Java: Strings, Wrapper Classes, and Arrays

Shirley B. Chu

June 19, 2020

De La Salle University
College of Computer Studies

Strings in Java

- A string is a series of characters.

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
    - A `String` object is immutable, i.e. once created, its value cannot be changed.

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
  - A `String` object is immutable, i.e. once created, its value cannot be changed.
  - Two ways to create a `String` object.

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
  - A `String` object is immutable, i.e. once created, its value cannot be changed.
  - Two ways to create a `String` object.
    - by assigning a literal;
    - by using the keyword `new`

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
    - A `String` object is immutable, i.e. once created, its value cannot be changed.
    - Two ways to create a `String` object.
        - by assigning a literal;
          ```
          String strWord = "Good day!";
          ```
        - by using the keyword `new`

Strings in Java

- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
    - A `String` object is immutable, i.e. once created, its value cannot be changed.
    - Two ways to create a `String` object.
        - by assigning a literal;
          ```
          String strWord = "Good day!";
          ```
        - by using the keyword `new`
          ```
          String strWord = new String ("Good day!");
          ```

Strings in Java

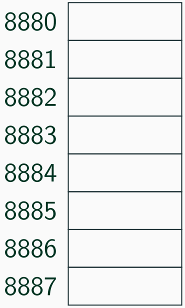
- A string is a series of characters.
- To represent a string in Java, the reference type `String` can be used.
    - A `String` object is immutable, i.e. once created, its value cannot be changed.
    - Two ways to create a `String` object.
        - by assigning a literal;
          `String strWord = "Good day!";`
        - by using the keyword `new`
          `String strWord = new String ("Good day!");`
    - To compare whether two `String` objects have the same value, use methods `equals()`, `compareTo()`, `equalsIgnoreCase()`, `compareToIgnoreCase()`.

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

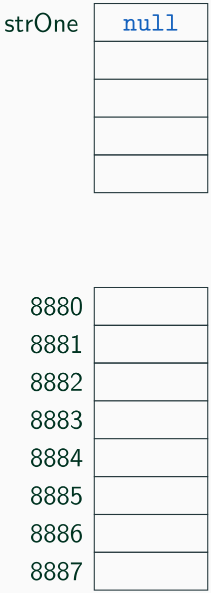
Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```



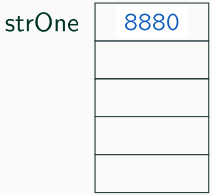
Code the following, and analyze.

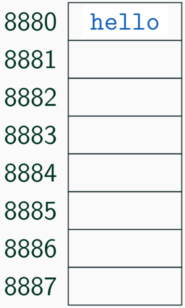
```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```



Code the following, and analyze.

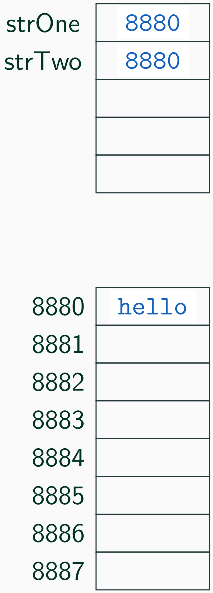
```java
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```





Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```



Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882

8880	hello
8881	
8882	hello
8883	
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```java
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```java
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Code the following, and analyze.

```
String strOne;
System.out.println (strOne);
strOne = "hello";
String strTwo = "hello";
String strThree = new String ("hello");
String strFour = new String ("hello");
String strFive = "hello";

System.out.println (strOne.equals (strTwo));
System.out.println (strOne.equals (strThree));
System.out.println (strThree.equals (strFour));
System.out.println (strThree.equals (strFive));

System.out.println (strOne == strTwo);
System.out.println (strOne == strThree);
System.out.println (strThree == strFour);
System.out.println (strThree == strFive);
System.out.println (strTwo == strFive);
```

strOne	8880
strTwo	8880
strThree	8882
strFour	8883
strFive	8880

8880	hello
8881	
8882	hello
8883	hello
8884	
8885	
8886	
8887	

Wrapper Classes

A **wrapper class**

- is a class whose object contains a primitive data type.

Wrapper Classes

A **wrapper class**

- is a class whose object contains a primitive data type.
- is used for converting primitive types to reference types.

