

Якісна документація в довільному стилі

Проблеми: у багатьох із нас є компанія друзів, настільки лінивих, що здається, ніби вони змагаються у цьому виді спорту. І за дивним збігом обставин, наша компанія завжди складається з рівно 100 осіб (не більше, не менше – це якось по-змовницьки, правда?).

Тож, коли ми нарешті збираємось усі разом на 6-му поверсі багатоповерхівки з непрацюючим ліфтом і вирішуємо замовити їжу, починається справжнє випробування. Проблема завжди одна й та сама: ніхто не хоче спуститися на 5-й поверх, щоб зустріти кур'єра та забрати їжу. І тут кожен вмикає "режим креативності", вигадуючи шедевральні причини, чому він точно не може це зробити. У підсумку, ми часто лишаємось голодними, бо перемогти в битві відмовлялок неможливо.

Рішення? Призначити відповідальну особу за допомогою машини, яка не знає, що таке "лінощі" або "винахідливі відмазки". Машина без вагань обере "щасливця", якому доведеться залишити затишний 6-й поверх і зіштовхнутись віч-на-віч із викликом — зустріч кур'єра.

Задача: оскільки ця ситуація вже не просто дратує, а фактично унеможлиблює нормальне існування, треба терміново вирішити проблему. Необхідно створити робота без будь-яких ознак емпатії, який не піддається хитрощам чи відмазкам і зможе призначити "щасливця", відповідального за зустріч з кур'єром.

Особливості задачі: через невідомі обставини на моїй клавіатурі відсутня клавіша "Enter", тому весь код доведеться писати в одному рядку. І ще одне: рандом має бути справжнім рандомом, а не ось цими псевдовипадковими підтасовками, де завжди обирається один і той самий "щасливець".

Рішення

Програму я писав на мові Сі. Чому саме на Сі? А вас це взагалі не повинно хвилювати.

Цей код — спроба створити псевдовипадкове число на основі адреси змінної і хаотичної суміші арифметичних та побітових операцій. Основна мета —

отримати щось, що виглядає як випадкове число, без використання стандартних генераторів випадкових чисел, бо ж навіщо полегшувати собі життя?

```
int main() { int dummyVar = 0xBADFACE; int notOne = 0xFFFFFFFF; int
hundred = 0xA * 0xA; int myPassword = 1234; int myLuckyNumber = 7; int
myBinaryUnluckyNumber = 0b0111; int zodiacSignFish = 69; int myBirthDay =
13; unsigned int seed = (unsigned long)&dummyVar; unsigned int
randomNumber = ((seed * seed / ((myPassword + zodiacSignFish) ^
myBirthDay) + seed * (myLuckyNumber + myBinaryUnluckyNumber)) % hundred) +
~notOne; ((randomNumber < ~notOne) || (randomNumber > hundred)) ?
(randomNumber = ~randomNumber) : (randomNumber = randomNumber); return
(hundred - randomNumber + ~notOne); }
```

Розбір коду

Змінні:

- `int dummyVar = 0xBADFACE;`

Змінна-манекен. Її єдине призначення — просто бути, щоб програма мала сенс свого існування. Ось кілька HEX-кодів, з яких можна скласти слова: ACE, BED, BAD, CAB, DEAF, FACE. Нічого дотепнішого, ніж BADFACE, я не вигадав.

- `int notOne = 0xFFFFFFFF;`

Тут все просто: це змінна, яка буквально означає "не дорівнює 1".

- `int hundred = 0xA * 0xA;`

Ну тут треба рівняння розв'язати. Щоб з'ясувати що таке 0xA.

$$A * A = 100;$$

$$A = \sqrt{100};$$

$$A = 10;$$

- `int myPassword = 1234;`

Логіну немає, тож я не хвилююся, що мене можуть зламати. Це ж явно найнадійніший пароль у світі.

- `int myLuckyNumber = 7;`

Щоб дізнатися своє щасливе число, необхідно провести нумерологічний розрахунок. Складіть усі числа з вашої дати народження. Наприклад, якщо ви народилися 13 жовтня 2000 необхідно зробити наступний розрахунок: $1+3+1+0+2+0+0+0 = 7$.

