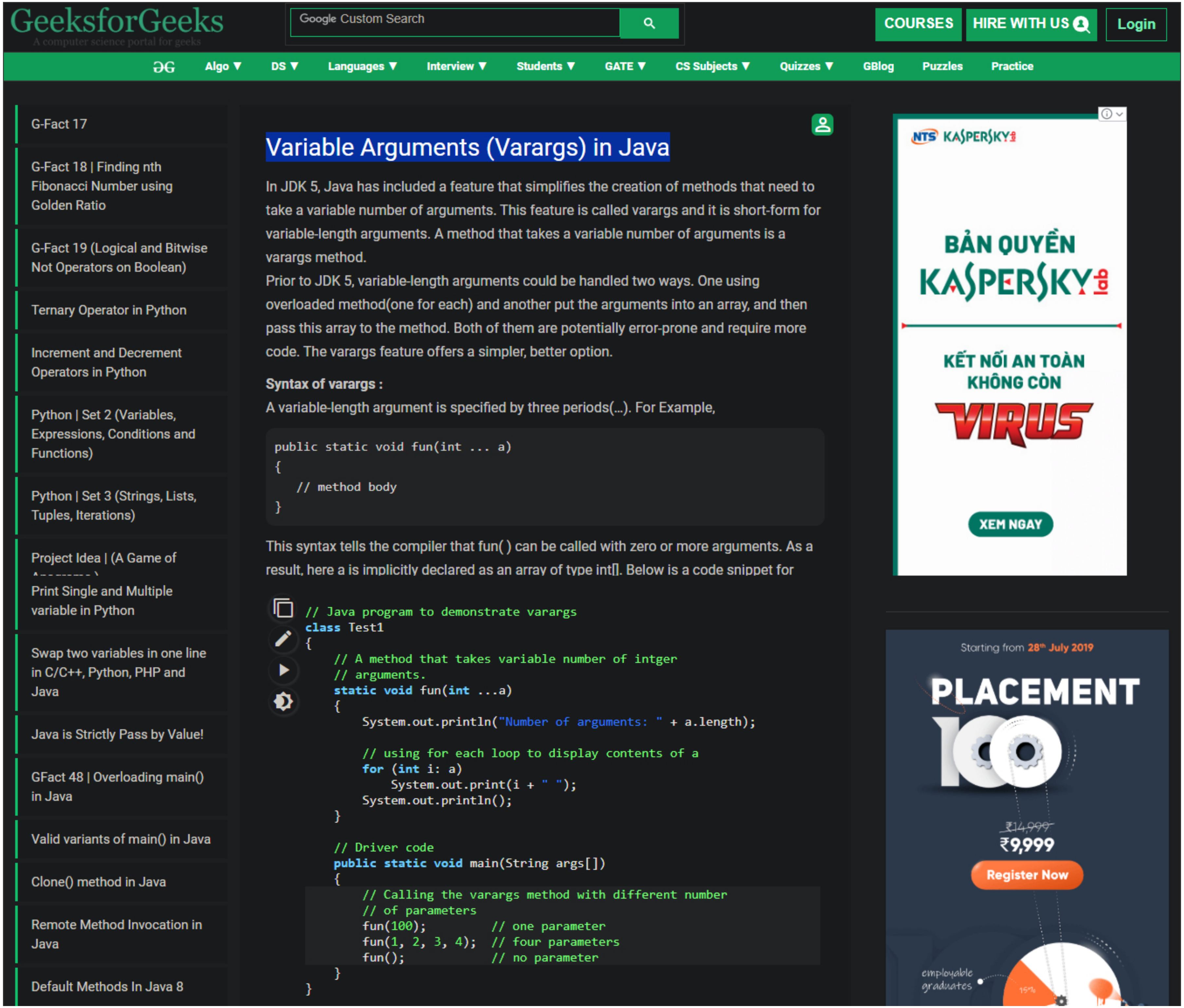


G-Fact 17
G-Fact 18 | Finding nth Fibonacci Number using Golden Ratio
G-Fact 19 (Logical and Bitwise Not Operators on Boolean)
Ternary Operator in Python
Increment and Decrement Operators in Python
Python | Set 2 (Variables, Expressions, Conditions and Functions)
Python | Set 3 (Strings, Lists, Tuples, Iterations)
Project Idea | (A Game of
Print Single and Multiple variable in Python
Swap two variables in one line in C/C++, Python, PHP and Java
Java is Strictly Pass by Value!
GFact 48 | Overloading main() in Java
Valid variants of main() in Java
Clone() method in Java
Remote Method Invocation in Java
Default Methods In Java 8

Variable Arguments (Varargs) in Java

In JDK 5, Java has included a feature that simplifies the creation of methods that need to take a variable number of arguments. This feature is called varargs and it is short-form for variable-length arguments. A method that takes a variable number of arguments is a varargs method.
Prior to JDK 5, variable-length arguments could be handled two ways. One using overloaded method(one for each) and another put the arguments into an array, and then pass this array to the method. Both of them are potentially error-prone and require more code. The varargs feature offers a simpler, better option.

