Varargs

Varargs tek method'a istediğimiz kadar parametre yollayarak sonuç almamizi sağlar.
Yani parametre sayımiz değişken ancak method'un yapacağı iş sabitse Varargs kullanarak tek method'la kodumuzu yazabiliriz

Varargs ozellik olarak list gibi calisir (uzunlugu esnektir) fakat varargs'in arka planinda Java Array ile calisir.

Varargs'i declare etmek icin data type'dan sonra ... kullaniriz. (int... sayi vb..)

Bir method'da varargs disinda parametre varsa varargs parametre olarak en sona yazilmalidir. (aksi durumda varargs nerede duracagini bilemez)

Bir method'da sadece 1 varargs kullanilabilir

```
public static void main(String[] args) {


    add(5,7); //12
    add(5,7,-15); //-3
    add(5,7,-15,20); // 17
    add(5,7,-15,20,30); // 47
}

public static void add(int... sayi ) {

    int toplam=0;
    for (int i : sayi) {
        toplam+=i;
    }
    System.out.println("girilen sayilarin toplami : "+ toplam);
}
```

Varargs

Which of the following compile? (Choose all that apply)

A. public void moreA(int... nums) {}

B. public void moreB(String values, int... nums) {}

C. public void moreC(int... nums, String values) {}

D. public void moreD(String... values, int... nums) {}

E. public void moreE(String[] values, ...int nums) {}

F. public void moreF(String... values, int[] nums) {}

G. public void moreG(String[] values, int[] nums) {}

Varargs

```
public class Go {

public static void main(String[] args) {
        new Go().Go(1,"hello");
        new Go().Go(2,"hello","hi");
}
                                TopJavaTutorial.com

public void Go(int x,String... y){
        System.out.print(y[y.length-x]+ " ");
}

}
```

String Builder

- StringBuilder “mutable” yani değiştirilebilir String elde etmemize olanak tanır.
- Böylece hafızada her seferinde yeni bir alan açılmadan var olan alan üzerinde değişiklik yapılabilir. Bu da StringBuilder sınıfını hafıza kullanımı olarak String sınıfının önüne geçirir.
- StringBuilder thread-safe değildir. Yani synchronized değildir. Thread’li bir işlem kullanılacaksa StringBuilder kullanılması güvenli değildir.
- Not: StringBuffer, StringBuilder’a benzer. StringBuilder, StringBuffer’dan hizlidir. Multi-thread icin StringBuffer kullanilir.

String Builder

1) StringBuilder sb1 = new StringBuilder() ===> Bos bir StringBuilder olusturur

2) StringBuilder sb2 = new StringBuilder("animal"); ===> Belli bir degeri olan StringBuilder olusturur

3) StringBuilder sb3 = new StringBuilder(5); ===> Ilk uzunlugu tahmin edilen bir StringBuilder olusturur.

```
StringBuilder sb = new StringBuilder(5);
```

0	1	2	3	4

```
sb.append("anim");
```

a	n	i	m	
0	1	2	3	4

```
sb.append("als");
```

a	n	i	m	a	l	s		
0	1	2	3	4	5	6	7	...

String Builder

1) append( ); StringBuilder'a ekleme yapar

```
public static void main(String[] args) {

    StringBuilder sb = new StringBuilder();
    sb.append("Mehmet");
        System.out.println(sb); // Mehmet
    sb.append(" Hoca");
        System.out.println(sb); // Mehmet Hoca
    sb.append(" Java"). append(" anlatir.");
        System.out.println(sb);
       // Mehmet Hoca Java anlatir.
       // append ile arka arkaya ekleme yapilabilir
    sb.append("Java cok guzel",0,4);
        System.out.println(sb);
       // Stringin tumu degil bir bolumu eklenebilir
       // Mehmet Hoca Java anlatir.Java
}
```

String Builder

2) length( ); StringBuilder'in uzunlugunu verir

```
StringBuilder sb = new StringBuilder();
sb.append("Mehmet");
System.out.println(sb.length());  // 6
```

