continuing example from last time:

\[ \text{list.length} \]

evaluates to 10 in this case.

array indices range:

\[ 0 \leq \text{index} \leq \text{list.length} - 1 \]

equivalently:

\[ 0 \leq \text{index} < \text{list.length} \]
\[
\text{ex.} \quad i := 1, j := 0
\]
\[
\begin{array}{c}
15 \\
14 \\
13 \\
12 \\
11 \\
10 \\
9 \\
8 \\
7 \\
6 \\
5 \\
4 \\
3 \\
2 \\
1 \\
0
\end{array}
\]
\[
\text{int } i = 2; j = 5;
\]
\[
\text{int } [i + j] = -15
\]
\[
\text{any int expression within range}
\]
Ex. `ArrayEx1.java`

Ex. `ArrayEx2.java`

```
list

0
1
2
```

Shortcut: declare, allocate and initialize an array:

```
int[] list = {0, 3, 6, 9, 12};
```

Shortcut for new
Note array types are always reference types.

Primitive variables:

```java
int a = 6;
```

Reference variables:

```java
int[] A = {5, 6, 7};
```
In general a reference variable stores address of some object

```
Reference Variable
```

```
Some Object
```

Ex. String word = "happy"

```
word
```

```
String Object
```

```
\['h', 'a', 'p', 'p', 'y'\]
```
Two categories of data types

- **Primitive**: byte, short, int, long, char, float, double, boolean

- **Reference**: all others
  - String, Scanner, ...
  - int, double, ..., String[]...

Note base type can be primitive

Or base type can be reference
Ex. Semantics of Primitive vars.
\[
\text{int } a = 6, \ b;
\]
\[
b = a;
\]
\[
b = 7;
\]
\[
\text{System.out.println()} \; // \; 6
\]

Ex. Semantics of Reference vars.
\[
\text{int } \{ \} \ A = \{ 5, 6, 7 \}
\]
\[
\text{int } \{ \} \ B;
\]
\[
\text{int } \{ \} \ C;
\]
\( R = A \)

\[ R[2] = 8 \]

```
System.out.println(A[23]);
// output: 8
```
This works since we pass array by reference.
copying arrays:

Ex Array Ex 4.java