Activity 9: Strings

Internally, the library class `java.lang.String` stores an array of characters. It also provides a variety of useful methods for comparing, manipulating, and searching text in general.

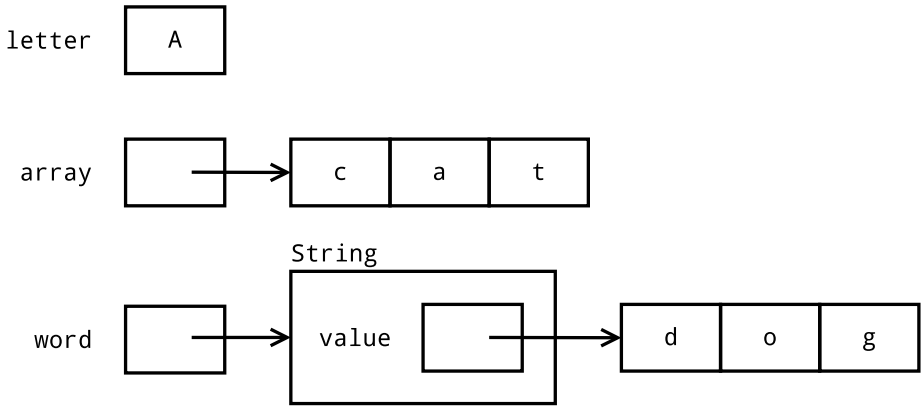
Model 1 Character Arrays

The primitive type `char` is used to store a single character, which can be a letter, a number, or a symbol. In contrast, the reference type `String` *encapsulates* an array of characters.

```
char letter;
letter = 'A';

char[] array;
array = new char[]
        {'c', 'a', 't'};

String word;
word = "dog";
```



Questions (15 min) Start time: _____

1. How is the syntax of character literals and string literals different?

2. What is the index of `'d'` in the string above? What is the index of `'g'`? In general, what is the index of the last character of a string?

3. Based on the diagram, what does it mean for a class to encapsulate data? How do you access data inside of a class?

4. Why can you use the `String` class in Java programs without having to import it first?

5. What is the value of a `char` variable? What is the value of an array variable? What is the value of a `String` variable?

6. Draw a memory diagram for the given code. Each variable should be a name next to a box containing its value.

```
String str;
str = "Hi!";

char let;
let = 'X';

int num;
num = -1;

double foo;
foo = num;

String hmm;
hmm = str;
```

7. Recall that the == operator compares the *value* of two variables. What does it mean for two `char` variables to be ==? What does it mean for two `String` variables to be ==?

8. How could you determine whether two character arrays have the same contents? In other words, how does the `Arrays.equals` method work internally?

Model 2 String Methods

Method	Returns	Description
`charAt(int)`	`char`	Returns the char value at the specified index of `this` string.
`indexOf(String)`	`int`	Returns the index within `this` string of the first occurrence of the specified substring.
`length()`	`int`	Returns the length of `this` string.
`substring(int, int)`	`String`	Returns a new string that is a substring of `this` string (from `beginIndex` to `endIndex - 1`).
`toUpperCase()`	`String`	Returns a copy of `this` string with all the characters converted to upper case.

Each method listed above is non-`static`. That is, they have an implicit parameter named `this` that is passed automatically. (Note: There are many other `String` methods not listed above.)

Questions (20 min) Start time: _____

9. If `str` contains the string `"hello world"`, then what is the return value of the following method calls?

a) `str.charAt(8)`

b) `str.indexOf("wo")`

c) `str.length()`

d) `str.substring(4, 7)`

e) `str.toUpperCase()`

10. Explain what precedes the `.` (dot) operator in the expressions above. What does it have to do with the keyword `this` in the model?

11. How many arguments does each method call in #9 have? (Hint: None of them have zero.)

a)

b)

c)

d)

e)

12. To compare strings, you must use either the `String.equals` or `String.compareTo` method. Predict the output of the following code.

```
String name1 = "Mark";
String name2 = "Ma" + "rk";
String name3 = "Mary";

// compare name1 and name2
if (name1 == name2) {
    System.out.println("name1 and name2 are identical");
} else {
    System.out.println("name1 and name2 are NOT identical");
}

// compare "Mark" and "Mark"
if (name1.equals(name2)) {
    System.out.println("name1 and name2 are equal");
} else {
    System.out.println("name1 and name2 are NOT equal");
}

// compare "Mark" and "Mary"
if (name1.equals(name3)) {
    System.out.println("name1 and name3 are equal");
} else {
    System.out.println("name1 and name3 are NOT equal");
}
```

13. What is the difference between *identical* and *equal* in the previous question?

14. Discuss the `stringMatch` problem on the next page. What three `String` methods will you need to solve it? (If you have time during the activity, complete the method.)

15. Discuss the `stringYak` problem on the next page. What two `String` methods will you need to solve it? (If you have time during the activity, complete the method.)

[CodingBat] Given two strings, return the number of positions where they contain the same substring of length two. So `"xxcaazz"` and `"xxbaaz"` yields 3, since the `"xx"`, `"aa"`, and `"az"` substrings appear in the same place in both strings.

```
public int stringMatch(String a, String b) {









}
```

[CodingBat] Suppose the string `"yak"` is unlucky. Given a string, return a version where all the `"yak"` are removed, but the `'a'` can be any character. The `"yak"` strings will not overlap.

```
stringYak("yakpak") → "pak"
stringYak("pakyak") → "pak"
stringYak("yak123ya") → "123ya"

public String stringYak(String str) {









}
```

Model 3 Common Mistakes

Program A	Program B

`String greeting = "hello world";`
`greeting.toUpperCase();`
`System.out.println(greeting);`
	`Scanner in = new Scanner(System.in);`
	`String line = in.nextLine();`
	`char letter = line.charAt(1);`
	`System.out.println(letter);`

Questions (10 min) Start time: _____

16. Write the output of each program in the space under the table above. What is the logic error you see when you run Program A?

17. In Program A, what is returned by the string method? What happens to the return value?

18. Describe two different ways you can fix the logic error in #17.

19. In general, what happens to `this` string when calling the methods in Model 2?

20. In what cases will Program B crash? What is the error message displayed?

21. Describe two different ways you can fix the run-time error in #20.