- `int myBinaryUnluckyNumber = 0b0111;`

Цифри в бінарному вигляді? Просто так, бо мені не подобається це число.

- `int zodiacSignFish = 69;`

І це не хтивий жарт на тему дорослих приколів. Це просто знак зодіаку мого товариша — Риби.

- `int myBirthday = 13;`

Номер дня в який я народився.

- `unsigned int seed = (unsigned long)&dummyVar;`

Справжня магія починається тут. Використовую адресу змінної `dummyVar`, щоб кожен запуск програми давав різний результат, залежно від розміщення цієї змінної в пам'яті.

Обчислення рандомного числа:

```
unsigned int randomNumber = ((seed * seed /  
((myPassword + zodiacSignFish) ^ myBirthday) +  
seed * (myLuckyNumber + myBinaryUnluckyNumber)) %  
hundred) + ~notOne;
```

Що ж тут відбувається? Відповідно до заповіді програмістів: "Працює? Не чіпай!". Після багатьох досліджень, проб і помилок, цей код генерує числа від 1 до 100. (Якщо чесно, то все так складно не задля приколу, але й для того щоб генерувались не тільки парні числа).

Перевірка отриманого числа:

```
((randomNumber < ~notOne) || (randomNumber >  
hundred)) ? (randomNumber = ~randomNumber) :  
(randomNumber = randomNumber);
```

Тут йде перевірки отриманого числа, щоб воно точно було в нашому діапазоні.

Повернення результату

```
return (hundred - randomNumber + ~notOne);
```

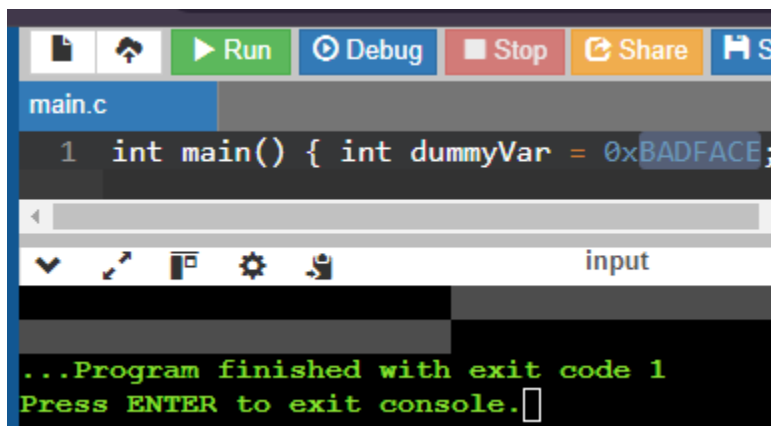
Якщо від 100 відняти наше випадкове число та додати потім 1, то ми отримаємо випадкове число від 1 до 100.

Тестування

Для спрощення використаємо онлайн компілятор, я обрав

[“https://www.onlinegdb.com/online_c_compiler#”](https://www.onlinegdb.com/online_c_compiler#)

Це своєрідна гарантія, що число в межах нашого діапазону. Якщо ні — воно просто інвертується, щоб вміститися в діапазон. Запустив програму, і можу підтвердити — вона генерує випадкові числа. Не знаю, наскільки випадкові, але вони точно різні.



The screenshot shows an online C compiler interface. At the top, there are buttons for 'Run', 'Debug', 'Stop', 'Share', and 'Save'. Below these is a text editor with a file named 'main.c' containing the following code:

```
1 int main() { int dummyVar = 0xBADF00F;
```

Below the code editor is an 'input' field. At the bottom, the output console shows the following message:

```
...Program finished with exit code 1
Press ENTER to exit console.
```

```
...Program finished with exit code 97
Press ENTER to exit console.
```

```
...Program finished with exit code 44
Press ENTER to exit console.
```

```
...Program finished with exit code 2
Press ENTER to exit console.
```

```
...Program finished with exit code 23
Press ENTER to exit console.
```

```
...Program finished with exit code 68  
Press ENTER to exit console.█
```

```
...Program finished with exit code 78  
Press ENTER to exit console.█
```

Висновок: ну короче, програма працює, числа рандомні, і, головне, проблема лінивих друзів вирішена.