

An aerial photograph of a winding asphalt road that snakes through a lush, green mountain valley. The road features several sharp, hairpin turns and is bordered by concrete walls. The surrounding landscape is steep and covered in dense vegetation. The sun is visible in the upper right corner, casting a warm glow over the scene.

<epam> | hws

Java Bugs and Fun Facts

Endre Ferencz
Endre_Ferencz@epam.com

2015



Java Bugs and Fun Facts

STRING

Substring

```
@Benchmark
public String parseParam() {
    return param.substring(1);
}
```

jdk1.7.0_05

? ns/op

jdk1.8.0_45

? ns/op

VOTE

<http://epa.ms/M1>

Substring

```
@Benchmark
public String parseParam() {
    return param.substring(1);
}
```

jdk1.7.0_05 6.939 ± 0.279 ns/op

jdk1.8.0_45 **38.137 ± 0.894** ns/op

Substring

```
// Package private constructor which shares value array for speed.  
String(int offset, int count, char value[]) {  
    this.value = value;  
    this.offset = offset;  
    this.count = count;  
}
```



```
public String(char value[], int offset, int count) {  
    // [...] Parameter checks  
    this.value = Arrays.copyOfRange(value, offset, offset+count);  
}
```

Hashcode

```
// Length: 1
String shortString = "1";           ? ns/op
// Length: 10
String averageString = "1234567890"; ? ns/op
// Length: 100
String veryLongString = "1234567890..."; ? ns/op
// Length: 100
String veryLongRandomString = "JSwkCyZ3W7IWm6tzJ..."; ? ns/op
// Length: 27
String someString = "electroanalytic exercisable"; ? ns/op
```

```
@Benchmark
public int shortString() {
    return shortString.hashCode();
}
...
```

VOTE
<http://epa.ms/M2>

Hashcode

<code>// Length: 1</code>		
<code>String shortString = "1";</code>	<code>2.669 ± 0.022</code>	<code>ns/op</code>
<code>// Length: 10</code>		
<code>String averageString = "1234567890";</code>	<code>2.682 ± 0.019</code>	<code>ns/op</code>
<code>// Length: 100</code>		
<code>String veryLongString = "1234567890...";</code>	<code>2.658 ± 0.018</code>	<code>ns/op</code>
<code>// Length: 100</code>		
<code>String veryLongRandomString = "JSwkCyZ3W7IWm6tzJ...";</code>	<code>2.646 ± 0.015</code>	<code>ns/op</code>
<code>// Length: 27</code>		
<code>String someString = "electroanalytic exercisable";</code>	<code>23.723 ± 0.249</code>	<code>ns/op</code>

@Benchmark

```
public int shortString() {  
    return shortString.hashCode();  
}  
...
```

Hashcode

Spot the BUG!

```
/** Cache the hash code for the string */
private int hash; // Default to 0

public int hashCode() {
    int h = hash;
    if (h == 0 && value.length > 0) {
        char val[] = value;

        for (int i = 0; i < value.length; i++) {
            h = 31 * h + val[i];
        }
        hash = h;
    }
    return h;
}
```


Implementation **SHOULD NOT** impact API!

```
/**
 * Returns a hash code for this string. The hash code for a
 * String object is computed as
 * 