Wrapper Classes

A **wrapper class**

- is a class whose object contains a primitive data type.
- is used for converting primitive types to reference types.
- provides several methods to convert primitive type to `String`, and vice versa.

```
int x = Integer.parseInt ("12345");
```

Wrapper Classes

Primitive types in Java have their corresponding wrapper classes.

Primitive Type	**Wrapper Class**
`boolean`	`Boolean`
`char`	`Character`
`int`	`Integer`
`double`	`Double`
`byte`	`Byte`
`short`	`Short`
`long`	`Long`
`float`	`Float`

Arrays in Java

- dynamically created objects
- hold a fixed number of values of a single type
- length is established at creation
- the number of brackets determine depth of the array

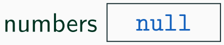
Using arrays

- declaration

Using arrays

- declaration

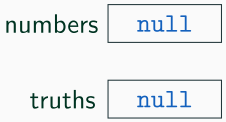
  ```
  int[] numbers;
  ```



Using arrays

- declaration

  ```
  int[] numbers;
  boolean[] truths;
  ```

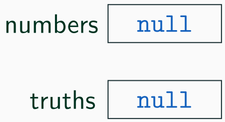


Using arrays

- declaration

  ```
  int[] numbers;
  boolean[] truths;
  ```

- dynamically allocated



Using arrays

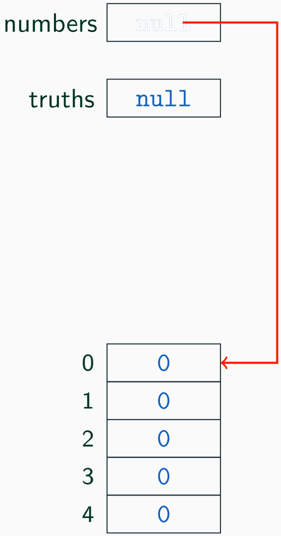
- declaration

  `int[] numbers;`

  `boolean[] truths;`

- dynamically allocated

  `numbers = new int[5];`



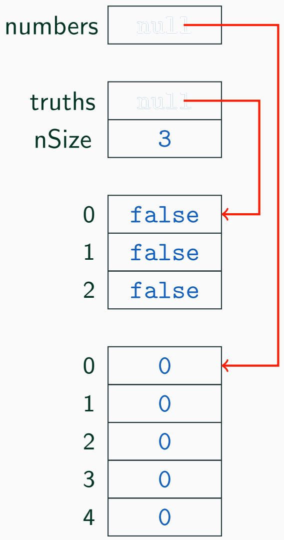
Using arrays

- declaration

```
int[] numbers;
boolean[] truths;
```

- dynamically allocated

```
numbers = new int[5];
int nSize = 3;
truths = new boolean[nSize];
```



Using arrays

- declaration

  `int[] numbers;`

  `boolean[] truths;`

- dynamically allocated

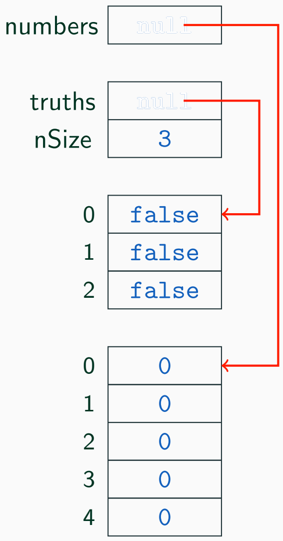
  `numbers = new int[5];`

  `int nSize = 3;`

  `truths = new boolean[nSize];`

- `length` attribute

  `numbers.length or truths.length`



Using arrays

- declaration

  ```
  int[] numbers;
  boolean[] truths;
  ```

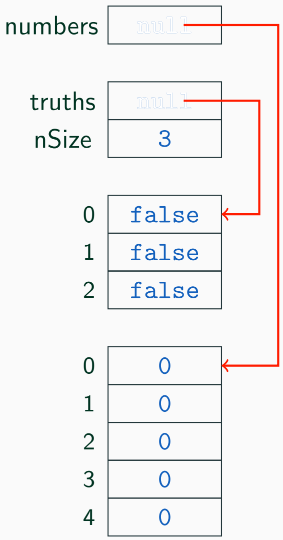
- dynamically allocated

  ```
  numbers = new int[5];
  int nSize = 3;
  truths = new boolean[nSize];
  ```