Syntax of varargs :
A variable-length argument is specified by three periods(…). For Example,

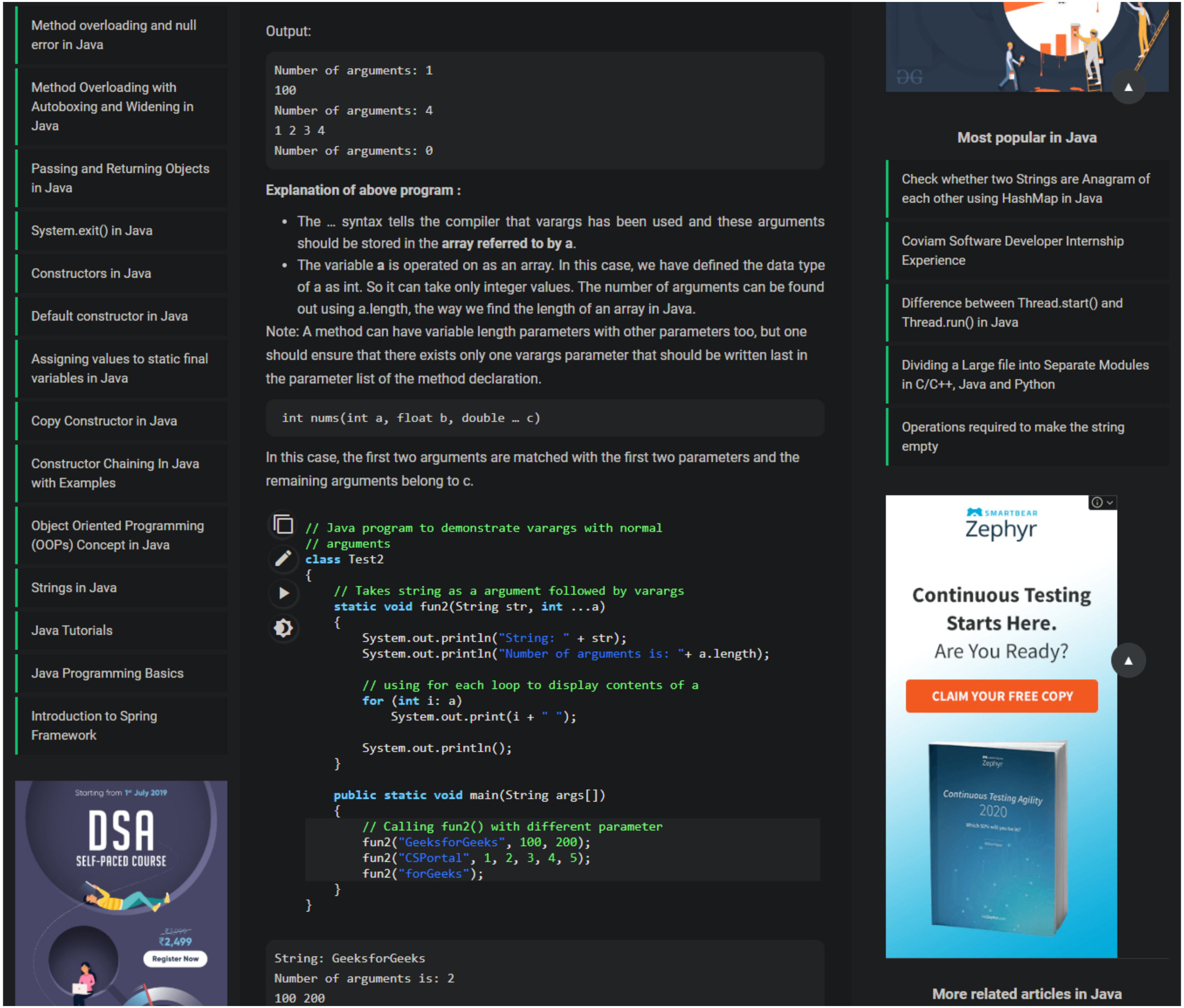
```
public static void fun(int ... a)
{
   // method body
}
```