```
s[0]*31^(n-1) + s[1]*31^(n-2) + ... + s[n-1]
```

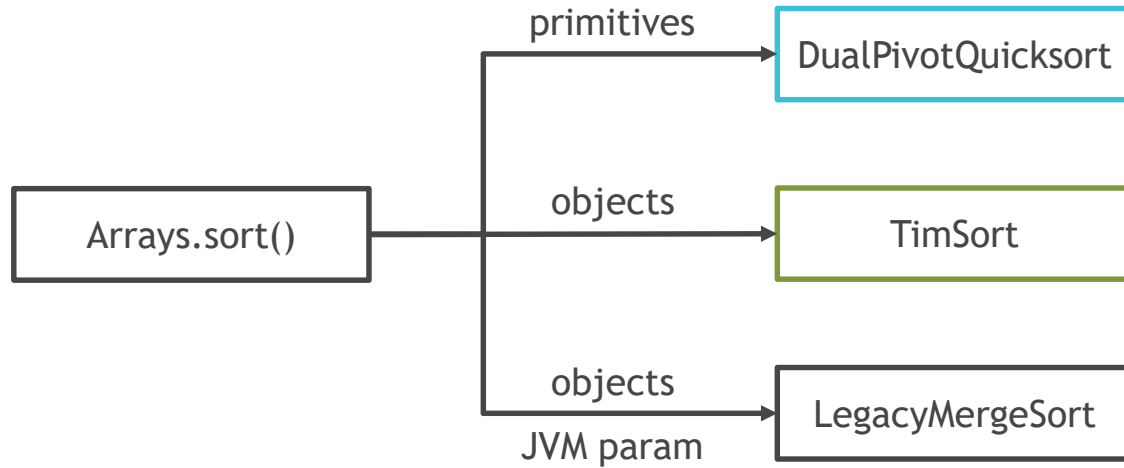

 * using int arithmetic, where s[i] is the
 * ith character of the string, n is the length of
 * the string, and ^ indicates exponentiation.
 * (The hash value of the empty string is zero.)
 *
 * @return a hash code value for this object.
 */
public int hashCode() { ... }
```



Java Bugs and Fun Facts

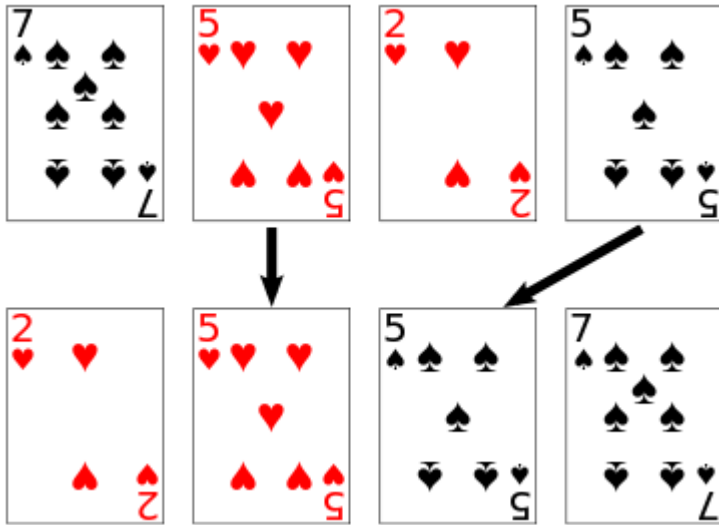
COLLECTIONS

Sort

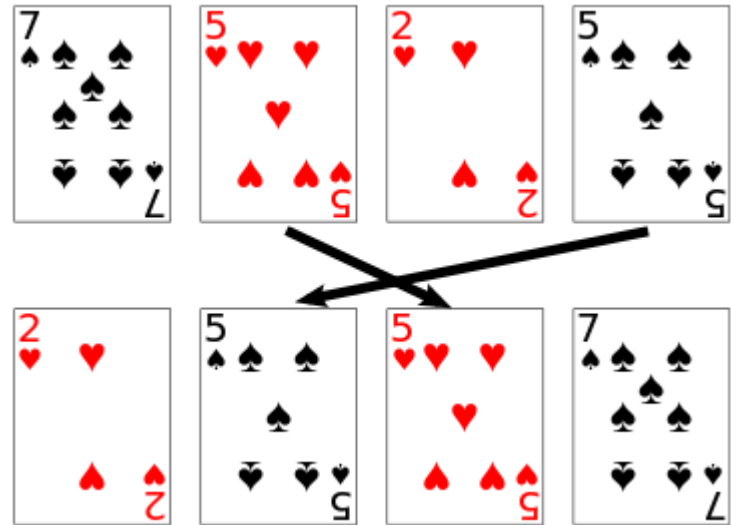


Stable sort

Stable



Not stable





Java Bugs and Fun Facts

PARAMETER PASSING

Parameter passing

```
public void inc1(int number) {  
    number++;  
}  
  
public void inc2(Integer number) {  
    number++;  
}
```

```
// Test code  
int n = 0;  
inc1(n);  
System.out.println(n);
```