- `length` attribute

  `numbers.length or truths.length`

- access



Using arrays

- declaration

```
int[] numbers;
boolean[] truths;
```

- dynamically allocated

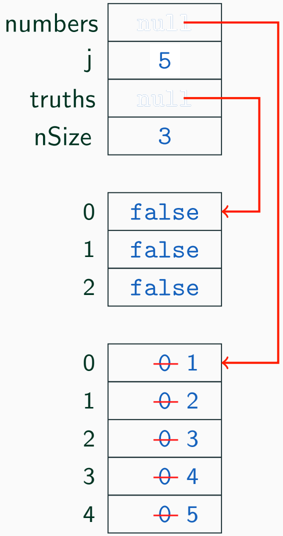
```
numbers = new int[5];
int nSize = 3;
truths = new boolean[nSize];
```

- `length` attribute

```
numbers.length or truths.length
```

- access

```
int j;
for (j = 0; j < numbers.length; j++)
    numbers[j] = j + 1;
```



Using arrays

- declaration

```
int[] numbers;
boolean[] truths;
```

- dynamically allocated

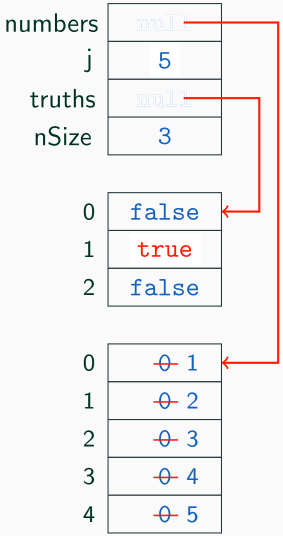
```
numbers = new int[5];
int nSize = 3;
truths = new boolean[nSize];
```

- `length` attribute

```
numbers.length or truths.length
```

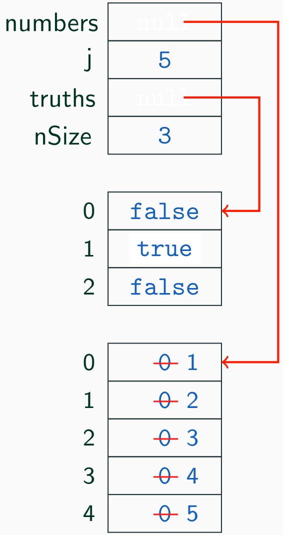
- access

```
int j;
for (j = 0; j < numbers.length; j++)
    numbers[j] = j + 1;

truths[1] = true;
```



Using arrays

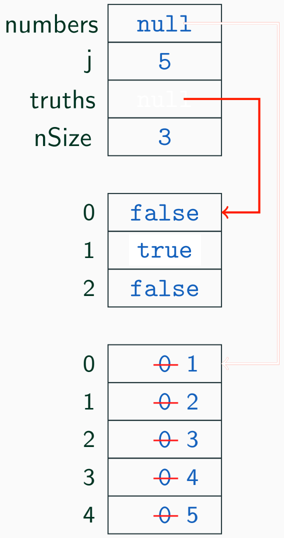
- assigning `null`



Using arrays

- assigning `null`

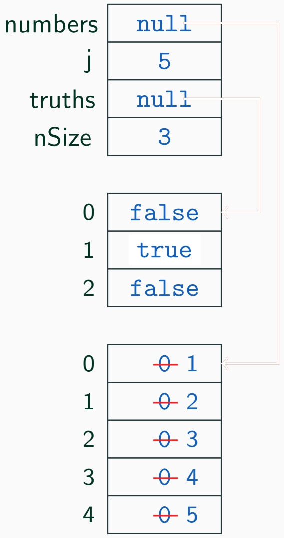
```
numbers = null;
```



Using arrays

- assigning `null`

```
numbers = null;
truths = null;
```

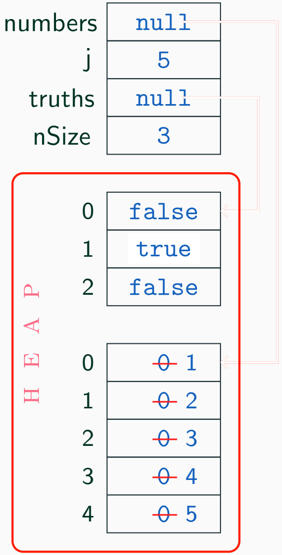


Using arrays

- assigning `null`

```
numbers = null;
truths = null;
```

- All objects are created in the Heap section of the memory.

