Game: Guess It

2 - Hàm

https://github.com/chauttm/advprogram

Nội dung

- Game: Đoán số (Guess It)
- **Chuyển quy trình thành chương trình**
  - Mô-đun hóa chương trình bằng việc sử dụng hàm
- Kỹ thuật:
  - Sinh số ngẫu nhiên
  - Vòng lặp, điều kiện vòng lặp
  - Tham số của hàm

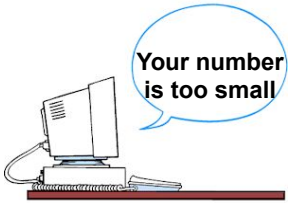
Đoán số: Luật chơi

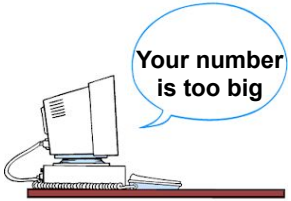
- Hai người: chủ trò - A, người chơi - B.
- Người A chọn số bất kỳ từ 1-100
- Người B đoán con số này
  - Nếu đúng, người B thắng.
  - Nếu sai, người A sẽ trả lời con số người B đoán là lớn hơn hay nhỏ hơn. Người B tiếp tục đoán số.
- Cho trẻ em học Toán
- Giúp hiểu thuật toán quan trọng:
  Tìm kiếm nhị phân (Binary Search)

Đoán số: Chương trình

- Giữa người (B) và máy (chủ trò - A).
- Máy "nghĩ" ra một con số từ 1 đến 100
- Người chơi đoán con số này
  - Nếu đúng, người chơi thắng cuộc. Nếu sai, máy sẽ trả lời con số người chơi đoán lớn hơn hay nhỏ hơn con số của máy để người chơi tiếp tục đoán số
- Cách khác: người (chủ trò - A) và máy (đoán - B).

Ví dụ một lần chơi





Nội dung

- Game: Đoán số (Guess It)
- **Chuyển quy trình thành chương trình**
  - Quá trình làm mịn dần

    thuật toán ⇒ mã giả ⇒ chương trình
  - Mô-đun hóa chương trình bằng việc sử dụng hàm
- Kỹ thuật:
  - Sinh số ngẫu nhiên
  - Vòng lặp, điều kiện vòng lặp

Mô tả thành các bước (bằng lời)

- Máy tính nghĩ số
- Nhập con số người chơi đoán
- Máy chọn câu trả lời phù hợp
- Lặp lại nếu người chơi chưa đoán đúng

Mô tả thành các bước (gần máy)

- B1: Máy tính nghĩ số
- B2: Nhập con số người chơi đoán
- B3: Máy chọn câu trả lời phù hợp.
- B4:
  Nếu người chơi đoán sai. Quay lại B2.
  Nếu người chơi đoán đúng. Chuyển tới B5.
- B5: Kết thúc.

Chương trình (mã giả, gần máy)

```
secretNumber = generateRandomNumber();    // B1
while (true) {
        guess = getPlayerGuess();    // B2
        printAnswer(guess, secretNumber);  // B3
        if (guess == secretNumber) break; // B4
        // else continue;
 }
// B5
```

Chương trình (ngôn ngữ C++)

```
secretNumber = generateRandomNumber();
do {
        guess = getPlayerGuess();
        printAnswer(guess, secretNumber);
 }  while (guess != secretNumber);
```

Tách hàm (mô-đun hóa)

Viết chương trình như kể một câu chuyện

Tìm cách đặt tên cho từng bước

- Nghĩ số: **int generateRandomNumber()**
- Nhập con số mà người chơi đoán:
  int getPlayerGuess()
- In câu trả lời theo kết quả đoán:
  void printAnswer(int guess, int secretNumber)
- Tên biến = cụm danh từ
- Tên hàm = cụm động từ

