

كلية الدراسات المتوسطة - الأزهر
College of Intermediate Studies-Alazhar



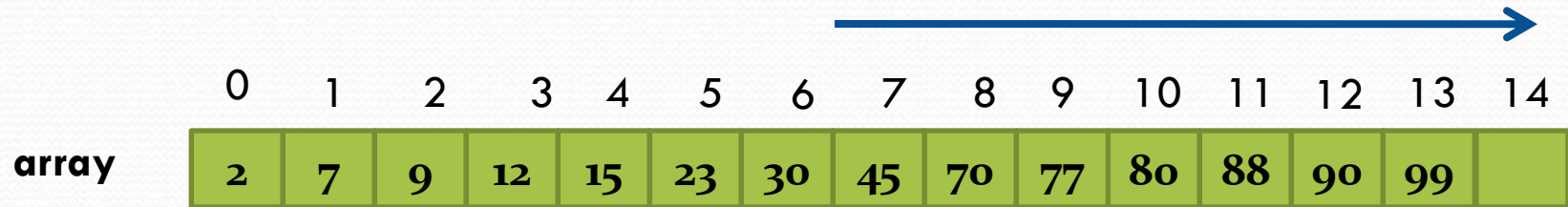
تراكيب البيانات

مدرس المساق/أ. م. رائد خضير

الوحدة الأولى
أساسيات لغة البرمجة الجافا

المادة العلمية إعداد
أ. م. رائد خضير

إضافة عنصر دون تأثير



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
array	2	7	9	12	15	23	30	45	70	77	80	88	90	99	

To insert value

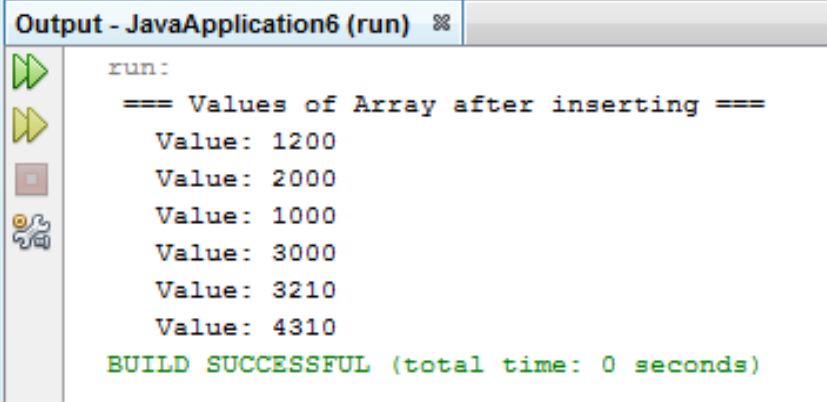
13

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
array	2	7	9	12	13	15	23	30	45	70	77	80	88	90	99

الشكل العلوي يوضح طريقة إضافة عنصر في مصفوفة مرتبة العناصر حسب القيمة بشكل تصاعدي

إضافة عنصر في مصفوفة غير مرتبة دون تأثير

```
1. static void Main(string[] args)
2.     {
3.         int[] salary = new int [6];
4.         salary[0] = 1200;
5.         salary[1] = 2000;
6.         salary[2] = 1000;
7.         salary[3] = 3210;
8.         salary[4] = 4310;
9.         // to add value 3000 in the position 3
10.        for (int i = salary.Length - 1; i >= 3; i--)
11.            salary[i] = salary[i - 1];
12.        salary[3] = 3000;
13.        system.out.println(" === Values of Array after inserting === ");
14.        for (int j = 0; j < salary.Length; j++)
15.            system.out.println(" Value: "+salary[j]);
16.    }
```



Output - JavaApplication6 (run) %

```
run:
=== Values of Array after inserting ===
Value: 1200
Value: 2000
Value: 1000
Value: 3000
Value: 3210
Value: 4310
BUILD SUCCESSFUL (total time: 0 seconds)
```

إضافة عنصر في مصفوفة مرتبة دون تأثير

```
1. static void Main(string[] args) {
2.     int[] salary = new int[6];
3.     salary[0] = 100;
4.     salary[1] = 200;
5.     salary[2] = 300;
6.     salary[3] = 1210;
7.     salary[4] = 2310;
8.     // To add value 250
9.     int i;
10.    for (i = 0; i < salary.Length; i++)
11.        if (salary[i] > 250)
12.            break;
13.    for (int j = salary.Length - 1; j >= i; j--)
14.        salary[j] = salary[j + 1];
15.    salary[i] = 250;
16.    system.out.println(" ");
17.    system.out.println("=== Values of Array after inserting ===");
18.    for (int j = 0; j < salary.Length; j++)
19.        system.out.println(" Value: " + salary[j]);
20. }
```

```
=== Values of Array after inserting ===
Value: 100
Value: 200
Value: 250
Value: 300
Value: 1210
Value: 2310
```

