Tju软件测试作业3

张环禹 软件4班 3016218159

- **Question**

1. Draw the control flow graph for the printPrimes() method.

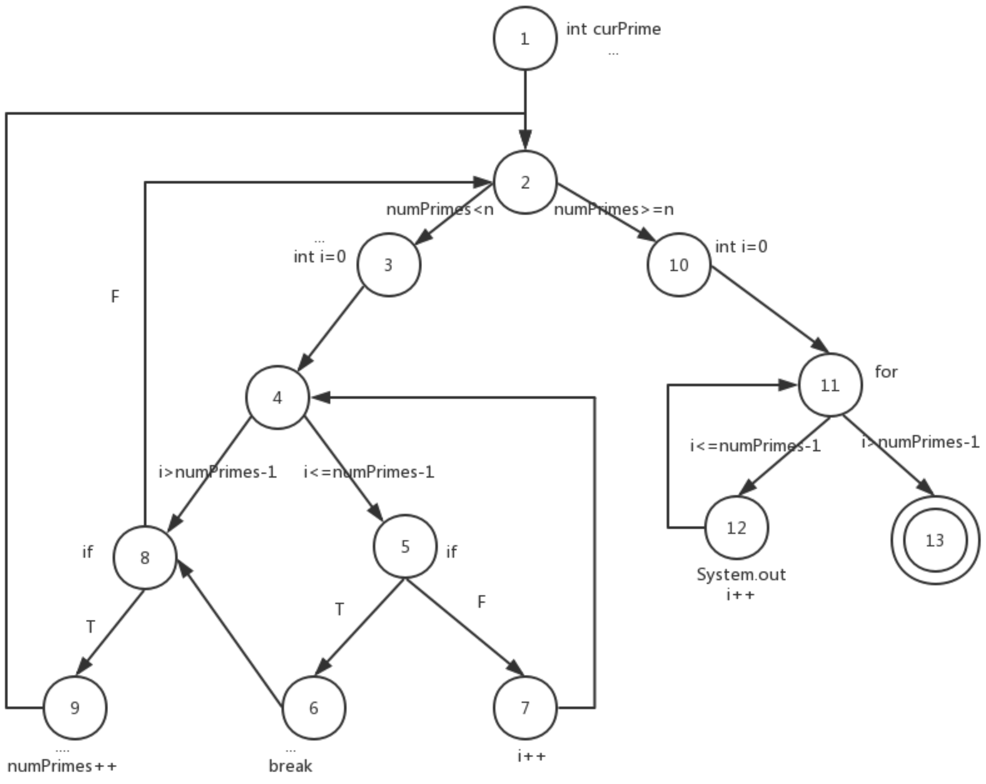
2. Consider test cases t1=(n=3) and t2=(n=5). Although these tour the same prime paths in printPrimes(), Design a simple fault that t2 would be more likely to discover than t1 would.

3. Find a test case such that the corresponding test path visits the edge that connects while's beginning statement to the for statement without going through the while's body.

4.Enumerate the test requirements for node coverage, edge coverage, and prime path coverage for the graph for printPrimes().

- **Answer**

1.



2. 我们可以将代码里的int [] primes=new int [100]; 修改为int [] primes=new int [4]; 这样，当测试n为3时用例通过，而n为5时用例数组越界，产生错误。
3. 当测试用例为n=1时会经过while起点和for语句的边，但不进入while循环体内。
4. 点覆盖：{1,2,3,4,5,6,7,8,9,10,11,12,13}

   边覆盖：{(1,2),(2,3),(2,10),(3,4),(4,5),(4,8),(5,6),(5,7),(6,8),(7,4),(8,2),(8,9),(9,2),(10,11),(11,12),(11,13),(12,11)};