Sao lại tiếng Anh ? Khó thế :(

- Tiếng Anh là ngôn ngữ của Công nghệ thông tin (IT - Information Technology):
  - Từ khoá ngôn ngữ lập trình
  - Tài liệu, sách vở tiếng Anh nhiều
  - Tìm kiếm trên Internet; Trao đổi với cộng đồng lập trình viên trên thế giới (không chỉ Anh, Mỹ, Úc)
- Viết phần mềm cho thế giới: gia công phần mềm hay đưa “app” của mình lên Internet.
- Học thêm tiếng Anh

Chiến lược phát triển

Quy trình / thuật toán ⇒ mã giả

⇒ chương trình chạy được

⇒ phần mềm hoàn chỉnh

- Chuyển quy trình thành chương trình tốt thiểu chạy được
- Chương trình chạy được càng sớm càng tốt, cố gắng lúc nào cũng chạy được.
- Thêm dần dần các chi tiết để hoàn thiện.
- Thêm đến đâu, chạy đúng đến đó
- Làm từ dễ đến khó

Guess It 1.0

```cpp
#include <iostream>
#include <cstdlib>

using namespace std;

int generateRandomNumber();
int getPlayerGuess();
void printAnswer(int guess, int secretNumber);

int main()
{
    int secretNumber = generateRandomNumber();
    int guess;

    do {
        guess = getPlayerGuess();
        printAnswer(guess, secretNumber);
    } while (guess != secretNumber);

    return 0;
}
```

Nội dung

- Game: Đoán số (Guess It)
- **Chuyển quy trình thành chương trình**
  - Mô-đun hóa chương trình bằng việc sử dụng hàm
- Kỹ thuật:
  - Sinh số ngẫu nhiên
  - Vòng lặp, điều kiện vòng lặp

Guess It 1.0

```
using namespace std;

int generateRandomNumber();
int getPlayerGuess();
void printAnswer(int guess, int secretNumber);

int main()
{
    int secretNumber = generateRandomNumber();
    int guess;

    do {
        guess = getPlayerGuess();
        printAnswer(number, secretNumber);
    } while (guess != secretNumber);

    return 0;
}
```

Còn thiếu định nghĩa các hàm

Sẽ làm nhanh những phần dễ để nhanh chóng có chương trình chạy được

Fake hàm sinh số ngẫu nhiên

Vì sao?

- Chưa biết làm
- Chưa cần làm
- Cố định số cần đoán để dễ test phần còn lại của game

```
int generateRandomNumber()
{
    return 42;
}
```

Nhập con số người chơi đoán

- Quá dễ

```
int guess;
cout << endl << "Enter your guess: ";
cin >> guess;
```

Guess It 1.0

```
int main()
{
    int secretNumber = generateRandomNumber();
    int guess;

    do {
        guess = getPlayerGuess();
        printAnswer(guess, secretNumber);
    } while (guess != secretNumber);

    return 0;
```

```
int getPlayerGuess()
{
    int guess;
    cout << endl << "Enter your guess (1..100): ";
    cin >> guess;
    return guess;
}
```

Máy tính chọn câu trả lời

- Lựa chọn bằng **if … else if … else** liên tiếp

```
if (guess > secretNumber) {
    cout << "Your number is too big." << endl;
} else if (guess < secretNumber) {
    cout << "Your number is too small." << endl;
} else {
    cout << "Congratulation! You win." << endl;
}
```

Guess It 1.0

```
int main()
{
    int secretNumber = generateRandomNumber();
    int guess;

    do {
        guess= getPlayerGuess();
        printAnswer(guess, secretNumber);
    } while (guess != secretNumber);
```

```
void printAnswer(int guess, int secretNumber)
{
    if (guess > secretNumber) {
        cout << "Your number is too big." << endl;
    } else if (guess < secretNumber) {
        cout << "Your number is too small." << endl;
    } else {
        cout << "Congratulation! You win." << endl;
    }
}
```

Guess It 1.0

```
void printAnswer(int guess, int secretNumber)
{
    if (guess > secretNumber) {
        cout << "Your number is too big." << endl;
    } else if (guess < secretNumber) {
        cout << "Your number is too small." << endl;
    } else {
        cout << "Congratulation! You win." << endl;
    }
}
```