This syntax tells the compiler that fun( ) can be called with zero or more arguments. As a result, here a is implicitly declared as an array of type int[]. Below is a code snippet for

```
// Java program to demonstrate varargs
class Test1
{
    // A method that takes variable number of intger
    // arguments.
    static void fun(int ...a)
    {
        System.out.println("Number of arguments: " + a.length);

        // using for each loop to display contents of a
        for (int i: a)
            System.out.print(i + " ");
        System.out.println();
    }

    // Driver code
    public static void main(String args[])
    {
        // Calling the varargs method with different number
        // of parameters
        fun(100);         // one parameter
        fun(1, 2, 3, 4);  // four parameters
        fun();            // no parameter
    }
}
```



Method overloading and null error in Java

Method Overloading with Autoboxing and Widening in Java

Passing and Returning Objects in Java

System.exit() in Java

Constructors in Java

Default constructor in Java

Assigning values to static final variables in Java

Copy Constructor in Java

Constructor Chaining In Java with Examples

Object Oriented Programming (OOPs) Concept in Java

Strings in Java

Java Tutorials

Java Programming Basics

Introduction to Spring Framework

Output:

```
Number of arguments: 1
100
Number of arguments: 4
1 2 3 4
Number of arguments: 0
```

Explanation of above program :

- The … syntax tells the compiler that varargs has been used and these arguments should be stored in the **array referred to by a**.
- The variable **a** is operated on as an array. In this case, we have defined the data type of a as int. So it can take only integer values. The number of arguments can be found out using a.length, the way we find the length of an array in Java.

Note: A method can have variable length parameters with other parameters too, but one should ensure that there exists only one varargs parameter that should be written last in the parameter list of the method declaration.

```
int nums(int a, float b, double … c)
```

In this case, the first two arguments are matched with the first two parameters and the remaining arguments belong to c.

```
// Java program to demonstrate varargs with normal
// arguments
class Test2
{
    // Takes string as a argument followed by varargs
    static void fun2(String str, int ...a)
    {
        System.out.println("String: " + str);
        System.out.println("Number of arguments is: "+ a.length);

        // using for each loop to display contents of a
        for (int i: a)
            System.out.print(i + " ");

        System.out.println();
    }

    public static void main(String args[])
    {
        // Calling fun2() with different parameter
        fun2("GeeksforGeeks", 100, 200);
        fun2("CSPortal", 1, 2, 3, 4, 5);
        fun2("forGeeks");
    }
}
```

```
String: GeeksforGeeks
Number of arguments is: 2
100 200
```

