**Thợ lành nghề #1 - Mở đầu Thảm họa**

Ngày 13 tháng 2 năm 2002.

Nhật ký thân mến,

Hôm nay đúng là một ngày xui xẻo – Tôi làm hỏng mọi chuyện. Tôi rất muốn gây ấn tượng với các ngài “cựu học việc” ở đây nhưng rốt cuộc chỉ làm rối tung cả lên.

Ðó là ngày đầu tiên tôi được làm một chân học việc của ông C. Tôi quả rất may mắn mới có được chân này. Ông C là một người có tiếng trong làng phát triển phần mềm. Cuộc thi để giành được chân học việc này đúng là nảy lửa. Những người theo học ông C thường sẽ trở thành những người thợ lành nghề được đánh giá cao. Ðiều này có nghĩa được làm việc với ông C có giá trị rõ ràng.

Tôi cứ ngỡ hôm nay sẽ được gặp ông C, nhưng tôi bị gã “cựu học việc” Jerry níu qua một bên. Gã bảo ông C luôn luôn yêu cầu phần định hướng cho người học việc trong những ngày đầu. Gã nói ông C nhất quyết cho rằng phần định hướng này rất thiết thực với người học việc, và chất lượng mã nguồn mà ông mong đợi ở họ.

Tôi háo hức kinh khủng. Ðây là cơ hội để tôi cho họ thấy mình là một tay lập trình giỏi cỡ nào. Thế là tôi bảo Jerry tôi không chờ được nữa. Gã đáp lại sự háo hức của tôi bằng cách bảo tôi thử viết một chương trình đơn giản. Gã muốn tôi dùng [Sàng Eratosthenes](http://vi.wikipedia.org/wiki/S%C3%A0ng_Eratosthenes) (Sieve) để tính các số nguyên tố. Gã còn bảo tôi phải chuẩn bị xong chương trình bao gồm trọn bộ các kiểm thử đơn vị để soát sau buổi ăn trưa.

Thật tuyệt! Tôi có gần 4 giờ đồng hồ để “xào nấu” một chương trình đơn giản như Sieve. Tôi quyết tâm làm một chương trình thật ấn tượng. Tôi đã viết mã ở Mã dẫn 1. Tôi nắm chắc là chương trình được chú thích cẩn thận và trình bày gọn gàng.

*Mã dẫn 1*

/\*\*

\* This class generates prime numbers up to a user-specified maximum.

\* The

\* algorithm used is the Sieve of Eratosthenes. &lt;p&gt; Eratosthenes of Cyrene, b.c.

\* 276 BC, Cyrene, Libya; d.c.194 BC,Alexandria. He was the first man to

\* calculate the circumference of the Earth, and was also known for working on

\* calendars with leap years and running the library at Alexandria.&lt;/p&gt; \* The

\* algorithm is quite simple: Given an array of integers starting at 2, cross

\* out all multiples of 2. Find the next uncrossed integer, and cross out all of

\* its multiples. \* Repeat until you have passed the square root of the maximum

\* value.

\* @authorAlphonse,

\* @version 13 Feb 2002 atp

\*/

import java.util.\*;

public class GeneratePrimes {

/\*\*

\* @param maxValue is the generation limit.

\*/

public static int[] generatePrimes(int maxValue)

{

if (maxValue &gt;= 2) { // the only valid case

// declarations

int s = maxValue + 1; // size of array

boolean[] f = new boolean[s];

int i;

// initialize array to true.

for (i = 0; i &lt; s; i++)

{

f[i] = true;

}

// get rid of known non-primes.

f[0] = f[1] = false;

// sieve

int j;

for (i = 2; i &lt; Math.sqrt(s) + 1; i++)

{

if (f[i]) { // if i is uncrossed, cross its multiples.

for (j = 2 \* i; j &lt; s; j += i)

f[j] = false; // multiple is not prime

}

}

// how many primes are there?

int count = 0;

for (i = 0; i &lt; s; i++) {

if (f[i]) {

count++; // bump count.

}

}

int[] primes = new int[count];

// move the primes into the result.

for (i = 0, j = 0; i &lt; s; i++) {

if (f[i]) // if prime

{

primes[j++] = i;

}

}

return primes; // return the primes.

} else // maxValue &lt; 2

{

return new int[0]; // return null array if bad input.