Kết quả

```
G:\advprogram\lec2-guessit\GuessIt.exe

Enter your guess (1..100): 50
Your number is too big.

Enter your guess (1..100): 25
Your number is too small.

Enter your guess (1..100): 42
Congratulation! You win.
```

Sinh số ngẫu nhiên

- Máy tính không thể thật sự “ngẫu nhiên”
- Sinh số “giả ngẫu nhiên” - pseudo random
  - Tìm kiếm Google: “C++ random”
  - Hàm **rand()** trong <cstdlib>
  - Hằng RAND_MAX

```
v1 = rand() % 100;        // v1 in the range 0 to 99
v2 = rand() % 100 + 1;    // v2 in the range 1 to 100
v3 = rand() % 30 + 1985; // v3 in the range 1985-2014
```

```
int randomNumber = rand() % 100 + 1;
```

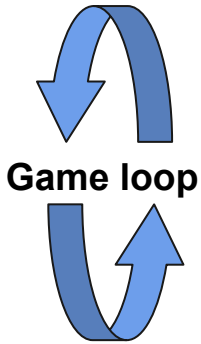
Guess It 1.1

```
using namespace std;

int generateRandomNumber();
int getPlayerGuess();
void printAnswer(int guess, int secretNumber);

int main()
{
    int secretNumber = generateRandomNumber();
    int guess;

    do {
        guess = getPlayerGuess();
        printAnswer(number, secretNumber);
    } while (guess != secretNumber);
```

```
int generateRandomNumber()
{
    return rand() % 100 + 1;
}
```

Lặp lại (Game loop)

- Nếu người chơi đoán sai, lặp lại bước nhập
- Cần hỏi người chơi ít nhất 1 lần
- Vòng lặp **do … while**




```
do {
    // Nhập số người chơi đoán
    // In câu trả lời phù hợp
} while (guess != secretNumber);
```

Thực hành

- Hiển thị số lần đoán của người chơi
- Điểm của người chơi = 100 - số lần đoán
- Điểm của người chơi =
  10000 - 2 x 100 - 2 x 99 - …
  tùy theo số lần người chơi đoán
- Cho phép chơi nhiều ván
  - Hỏi người chơi có muốn chơi tiếp không ?
  - Gợi ý: đưa toàn bộ mã trong hàm **main()** vào một hàm **playGuessIt()**

Cho phép chơi nhiều ván

- Bạn có nhận ra mỗi lần chạy chương trình, máy "nghĩ" ra cùng một con số?
  - Lý do: chưa khởi tạo hạt giống sinh số ngẫu nhiên
- Lệnh: **srand(time(0));**

  khởi tạo "hạt giống" (seed) cho hàm **rand()** bằng thời gian bắt đầu chạy chương trình
  - Lưu ý: chỉ cần gọi **srand()** một lần và gọi trước khi bắt đầu dùng **rand()**.

  Để mỗi lần chạy, chương trình dùng một hạt giống khác nhau: dùng thời gian hiện hành làm seed

Tổng kết

- *Viết chương trình theo kiểu top-down*
  - Dùng hàm để chương trình dễ đọc như một câu chuyện
  - Mỗi hàm dài không quá 01 trang để dễ kiểm soát
- Tìm kiếm, tra cứu kỹ thuật lập trình
  - Không thể thiếu tiếng Anh
- Sinh số ngẫu nhiên
  - Tạo seed một lần ở đầu chương trình srand(time(0))
  - Lấy giá trị ngẫu nhiên rand()
- Truyền tham số vào hàm
  - Tham trị / tham biến
- Lựa chọn bằng **if … else if … else**
- Vòng lặp **do … while**