VOTE

<http://epa.ms/M3>

Parameter passing

```
public void inc1(final int number) {  
    number++;  
}
```



```
public void inc2(final Integer number) {  
    number++;  
}
```



```
// Test code  
int n = 0;  
inc1(n);  
System.out.println(n);
```

Parameter passing

```
public void inc1(Date date) {  
    date = new Date(date.getTime() + DAY);  
}  
  
public void inc2(Date date) {  
    date.setTime(date.getTime() + DAY);  
}
```

```
// Test code  
Date date = new Date();  
inc1(date);  
System.out.println(date);
```

VOTE
<http://epa.ms/M4>

Parameter passing

```
public void inc1(final Date date) {  
    date = new Date(date.getTime() + DAY);  
}  
  
public void inc2(final Date date) {  
    date.setTime(date.getTime() + DAY);  
}
```



Works, but...

```
// Test code  
Date date = new Date();  
inc1(date);  
System.out.println(date);
```



Java Bugs and Fun Facts

NUMBERS

The magic number

```
// difference is non-trivial.  
// could scale addend by ratio of difference to  
// halfUlp here, if we bothered to compute that difference.  
// Most of the time ( I hope ) it is about 1 anyway.  
dValue += ulp( dValue, overvalue );  
if ( dValue == 0.0 || dValue == Double.POSITIVE_INFINITY )  
    break correctionLoop; // oops. Fell off end of range.  
continue; // try again.
```

sun.misc.FloatingDecimal, JDK 6 Update 23
Excerpt from a very long method (348 lines)

A vibrant street scene in London during the "golden hour" of sunset. The sun is low in the sky, casting a warm, golden glow over the scene. In the foreground, a large, diverse crowd of pedestrians is walking across the street. To the left, a grand, classical-style building with a Union Jack flag flying from a balcony is visible. A street sign on the building reads "GENT STREET W1". In the middle ground, a red flag with the "Superdry" logo is attached to a building. To the right, a Starbucks logo is visible on a building facade. The overall atmosphere is busy and lively.

Java Bugs and Fun Facts

TWEEDLEDUM

Tweedledum

- Provide declarations for the variables `x` and `i` such that

– Correct:

```
x = x + i
```

– Compile error:

```
x += i
```

Tweedledum

- Provide declarations for the variables `x` and `i` such that

– Correct:

```
x = x + i
```

– Compile error:

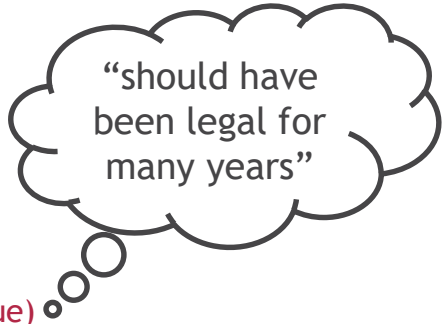
```
x += i
```

Solution (1.4.2 <= Java version < 7)

```
Object x = "Hello ";  
String i = "Java!";
```

- **History**

- 2002-09-02, JDK-4642850 : javac allows Object += String (Fixed)
- 2008-04-10, JDK-6686855 : Section 15.26.2 in JLS 3 incorrectly allows... (Not an issue)
- 2011-05-18, JDK-4741726 : allow Object += String (Fixed)
- 2011-06-30, JDK-7058838 : Changed behavior of compound assignment (+=) (Not an issue)



“should have been legal for many years”

Further reading

- *Java Puzzlers: Traps, Pitfalls, and Corner Cases*

By Joshua Bloch and Neal Gafter

- Effective Java

By Joshua Bloch

- `java.lang.String` Catechism [presentation]

By Aleksey Shipilëv

