Họ và tên: Phạm Văn Tiến
Msv: 20021449

Chương trình tính số fibonaci thứ n

Phương thức Fibonacci(long n) để tính số thứ n trong dãy Fibonacci. Phương thức có kiểu trả về là long. Công thức được tính như sau:

$$F_n := F(n) := \begin{cases} 0, & \text{khi } n = 0; \\ 1, & \text{khi } n = 1; \\ F(n-1) + F(n-2) & \text{khi } n > 1. \end{cases}$$

Trường hợp n < 0, hàm trả về giá trị -1. Trường hợp số Fibonacci thứ n vượt quá giá trị lớn nhất của kiểu dữ liệu long, phương thức trả về giá trị Long.MAX_VALUE.

Định dạng đầu vào: số tự nhiên n.

Định dạng đầu ra: số Fibonacci thứ n.

Điều kiện: n là số tự nhiên n <= 100

I. Kiểm thử giá trị biên

Các giá trị biên:
-Min: n=0

```
@Test
public void testFibonacci_1() {
    assertEquals( expected: 0,Code.fibonacci( n: 0));
}
```

-Max: n=100

```
@Test
public void testFibonacci_2() {
    assertEquals(Long.MAX_VALUE,Code.fibonacci( n: 100));
}
```

-Min+: n=1

```
//Min+
@Test
public void testFibonacci_3() {
    assertEquals( expected: 1,Code.fibonacci( n: 1));
}
```

-Max-: n=99

```
//Max--
@Test
public void testFibonacci_4() {
    assertEquals(Long.MAX_VALUE,Code.fibonacci( n: 99));
}
```

-Nom: n=2

```
//Nom
@Test
public void testFibonacci_5() {
    assertEquals( expected: 1,Code.fibonacci( n: 2));
}
```

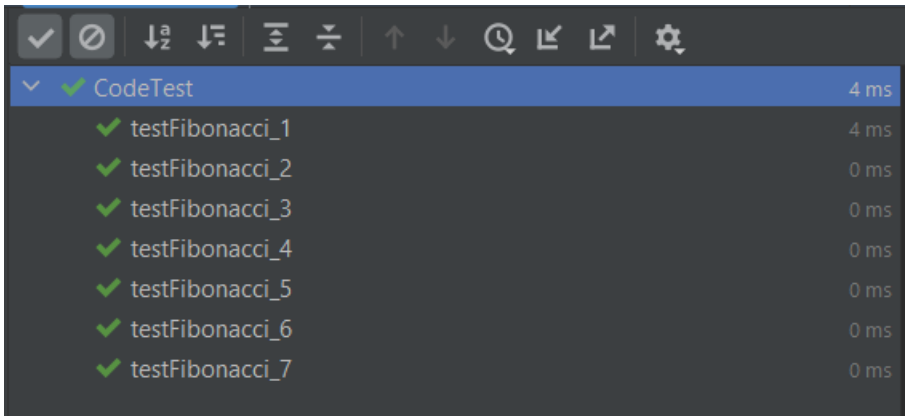
-Min-: n=-1

```
//Min-
@Test
public void testFibonacci_6() {
    assertEquals( expected: -1,Code.fibonacci( n: -1));
}
```

-Max+: n = 101:

```
//Max+
    @Test
public void testFibonacci_7() {
    assertEquals(Long.MAX_VALUE,Code.fibonacci( n: 101));
}
```

Kết quả chạy kiểm thử:



Bảng kiểm thử

Test Case	Input	Output		Result
	n	Expected	Actual	
1	0	0	0	Pass
2	100	9223372036854775807	9223372036854775807	Pass
3	1	1	1	Pass
4	99	9223372036854775807	9223372036854775807	Pass
5	2	1	1	Pass
6	-1	-1	-1	Pass
7	101	9223372036854775807	9223372036854775807	Pass

II. Kiểm thử tương đương

Các miền tương đương :

A0: (MinLong,-1]
A1: {0}
A2: {1}
A3: [2..92] (Số fibonacci thứ 92 là số lớn nhất kiểu Long có thể biểu diễn)
A4: [93..100]

Test case1: n=-2 (A0)

```
@Test
public void testFibonacci_1() {
    assertEquals( expected: -1,Code.fibonacci( n: -2));
}
```

Test case 2: n=0 (A1)

```
@Test
public void testFibonacci_1() {
    assertEquals( expected: 0,Code.fibonacci( n: 0));
}
```

Test case 3: n=1 (A2)

```
@Test
public void testFibonacci_2() {
    assertEquals( expected: 1,Code.fibonacci( n: 1));
}
```

Test case 4: n=92 (A3)

```
@Test
public void testFibonacci_3() {
    assertEquals( expected: 7540113804746346429L,Code.fibonacci( n: 92));
}
```

Test case 5: n=93 (A4)

```java
@Test
public void testFibonacci_4() {
    assertEquals(Long.MAX_VALUE,Code.fibonacci( n: 93));
}
```

Kết quả chạy kiểm thử:

Bảng kiểm thử

Test Case	Input	Output		Result
	n	Expected	Actual	
1	-2	-1	-1	Pass
2	0	0	0	Pass
3	1	1	1	Pass
4	92	7540113804746346429	7540113804746346429	Pass
5	93	9223372036854775807	9223372036854775807	Pass