3) capacity( );StringBuilder'un kapasitesini verir

```
public static void main(String[] args) {

    StringBuilder sb = new StringBuilder(5);

    System.out.println(sb.length()); // 0
    System.out.println(sb.capacity()); // 5

    sb.append("Kemal"); // Kemal
    System.out.println(sb.length()); // 5
    System.out.println(sb.capacity()); // 5

    sb.append(" Can"); // Kemal Can
    System.out.println(sb.length()); // 9
    System.out.println(sb.capacity()); // 12

}
```

String Builder

4) charAt( );StringBuilder'da istenen index'deki karakteri verir

5) delete(4,7); StringBuilder'da istenen index'ler arasindaki karakterleri siler.

6) deleteCharAt(7); StringBuilder'da istenen index'deki tek karakteri siler

7) equals( );Iki StringBuilder'in degerlerinin karsilastirir.

NOT 1: equals( ) method'unda parantez icine String yazarsak hata vermez ama false doner.

NOT 2: equals( ) method'u == gibi calisir

String Builder

8) indexOf( );StringBuilder'da istenen karakterin index'ini verir.

9) insert(3, "Java "); StringBuilder'da istenen indexden baslayarak istenen karakteri ekler.

10) insert(3, "Java ",1,2); StringBuilder'da istenen indexden baslayarak verilen String'in istenen parcasini ekler.

11) replace(3, 8, " Ali ");

StringBuilder'da istenen index'ler arasindaki bolumun yerine verilen String'i ekler.

String Builder

12) reverse( ); StringBuilder'i tersine cevirir.

13) setCharAt(3, 'k'); StringBuilder'da istenen index'deki karakteri istedigimiz karakter yapar.

14) subSequence(3,7); StringBuilder'da istenen indexler arasindaki karakterleri dondurur.

15) toString( ); StringBuilder'i String'e cevirir.

toString( )'den sonra nokta koyup String method'lari kullanilabilir.

String Builder

16) trimToSize( ); StringBuilder'in capacity ile length'ini esitler.

17) compareTo( ); 2 StringBuilder'in tum karakterlerinin esitligini kontrol eder. (0 ise esit)

Soru : For loop ile 1000 defa bir islem yapalim. Oncesinde ve sonrasinda zamani kontrol edip StringBuilder ve String class'larinin performanslarini karsilastiralim.

Ipucu: long TimeSb = System.nanoTime(); kullanalim

String Builder

Soru 1:

What is the result of the following code?

```
7: StringBuilder sb = new StringBuilder();
8: sb.append("aaa").insert(1, "bb").insert(4, "ccc");
9: System.out.println(sb);
```

A. abbaaccc

B. abbaccca

C. bbaaaccc

D. bbaaccca

E. An exception is thrown.

F. The code does not compile.

String Builder

Soru 2:

What is the result of the following code?

```
2: String s1 = "java";
3: StringBuilder s2 = new StringBuilder("java");
4: if (s1 == s2)
5:  System.out.print("1");
6: if (s1.equals(s2))
7:  System.out.print("2");
```

A. 1

B. 2

C. 12

D. No output is printed.

E. An exception is thrown.

F. The code does not compile.

String Builder

Soru 3 :

Which are the results of the following code? (Choose all that apply)

```
String numbers = "012345678";
System.out.println(numbers.substring(1, 3));
System.out.println(numbers.substring(7, 7));
System.out.println(numbers.substring(7));
```

A. 12

B. 123

C. 7

D. 78

E. A blank line.

F. An exception is thrown.

G. The code does not compile.

String Builder

Soru 4:

What is the result of the following code?

```
4: int total = 0;
5: StringBuilder letters = new StringBuilder("abcdefg");
6: total += letters.substring(1, 2).length();
7: total += letters.substring(6, 6).length();
8: total += letters.substring(6, 5).length();
9: System.out.println(total);
```

A. 1

B. 2

C. 3

D. 7

E. An exception is thrown.

F. The code does not compile.