```
public boolean isPrimeNumber1(int number) {
            if(number == 1) {                                   1
                  return false;                                 1
            }
            boolean prime = true;                               1
            for (int i = 2; prime && i < number; i++) {         n-1
                  if(number % i == 0) {                         n-2
                        prime = false;                          n-2
                  }
            }
            return prime;                                       1
      }
```

Complejidad temporal -1+3n = O(n)

```
public boolean isPrimeNumber2(int number)
      {
          boolean prime = true;                                 1
          if(number < 2)                                        1
          {
             prime = false;                                     1
          }
          Else                                                  1
          {
              for(int x = 2; x*x <= number; x++)                ⌊log₂(n)⌋
              {
                  if(number % x == 0){                          ⌊log₂(n)⌋-1
                   prime = false;                               ⌊log₂(n)⌋-1
                   break;                                       ⌊log₂(n)⌋-1
                  }
              }
          }
          return prime;                                         1
      }
```

Complejidad temporal 2+4⌊$log_2(n)$⌋ = O(log(n))

```
public boolean isPrimeNumber3(int n){
            int a = 0;                              1
            for (int i = 1; i <= n; i++) {          n+1
                  if (n % i == 0) {                 n
                        a++;                        n
                  }
            }
            if (a != 2)                             1
                  return false;                     1
            else                                    1
                  return true;                      1
      }
```

Complejidad temporal 6+3n=O(n)