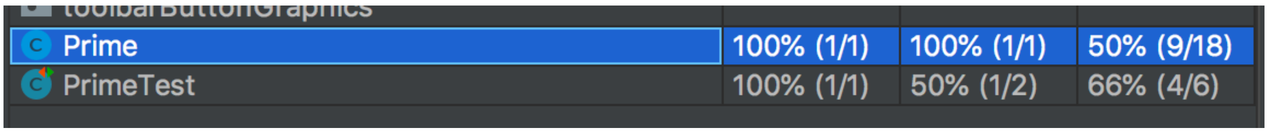
   主路径覆盖：{(1,2,3,4,8,9),(1,2,3,4,5,7),(1,2,3,4,5,6,8,9),(1,2,10,11,12),(1,2,10,11,13),(2,3,4,8,9,2),(2,3,4,8,2),(2,3,4,5,7),(2,3,4,5,6,8,9,2),(2,3,4,5,6,8,2),(2,10,11,12),(2,10,11,13),(3,4,5,6,8,9,2,3),(3,4,5,6,8,2,3),(3,4,8,9,2,3),(3,4,8,2,3),(3,4,5,6,8,9,2,10,11,12),(3,4,5,6,8,2,10,11,12),(3,4,5,6,8,9,2,10,11,13),(3,4,5,6,8,2,10,11,13),(3,4,8,9,2,10,11,12),(3,4,8,2,10,11,12),(3,4,8,9,2,10,11,13),(3,4,8,2,10,11,13),(4,5,7,4),(4,5,6,8,9,2,3,4),(4,5,6,8,2,3,4),(4,8,9,2,3,4),(4,8,2,3,4),(5,7,4,5),(5,6,8,9,2,3,4,5),(5,6,8,2,3,4,5),(6,8,9,2,3,4,5,6),(6,8,9,2,3,4,5,6),(6,8,9,2,3,4,5,7),(6,8,2,3,4,5,7),(7,4,5,7),(7,4,8,9,2,3),(7,4,8,2,3),(7,4,5,6,8,9,2,3),(7,4,5,6,8,2,3),(7,4,5,6,8,9,2,10,11,12),(7,4,5,6,8,9,2,10,11,13),(7,4,5,6,8,2,10,11,12),(7,4,5,6,8,2,10,11,13),(7,4,8,9,2,10,11,12),(7,4,8,9,2,10,11,13),(7,4,8,2,10,11,12),(7,4,8,2,10,11,13),(8,2,3,4,8),(8,9,2,3,4,8),(8,2,3,4,5,6,8),(8,9,2,3,4,5,6,8),(9,2,3,4,8,9),(9,2,3,4,5,6,8,9),(11,12,11),(12,11,12),(12,11,13)}.

- 基于Junit及Eclemma实现一个主路径覆盖的测试

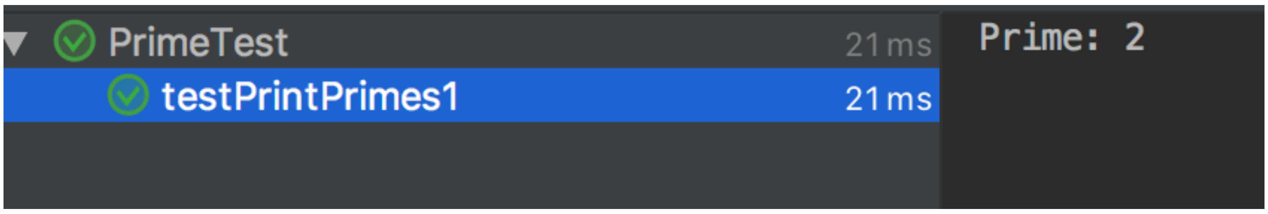
编写测试程序

```
import org.junit.Test;

public class PrimeTest {
    Prime prime =new Prime();

    @Test
    public void testPrintPrimes1() {
        prime.printPrimes(1);
    }

    @Test
    public void testPrintPrimes2() {
        prime.printPrimes(5);
    }
}
```

