

CS106A REVIEW

QUICK OVERVIEW

- Strings
- Data Structures
 - Array
 - Grid
 - ArrayList
 - HashMap (with File Reading)
- Classes

STRING TIPS

Access characters of a String by index, starting with 0

Strings are immutable: you can't change them, you can only create new ones

If you are removing or adding characters, it's easier to go from the back than from the front (otherwise, you have to adjust your index)

Make sure you're familiar with String/Character methods (it makes your life a lot easier not to have to rewrite them)

GOOD CHARACTER METHODS TO KNOW

static boolean isDigit(char ch)

Determines if the specified character is a digit.

static boolean isLetter(char ch)

Determines if the specified character is a letter.

static boolean isLetterOrDigit(char ch)

Determines if the specified character is a letter or a digit.

static boolean isLowerCase(char ch)

Determines if the specified character is a lowercase letter.

static boolean isUpperCase(char ch)

Determines if the specified character is an uppercase letter.

static boolean isWhitespace(char ch)

Determines if the specified character is **whitespace** (spaces and tabs).

static char toLowerCase(char ch)

Converts **ch** to its lowercase equivalent, if any. If not, **ch** is returned unchanged.

static char toUpperCase(char ch)

Converts **ch** to its uppercase equivalent, if any. If not, **ch** is returned unchanged.

Slide Credit: Chris Piech

GOOD STRING METHODS TO KNOW

int length()

Returns the length of the string

char charAt(int index)

Returns the character at the specified index. Note: Strings indexed starting at 0.

String substring(int p1, int p2)

Returns the substring beginning at **p1** and extending up to but not including **p2**

String substring(int p1)

Returns substring beginning at **p1** and extending through end of string.

boolean equals(String s2)

Returns true if string **s2** is equal to the receiver string. This is case sensitive.

int compareTo(String s2)

Returns integer whose sign indicates how strings compare in lexicographic order

int indexOf(char ch) or int indexOf(String s)

Returns index of first occurrence of the character or the string, or -1 if not found

String toLowerCase() or String toUpperCase()

Returns a lowercase or uppercase version of the receiver string

And many more on the
javadocs for Strings!
Google “java String”

Slide Credit: Chris Piech

STRING EXAMPLE

Given a template string, template, and a pattern string to remove, pattern, return a new string with all occurrences of the pattern in the template removed:

```
removeAll("banana", "na") -> "ba"
```


```
removeAll("Hello World!", " ") -> "HelloWorld!"
```

STRING EXAMPLE

```
public String removeAll(String template, String pattern) {  
    String result = "";  
    for (int i = 0; i < template.length(); i++) {  
  
        result += template.charAt(i);  
  
    }  
    return result;  
}
```

STRING EXAMPLE

```
public String removeAll(String template, String pattern) {  
    String result = "";  
    for (int i = 0; i < template.length(); i++) {  
        String substring = "";
```


 `substring = template.substring(i, i + pattern.length());`

```
        } else {  
            result += template.charAt(i);  
        }  
    }  
    return result;  
}
```



STRING EXAMPLE

```
public String removeAll(String template, String pattern) {
    String result = "";
    for (int i = 0; i < template.length(); i++) {
        String substring = "";

        substring = template.substring(i, i + pattern.length());
    }
    if (substring.equals(pattern)) {
        i += pattern.length() - 1;
    } else {
        result += template.charAt(i);
    }
}
return result;
}
```



STRING EXAMPLE

```
public String removeAll(String template, String pattern) {  
    String result = "";  
    for (int i = 0; i < template.length(); i++) {  
        String substring = "";  
         if (i + pattern.length() <= template.length()) {  
            substring = template.substring(i, i + pattern.length());  
        }  
        if (substring.equals(pattern)) {  
            i += pattern.length() - 1;  
        } else {  
            result += template.charAt(i);  
        }  
    }  
    return result;  
}
```

DATA STRUCTURES!

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	arr[index]	arr[rowIndex][colIndex]	list.get(index) list.set(index, elem)	map.get(key) map.put(key, value)
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	arr[index]	arr[rowIndex][colIndex]	list.get(index) list.set(index, elem)	map.get(key) map.put(key, value)
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	arr[index]	arr[rowIndex][colIndex]	list.get(index) list.set(index, elem)	map.get(key) map.put(key, value)
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	<code>arr[index]</code>	<code>arr[rowIndex][colIndex]</code>	<code>list.get(index)</code> <code>list.set(index, elem)</code>	<code>map.get(key)</code> <code>map.put(key, value)</code>
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	arr[index]	arr[rowIndex][colIndex]	list.get(index) list.set(index, elem)	map.get(key) map.put(key, value)
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