Most popular in Java

Check whether two Strings are Anagram of each other using HashMap in Java

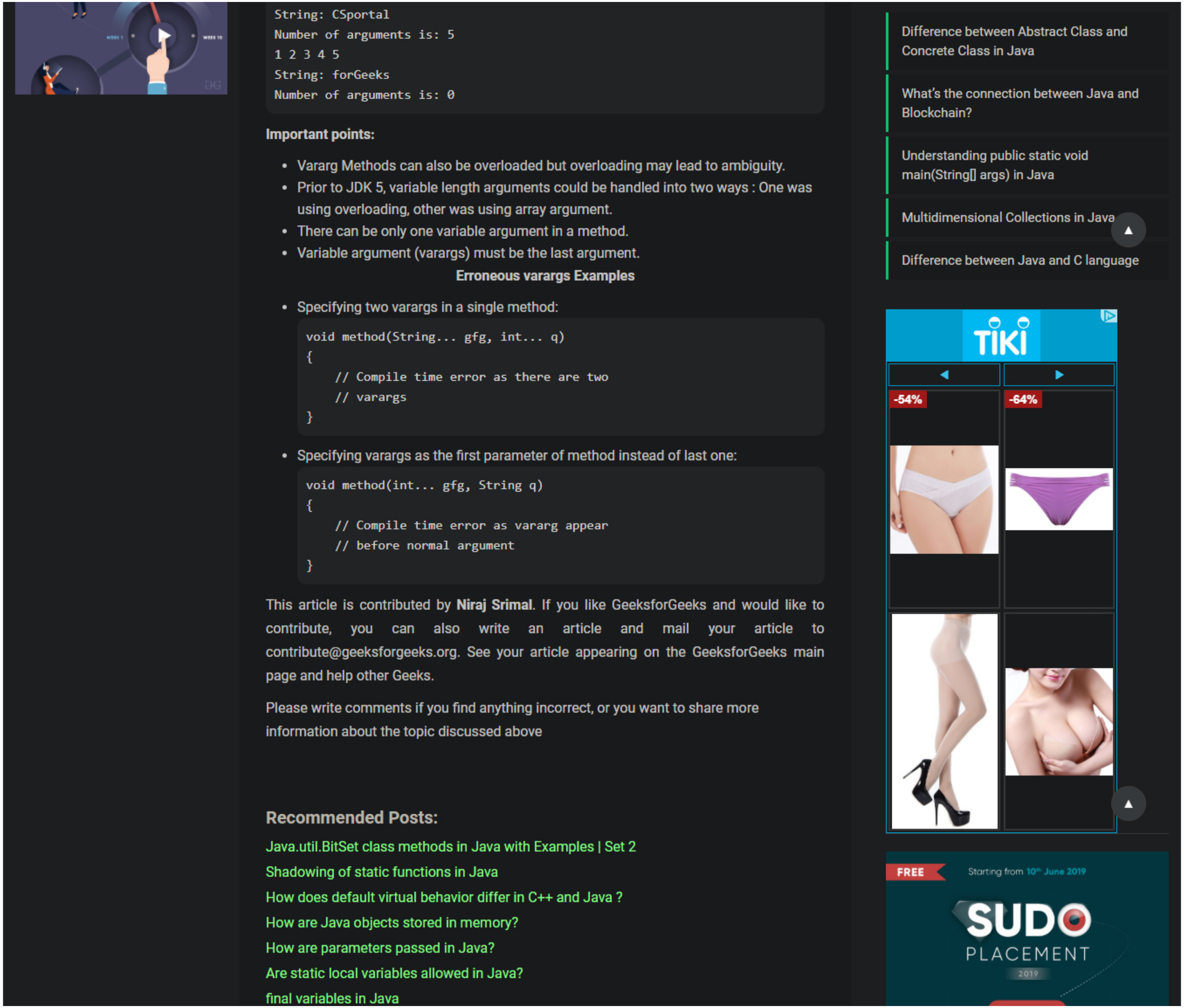
Coviam Software Developer Internship Experience

Difference between Thread.start() and Thread.run() in Java

Dividing a Large file into Separate Modules in C/C++, Java and Python

Operations required to make the string empty

More related articles in Java



```
String: CSportal
Number of arguments is: 5
1 2 3 4 5
String: forGeeks
Number of arguments is: 0
```

Important points:

- Vararg Methods can also be overloaded but overloading may lead to ambiguity.
- Prior to JDK 5, variable length arguments could be handled into two ways : One was using overloading, other was using array argument.
- There can be only one variable argument in a method.
- Variable argument (varargs) must be the last argument.

Erroneous varargs Examples

- Specifying two varargs in a single method:

```
void method(String... gfg, int... q)
{
    // Compile time error as there are two
    // varargs
}
```

- Specifying varargs as the first parameter of method instead of last one:

```
void method(int... gfg, String q)
{
    // Compile time error as vararg appear
    // before normal argument
}
```

This article is contributed by **Niraj Srimal**. If you like GeeksforGeeks and would like to contribute, you can also write an article and mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above

Recommended Posts:

Java.util.BitSet class methods in Java with Examples | Set 2
Shadowing of static functions in Java
How does default virtual behavior differ in C++ and Java ?
How are Java objects stored in memory?
How are parameters passed in Java?
Are static local variables allowed in Java?
final variables in Java

Difference between Abstract Class and Concrete Class in Java
What's the connection between Java and Blockchain?
Understanding public static void main(String[] args) in Java
Multidimensional Collections in Java
Difference between Java and C language

- Default constructor in Java
- Assigning values to static final variables in Java
- Comparison of Exception Handling in C++ and Java
- Does Java support goto?
- Arrays in Java
- Inheritance and constructors in Java
- More restrictive access to a derived class method in Java
- Comparison of static keyword in C++ and Java

Article Tags : Java

Practice Tags : Java

8

□ To-do □ Done

2.3

Based on **24** vote(s)

Feedback/ Suggest Improvement   Add Notes   Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Previous
Reflection in Java

Next
Using underscore in Numeric Literals in Java

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

GeeksforGeeks
A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org

COMPANY
About Us
Careers
Privacy Policy
Contact Us

LEARN
Algorithms
Data Structures
Languages
CS Subjects
Video Tutorials

PRACTICE
Courses
Company-wise
Topic-wise
How to begin?

CONTRIBUTE
Write an Article
Write Interview Experience
Internships
Videos