1️⃣ 测试用例为1时运行结果如下

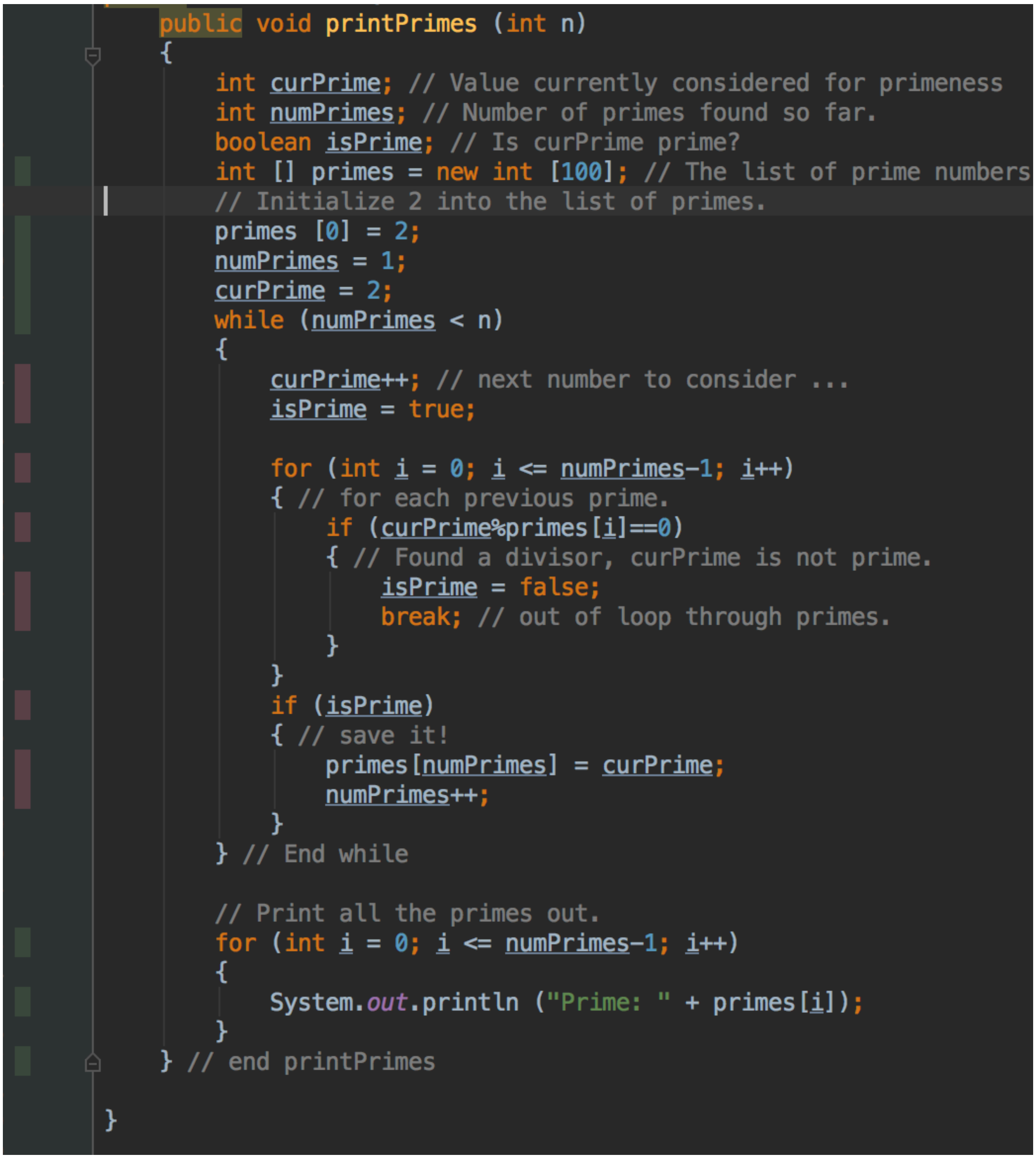


Prime	100% (1/1)	100% (1/1)	50% (9/18)
PrimeTest	100% (1/1)	50% (1/2)	66% (4/6)



```
PrimeTest 21ms
  testPrintPrimes1 21ms
```