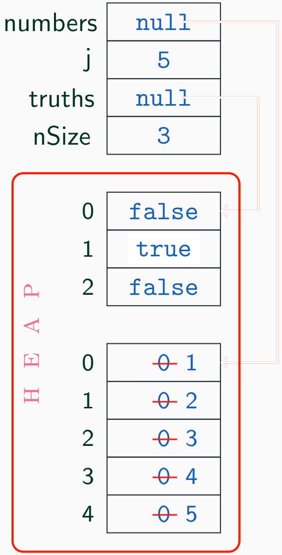


Using arrays

- assigning `null`

```
numbers = null;
truths = null;
```

- All objects are created in the Heap section of the memory.
- Java's automatic garbage collection

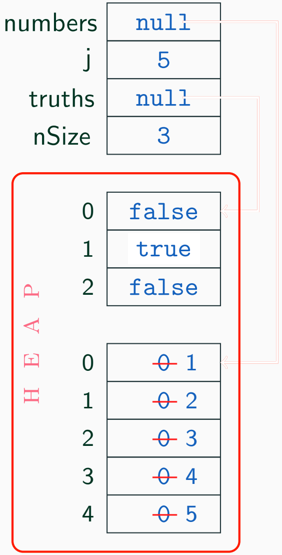


Using arrays

- assigning `null`

  ```
  numbers = null;
  truths = null;
  ```

- All objects are created in the Heap section of the memory.
- Java's automatic garbage collection
  - the process of looking at the heap memory, identifying objects that are in use and those that are not, and removing the unused objects

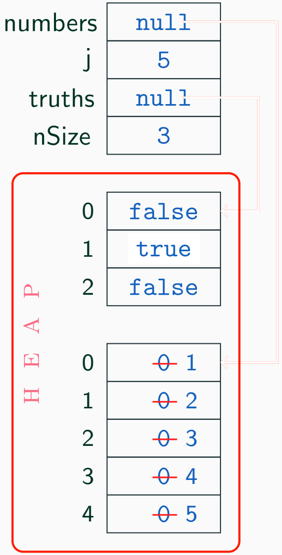


Using arrays

- assigning `null`

```
numbers = null;
truths = null;
```

- All objects are created in the Heap section of the memory.
- Java's automatic garbage collection
  - the process of looking at the heap memory, identifying objects that are in use and those that are not, and removing the unused objects
  - does not happen instantly

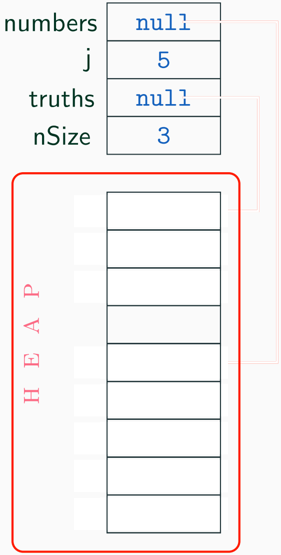


Using arrays

- assigning `null`

```
numbers = null;
truths = null;
```

- All objects are created in the Heap section of the memory.
- Java's automatic garbage collection
  - the process of looking at the heap memory, identifying objects that are in use and those that are not, and removing the unused objects
  - does not happen instantly



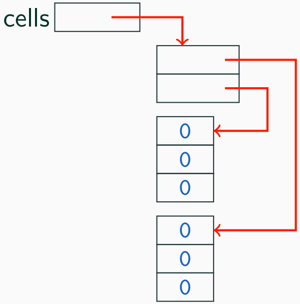
More arrays

- multi-dimensional arrays

More arrays

- multi-dimensional arrays

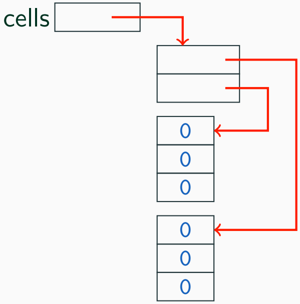
```
int[][] cells;
cells = new int[2][3];
```

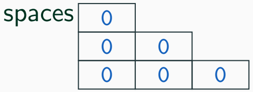


More arrays

- multi-dimensional arrays

```
int[][] cells;
cells = new int[2][3];

int[][] spaces;
spaces = new int[3][];
int j;
for (j = 0; j < spaces.length; j++)
    spaces[j] = new int[j + 1];
```





☺ **Thank you!** ☺