A QUICK COMPARISON

	Array	2D Array	ArrayList	HashMap
Stores...	Fixed number of elements in a list	Grid/matrix of elements	Elements in a list	Key-Value Pairs
Type of element	Objects & Primitives	Objects & Primitives	Objects	Objects
Ordered?	Ordered	Ordered	Ordered	Unordered
Access Elements	arr[index]	arr[rowIndex][colIndex]	list.get(index) list.set(index, elem)	map.get(key) map.put(key, value)
Used when	Know the number of elements Histograms	Two “dimensions” of data (rows vs. columns)	Need ArrayList methods Unknown number of elements	Need to look up an object (value) via another object (key) Database
Example	Yahtzee Dice Frequencies, Yahtzee Dice	Images, Yahtzee score card	Hangman word list, list of names graphed in NameSurfer	NameSurferDatabase

HELPFUL ARRAY/2D ARRAY TIPS

Needs to be initialized with a size

Arrays and 2D arrays default initialize all values (e.g. an array of ints are all 0)

Get the size with: `arr.length`

Iterating over a 2D array: use a double for loop

Accessing 2D array elements: `arr[row][col]`

ARRAY/2D ARRAY EXAMPLE

Write a method where given a `GImage`, you return a **new** `GImage`, rotated 90 degrees to the left

Observation: where does the top left pixel end up? What are the new dimensions?



ARRAY/2D ARRAY EXAMPLE

```
public GImage rotateLeft(GImage image) {  
    int[][] original = image.getPixelArray();
```

```
}
```

ARRAY/2D ARRAY EXAMPLE

```
public GImage rotateLeft(GImage image) {  
    int[][] original = image.getPixelArray();  
    int numRows = original.length;  
    int numCols = original[0].length;
```

```
}
```

ARRAY/2D ARRAY EXAMPLE


```
public GImage rotateLeft(GImage image) {  
    int[][] original = image.getPixelArray();  
    int numRows = original.length;  
    int numCols = original[0].length;  
    → int[][] result = new int[numCols][numRows];
```

```
    → return new GImage(result);  
}
```


ARRAY/2D ARRAY EXAMPLE

```
public GImage rotateLeft(GImage image) {
    int[][] original = image.getPixelArray();
    int numRows = original.length;
    int numCols = original[0].length;
    int[][] result = new int[numCols][numRows];
    for (int row = 0; row < numRows; row++) {
        for (int col = 0; col < numCols; col++) {

        }
    }
    return new GImage(result);
}
```




ARRAY/2D ARRAY EXAMPLE

```
public GImage rotateLeft(GImage image) {
    int[][] original = image.getPixelArray();
    int numRows = original.length;
    int numCols = original[0].length;
    int[][] result = new int[numCols][numRows];
    for (int row = 0; row < numRows; row++) {
        for (int col = 0; col < numCols; col++) {
             int newRow = (numCols - 1) - col;
            int newCol = row;

        }
    }
    return new GImage(result);
}
```


ARRAY/2D ARRAY EXAMPLE

```
public GImage rotateLeft(GImage image) {
    int[][] original = image.getPixelArray();
    int numRows = original.length;
    int numCols = original[0].length;
    int[][] result = new int[numCols][numRows];
    for (int row = 0; row < numRows; row++) {
        for (int col = 0; col < numCols; col++) {
             result[newRow][newCol] = original[row][col];
        }
    }
    return new GImage(result);
}
```

HELPFUL ARRAYLIST METHODS

boolean add(<T> element)

Adds a new element to the end of the **ArrayList**; the return value is always **true**.

void add(int index, <T> element)

Inserts a new element into the **ArrayList** before the position specified by **index**.

<T> remove(int index)

Removes the element at the specified position and returns that value.

boolean remove(<T> element)

Removes the first instance of **element**, if it appears; returns **true** if a match is found.

void clear()

Removes all elements from the **ArrayList**.

int size()

Returns the number of elements in the **ArrayList**.

<T> get(int index)

Returns the object at the specified index.

<T> set(int index, <T> value)

Sets the element at the specified index to the new value and returns the old value.

int indexOf(<T> value)

Returns the index of the first occurrence of the specified value, or **-1** if it does not appear.

boolean contains(<T> value)

Returns **true** if the **ArrayList** contains the specified value.

boolean isEmpty()

Returns **true** if the **ArrayList** contains no elements.

Thanks for the slide,
Chris Piech

ARRAYLIST EXAMPLE

- Given a list and a String to remove, remove all instances of that String from the list

Example:

```
removeString(["a", "b", "a", "c"], a) -> ["b", "c"]
```

Tip: iterating from the end of the list makes it easier to avoid errors with indices

ARRAYLIST EXAMPLE

```
public void removeString(ArrayList<String> list, String toRemove) {  
    for (int i = list.size() - 1; i >= 0; i--) {  
        if (list.get(i).equals(toRemove)) {  
            list.remove(i);  
        }  
    }  
}
```

HASHMAP EXAMPLE

Section leaders can get prizes for helping the most number of students in LaIR. Write a method that takes in a filename corresponding to a file of help requests, and outputs the section leader's name who helped the most people. (In the event of a tie, your program can output any of the winning names)

The file should look like below, where the section leader's name is first, followed by a space and then the name of the student.