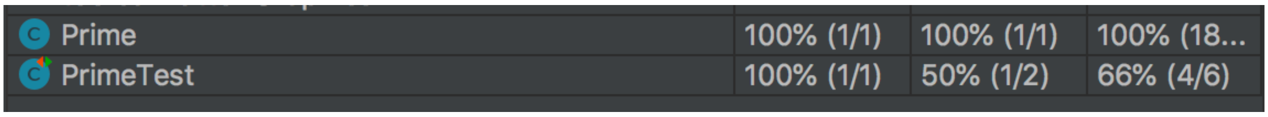
```
Prime: 2
```



```
    public void printPrimes (int n)
    {
        int curPrime; // Value currently considered for primeness
        int numPrimes; // Number of primes found so far.
        boolean isPrime; // Is curPrime prime?
        int [] primes = new int [100]; // The list of prime numbers
        // Initialize 2 into the list of primes.
        primes [0] = 2;
        numPrimes = 1;
        curPrime = 2;
        while (numPrimes < n)
        {
            curPrime++; // next number to consider ...
            isPrime = true;

            for (int i = 0; i <= numPrimes-1; i++)
            { // for each previous prime.
                if (curPrime%primes[i]==0)
                { // Found a divisor, curPrime is not prime.
                    isPrime = false;
                    break; // out of loop through primes.
                }
            }
            if (isPrime)
            { // save it!
                primes[numPrimes] = curPrime;
                numPrimes++;
            }
        } // End while

        // Print all the primes out.
        for (int i = 0; i <= numPrimes-1; i++)
        {
            System.out.println ("Prime: " + primes[i]);
        }
    } // end printPrimes

}
```