}

}

}

Sau đó tôi viết một kiểm thử đơn vị cho GeneratePrimes. Xem ở mã ở Mã dẫn 2. Ðoạn mã này dùng khung làm việc JUnit như Jerry đã chỉ dẫn. Nó tiếp cận kiểu thống kê; kiểm tra xem cái “generator” có thể tạo ra các số nguyên tới 0, 2, 3 và 100. Trong trường hợp thứ nhất hẳn không có số nguyên nào cả. Trong trường hợp thứ hai phải có một số nguyên và nó phải là số 2. Trường hợp thứ ba có hai số nguyên và chúng là số 2 và 3. Trường hợp cuối phải là 25 số nguyên và số cuối phải là 97. Nếu các kiểm thử đều đạt, thì tôi giả định là “generator” làm việc đúng. Tôi e rằng khó có thể tin cậy tuyệt đối cách ở trên, nhưng tôi không nghĩ ra được một trường hợp nào một hàm có thể sai mà các bước kiểm tra vẫn đúng.

*Mã dẫn 2*

import junit.framework.\*;

import java.util.\*;

public class TestGeneratePrimes extends TestCase {

public static void main(String args[]) {

junit.swingui.TestRunner.main(new String[]{&quot;TestGeneratePrimes&quot;});

}

public TestGeneratePrimes(String name) {

super(name);

}

public void testPrimes() {

int[] nullArray = GeneratePrimes.generatePrimes(0);

assertEquals(nullArray.length, 0);

int[] minArray = GeneratePrimes.generatePrimes(2);

assertEquals(minArray.length, 1);

assertEquals(minArray[0], 2);

int[] threeArray = GeneratePrimes.generatePrimes(3);

assertEquals(threeArray.length, 2);

assertEquals(threeArray[0], 2);

assertEquals(threeArray[1], 3);

int[] centArray = GeneratePrimes.generatePrimes(100);

assertEquals(centArray.length, 25);

assertEquals(centArray[24], 97);

}

}

Tôi làm việc này mất khoảng một giờ đồng. Jerry không muốn gặp tôi trước bữa ăn trưa, bởi thế, tôi dành trọn bộ thời gian còn lại đọc cuốn Design Patterns mà Jerry đưa cho.

Sau buổi ăn trưa, tôi ghé văn phòng của Jerry và cho gã biết tôi đã thực hiện xong chương trình. Gã nhìn tôi và với một nụ cười khó tả, hắn nói: “Ðược lắm, hãy xem thử nó thế nào.”

Sau bữa trưa, gã dẫn tôi và phòng máy và cho tôi ngồi trước một máy. Gã ngồi bên cạnh tôi và yêu cầu tôi đưa chương trình của tôi vào máy này. Thế là tôi chuyển mã nguồn từ máy laptop của tôi lên.

Jerry xem xét hai mã nguồn chừng năm phút rồi gã lắc đầu và bảo: “Mày không thể đưa những cái này cho ông C xem được! Nếu tao để ông ấy xem mấy cái này, thì ông sẽ đuổi cả tao lẫn mày. Ông ấy không phải là người kiên nhẫn đâu.”

Tôi đánh thót một cái nhưng cố giữ bình tĩnh và hỏi gã: “Vậy nó sai chỗ nào?”

Jerry thở dài và nói: “Tụi mình nên cùng xem xét toàn bộ mã nguồn này. Tao sẽ chỉ cho mày từng điểm theo cách ông C muốn thực hiện nó.”

“Quá rõ ràng”, gã tiếp tục, “nên tách cái hàm main thành ba hàm riêng biệt. Hàm thứ nhất khởi tạo tất cả các biến và thiết lập cái “sieve”. Hàm thứ hai thực sự thi hành cái “sieve” và hàm thứ ba tải kết quả của “sieve” vào một dãy số nguyên.”

Tôi nhận ra được ý gã. Có ba khái niệm chôn trong cái hàm đó. Tuy vậy, tôi không biết gã muốn tôi phải làm gì.

Gã nhìn tôi một lúc, rõ ràng là đang đợi xem tôi phản ứng sao. Nhưng rốt cuộc gã thở dài, lắc đầu và….

**Bài tiếp:** [Thợ lành nghề #2 – Chế độ ăn kiêng tăng cường](https://codegym.vn/blog/2018/07/31/tho-lanh-nghe-2-che-do-an-kieng-tang-cuong/)

**Tác giả:** Robert C. Martin

**Người dịch:** Hoàng Ngọc Diêu (conmale)