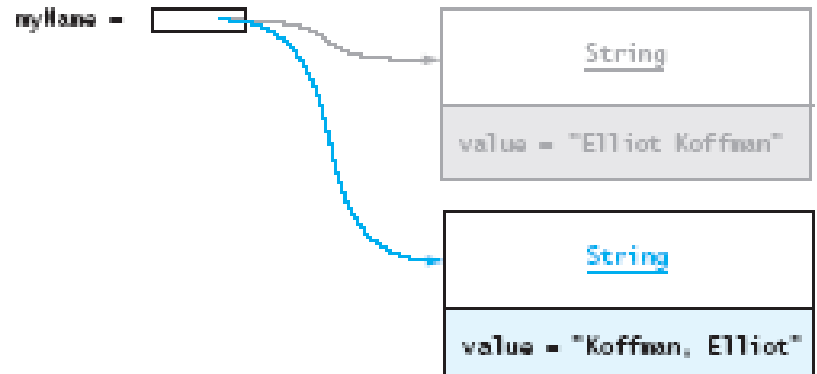

String

- النص في لغة الجافا عبارة عن مصفوفة من نوع حرف. وهي عبارة عن Objects من نوع نص.
- بمجرد ان يتم انشاء المتغير من نوع نص (String) يكون غير قابل للتعديل.
- لتحصل على نص قابل للتغيير استخدم النوع StringBuffer.
- لإنشاء متغير من نوع نص يوجد (constructor) افتراضي يمكننا من انشاء متغير من نوع String فارغ حسب الصيغة التالية.

```
String s = new String();
```

FIGURE A.4

Old and New Strings
Referenced by myName



إنشاء متغير من نوع String

- `String str = "abc";` is equivalent to:

```
char data[] = {'a', 'b', 'c'};
```

```
String str = new String(data);
```

- إذا حدث تغيير على المصفوفة السابقة `data[]` لن يؤثر على المتغير `.str`

- من الممكن إنشاء متغير من نوع `String` بتمرير قيم متغير آخر من نفس النوع في حملة الانشاء كما هوة موضح في المثال التالي.

```
String str2 = new String(str);
```


العمليات على String

- عملية حساب حجم النص وهي `length()` وتقم بإرجاع طول النص.
5 الناتج `// System.out.println("Hello".length());` مثال
- الرمز `+` يستخدم لعمل ضم نصين او أكثر.
- مثال: `String myname = "Harry"`
`String str = "My name is" + myname + ".";`
- نفس العمل `+` تقوم به `concat()` تضيف النص المضاف الى نهاية النص الاصيل. اذا كان المتغير الاول فارغ ترجع العملية بالنص المضاف.

```
public String concat(String str)
"to".concat("get").concat("her") returns
"together"
```

العمليات على String

- الحروف الموجودة في متغير من نوع String تسترجع باستخدام رقمها الفهرسي.

`public char charAt(int index)`

- ارقام الفهرس تبدأ من 0 الى `length() - 1` كما في فهرس المصفوفات الاحادية.

```
char ch ;
```

```
ch = "abc".charAt(1); // ch = "b"
```

- فحص عملية التساوي `equals()` : ناتج الفحص يكون True or False

`public boolean equals (Object anObject)`

`str.equals("xyz")`

`str.equalsIgnoreCase("xyz")` لتجاهل اذا كان حرف كبير او صغير.

العمليات على String

- **startsWith()** - فحص اذا كان النص يبدأ بحرف او نص.

```
public boolean startsWith(String prefix)  
"Figure".startsWith("Fig"); // true
```

- **endsWith()** - فحص اذا كان النص ينتهي بحرف او نص.

```
public boolean endsWith(String suffix)  
"Figure".endsWith("re"); // true
```

العمليات على String

indexOf - تبحث عن اول ظهور للحرف لو النص المطلوب وترجع رقمه المفهرس ان وجد او -1 اذا لم يكن موجود.

تعيد فهرس - **indexOf**(int ch) public int
الحرف المطلوب

تعيد - **indexOf**(String str) public int
فهرس النص المطلوب ايجداه.

```
String str = "How was your day today?";  
System.out.println(str.indexOf('t'));  
System.out.println(str.indexOf("was"));
```

1. **indexOf**(String str, int fromIndex)
2. **indexOf**(String str , int fromIndex)

العمليات على String

- `lastIndexOf()` تبحث عن آخر ظهور للحرف او النص.
- `substring()` ترجع جزء من النص الاصلي كنص جديد, النص الجديد يبدأ من رقم الفهرس المدخل وينتهي بأخر حرف في النص القديم.

```
public String substring(int beginIndex)
```

Eg: "unhappy".substring(2) returns "happy«

- `public String substring(int beginIndex, int endIndex)`

Eg: "smiles".substring(1, 5) returns "mile"

العمليات على String

- `replace()` – ترجع نص جديد مستبدل به الحرف المراد استبداله بالحرف الآخر.

```
public String replace(char oldChar,  
    char newChar)
```

```
"mesquite in your  
cellar".replace('e', 'o') returns  
"mosquito in your collar"
```

العمليات على String

- **trim()** - ترجع نسخة من النص مزال منه الفراغات في البداية والنهاية.

```
public String trim()
```

```
String s = "  Hi Mom!  ".trim();
```

```
S = "Hi Mom!"
```

- **toLowerCase()** - تحول جميع الحروف الى الشكل الصغير.

- **toUpperCase()** - تحول جميع الحروف الى الشكل الكبير.

```
public String toLowerCase()
```

```
public String toUpperCase()
```

مثال: `"HELLO THERE".toLowerCase();`

`"hello there".toUpperCase();`