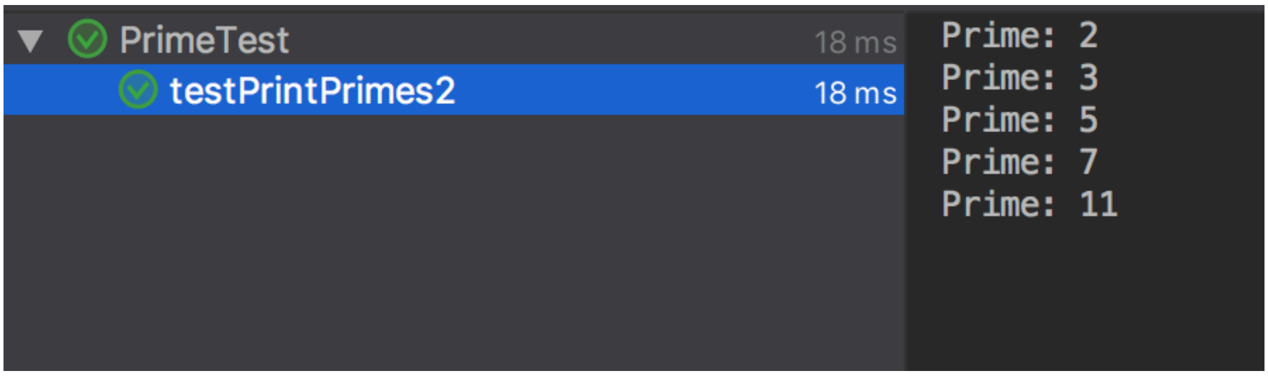
可以发现n=1时未进入while循环体内，仅执行整个代码的50%

2 测试用例为5时运行结果如下

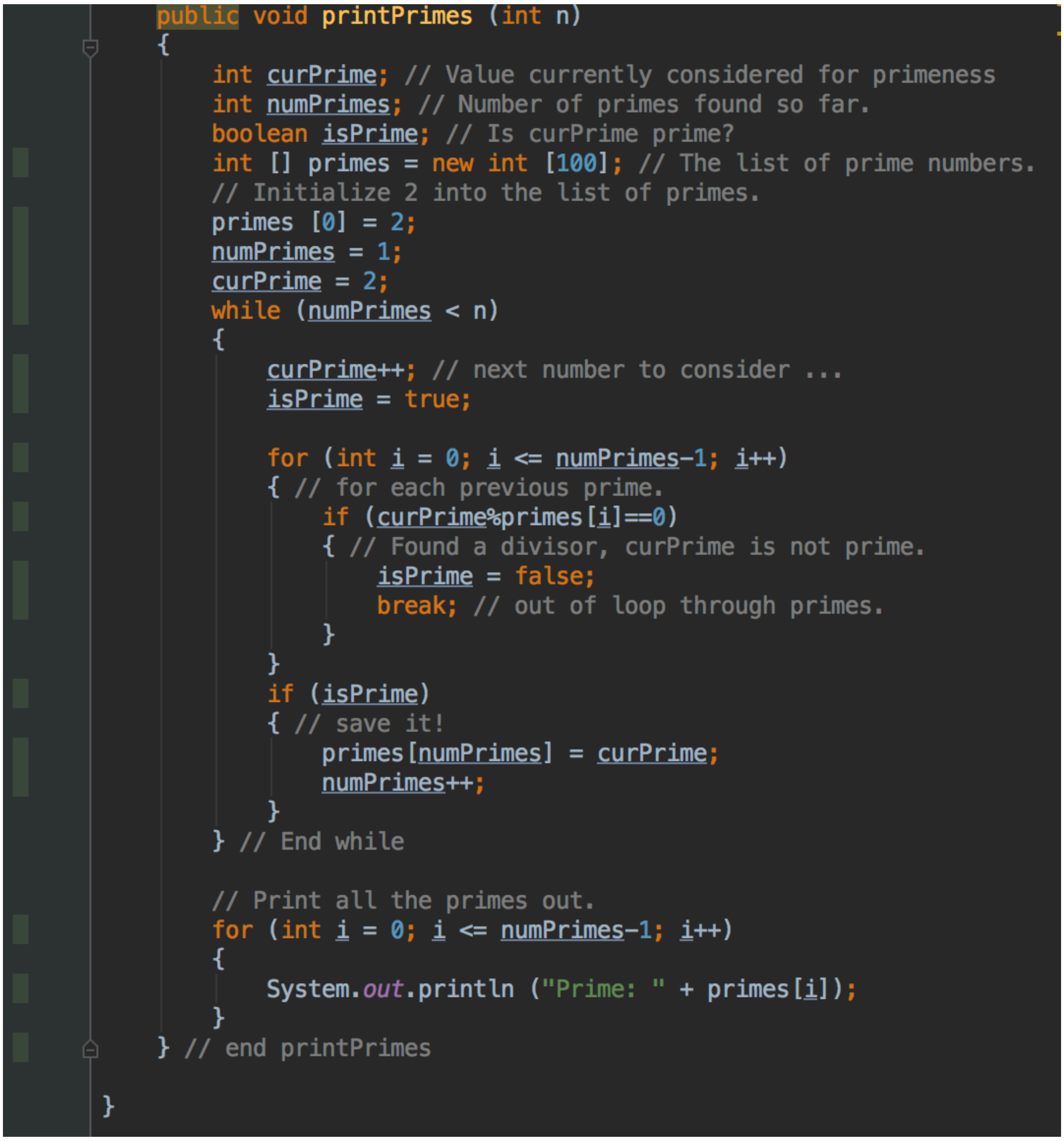


Prime	100% (1/1)	100% (1/1)	100% (18...
PrimeTest	100% (1/1)	50% (1/2)	66% (4/6)

可见当测试用例n=5时，整体代码均被运行，符合覆盖要求。



```
Prime: 2
Prime: 3
Prime: 5
Prime: 7
Prime: 11
```



```
    public void printPrimes (int n)
    {
        int curPrime; // Value currently considered for primeness
        int numPrimes; // Number of primes found so far.
        boolean isPrime; // Is curPrime prime?
        int [] primes = new int [100]; // The list of prime numbers.
        // Initialize 2 into the list of primes.
        primes [0] = 2;
        numPrimes = 1;
        curPrime = 2;
        while (numPrimes < n)
        {
            curPrime++; // next number to consider ...
            isPrime = true;

            for (int i = 0; i <= numPrimes-1; i++)
            { // for each previous prime.
                if (curPrime%primes[i]==0)
                { // Found a divisor, curPrime is not prime.
                    isPrime = false;
                    break; // out of loop through primes.
                }
            }
            if (isPrime)
            { // save it!
                primes[numPrimes] = curPrime;
                numPrimes++;
            }
        } // End while

        // Print all the primes out.
        for (int i = 0; i <= numPrimes-1; i++)
        {
            System.out.println ("Prime: " + primes[i]);
        }
    } // end printPrimes

}
```

可见当测试用例n=5时，整体代码均被运行，符合覆盖要求。