Máy chơi Guess It

- Đặt vấn đề: đảo vai trò người và máy
  - Người làm chủ trò, nghĩ số từ 1 đến 100
  - Máy đoán số
  - Người “thông báo” cho máy giá trị máy đoán lớn hơn, nhỏ hơn hay đúng bằng giá trị cần tìm
- Có nhiều cách chơi (thuật toán đoán)
  - Đoán ngẫu nhiên
  - Đoán tuần tự từ 1 đến 100 (hoặc ngược lại)
  - Đoán “đại” một số, nhận trả lời của người chơi để phán đoán lần sau nên đoán thế nào

Thuật toán chung

B1: select a number X in [1, 100]

B2: ask for host's answer on X

B3: if X is correct, exit (win)

else goto B1

Thuật toán chung

```
int X, answer;
do {
    X = selectNumber(1, 100);
    answer = getHostAnswer(X);
    if (answer == '=')
        cout << "Your number is " << X << endl;
} while (answer != '=');
```

```
char getHostAnswer(int X)
{
    char answer;
    cout << "Is " << X
         << " your number? ";
    cin >> answer;
    return answer;
}
```

tùy thuộc vào hàm selectNumber() mà ta có các cách đoán (thuật toán đoán) khác nhau

Quy ước trả lời:
answer có thể là >, <, =
>: X lớn hơn số cần đoán
<: X nhỏ hơn số cần đoán
=: X bằng số cần đoán

Đoán ngẫu nhiên (may rủi)

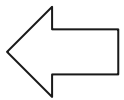
```
int selectNumber(int low, int high)
{
    return rand() % (high-low+1) + low;
}
```

Rõ ràng không thể biết lúc nào thuật toán này đoán được số cần tìm

Tìm kiếm tuần tự (chắc ăn)

```
int selectNumber(int low, int high)
{
    return low;
}
...
int X, answer, low = 1;
do {
    X = selectNumber(low, 100);
    answer = getHostAnswer(X);
    if (answer != '=') low++;
} while (answer != '=');
```

- Lần lượt đoán các số từ 1 đến 100
- Thuật toán chắc chắn đoán ra số cần tìm
- Nếu số cần đoán là 100 thì cần 100 lần đoán



Nhích cận dưới của lần đoán sau lên 1 đơn vị

Đoán đại rồi chỉnh khoảng tin cậy

```
int selectNumber(int low, int high)
{
    return rand() % (high-low+1) + low;
}
...
int X, answer, low = 1, high = 100;
do {
    X = selectNumber(low, high);
    answer = getHostAnswer(X);
    if (answer == '>')  high = X-1; // X lớn hơn nên giảm high
    if (answer == '<') low  = X+1; // X nhỏ hơn nên tăng low
} while (answer != '=');
```

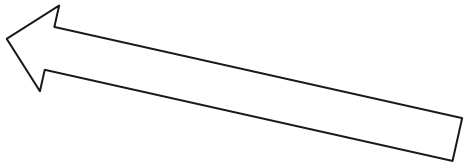
- Tuy may rủi nhưng chắc chắn đoán ra số cần tìm (*tại sao*)
- Nếu số X chỉ tăng hoặc giảm 1 đơn vị so với lần lặp trước → *giống tìm kiếm tuần tự* (không may)

Cải tiến khoảng tin cậy

- Thuật toán chia đôi (tìm kiếm nhị phân)
  - Chọn số X là điểm giữa khoảng [low, high]
  - Mỗi lần đoán (sai), kích thước khoảng tin cậy giảm ít nhất một nửa
  - Số lần đoán tối đa ≈ $\log_2 100 = 7$

Thuật toán chia đôi

```
int selectNumber(int low, int high)
{
    return (low+high) / 2;
}
...
int X, answer, low = 1, high = 100;
do {
    X = selectNumber(low, high);
    answer = getHostAnswer(X);
    if (answer == '>')  high = X-1; // X lớn hơn nên giảm high
    if (answer == '<') low  = X+1; // X nhỏ hơn nên tăng low
} while (answer != '=');
```



chắc chắn giảm kích thước
khoảng tin cậy ít nhất một nửa