**Hangman 2.2**
https://raw.githubusercontent.com/tqlong/advprogram/master/lec3-hangman/HangMan_2_2.cpp

```
int main()
{
    srand(time(0));

    char ch;
    string word, secretWord;
    int incorrectGuess;
    string incorrectChars, correctChars;
    const int MAX_GUESSES = 7;

    initialize(word, secretWord, incorrectGuess, incorrectChars, correctChars);

    do {
        render(word, secretWord, incorrectGuess, incorrectChars, correctChars, MAX_GUESSES);
        cin >> ch;
        update(ch, word, secretWord, incorrectGuess, incorrectChars, correctChars);
    } while (word != secretWord && incorrectGuess < MAX_GUESSES);
    render(word, secretWord, incorrectGuess, incorrectChars, correctChars, MAX_GUESSES);
    return 0;
}
```