Ashley csStudent1

Chris csStudent2

Nick csStudent3

BRIEF DETOUR: FILE READING COMPONENTS

Always use a try/catch

Open the file in a
BufferedReader

Get a line

while line is not null:

 Do something with
 the line

 Read the next line

Close the
BufferedReader

```
public void readFile(String filename) {  
    try {  
        BufferedReader br =  
            new BufferedReader(new FileReader(filename));  
        String line = br.readLine();  
        while (line != null) {  
            //do something with line  
            line = br.readLine();  
        }  
        br.close();  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}
```

HASHMAP EXAMPLE - READING THE FILE

```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));


        String line = br.readLine();
        while (line != null) {

            line = br.readLine();
        }
        br.close();

    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```


HASHMAP EXAMPLE - READING THE FILE

```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));

        String line = br.readLine();
        while (line != null) {
            
            String[] parts = line.split(" ");
            String slName = parts[0];


            line = br.readLine();
        }
        br.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```


HASHMAP EXAMPLE - READING THE FILE

```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        HashMap<String, Integer> helpRequests = new HashMap<String, Integer>();
        String line = br.readLine();
        while (line != null) {
            String[] parts = line.split(" ");
            String slName = parts[0];

            line = br.readLine();
        }
        br.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```


HASHMAP EXAMPLE - READING THE FILE

```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        HashMap<String, Integer> helpRequests = new HashMap<String, Integer>();
        String line = br.readLine();
        while (line != null) {
            String[] parts = line.split(" ");
            String slName = parts[0];
            if (helpRequests.containsKey(slName)) {
                 helpRequests.put(slName, helpRequests.get(slName) + 1);
            }
            line = br.readLine();
        }
        br.close();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```

HASHMAP EXAMPLE - READING THE FILE


```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        HashMap<String, Integer> helpRequests = new HashMap<String, Integer>();
        String line = br.readLine();
        while (line != null) {
            String[] parts = line.split(" ");
            String slName = parts[0];
            if (helpRequests.containsKey(slName)) {
                helpRequests.put(slName, helpRequests.get(slName) + 1);
            } else {
                helpRequests.put(slName, 1);
            }
            line = br.readLine();
        }
        br.close();

    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```



HASHMAP EXAMPLE - READING THE FILE

```
public String findBestLairHelper(String filename) {
    try {
        BufferedReader br = new BufferedReader(new FileReader(filename));
        HashMap<String, Integer> helpRequests = new HashMap<String, Integer>();
        String line = br.readLine();
        while (line != null) {
            String[] parts = line.split(" ");
            String slName = parts[0];
            if (helpRequests.containsKey(slName)) {
                helpRequests.put(slName, helpRequests.get(slName) + 1);
            } else {
                helpRequests.put(slName, 1);
            }
            line = br.readLine();
        }
        br.close();
        return getBestLairHelper(helpRequests);
    } catch (IOException e) {
        e.printStackTrace();
    }
    return "";
}
```




HASHMAP EXAMPLE - FINDING THE MOST REQUESTS

```
public String getBestLairHelper(HashMap<String, Integer> helpRequests) {  
    String bestLairHelper = "";  
    int highestCount = 0;  
  
    return bestLairHelper;  
}
```


HASHMAP EXAMPLE - FINDING THE MOST REQUESTS

```
public String getBestLairHelper(HashMap<String, Integer> helpRequests) {  
    String bestLairHelper = "";  
    int highestCount = 0;  
    for (String sl : helpRequests.keySet()) {  
  
        return bestLairHelper;  
    }  
}
```

HASHMAP EXAMPLE - FINDING THE MOST REQUESTS

```
public String getBestLairHelper(HashMap<String, Integer> helpRequests) {  
    String bestLairHelper = "";  
    int highestCount = 0;  
    for (String sl : helpRequests.keySet()) {  
         int currCount = helpRequests.get(sl);  
  
        return bestLairHelper;  
    }  
}
```

HASHMAP EXAMPLE - FINDING THE MOST REQUESTS

```
public String getBestLairHelper(HashMap<String, Integer> helpRequests) {  
    String bestLairHelper = "";  
    int highestCount = 0;  
    for (String sl : helpRequests.keySet()) {  
        int currCount = helpRequests.get(sl);  
         if (currCount > highestCount) {  
            highestCount = currCount;  
            bestLairHelper = sl;  
        }  
    }  
    return bestLairHelper;  
}
```


CLASSES!

PARTS OF A CLASS

Instance Variables

Constructor

Getter/Setter Methods (usually directly return or change instance variables)

Other methods (might indirectly change instance variables; what the class does)

HELPFUL HINTS FOR CLASSES

- If you are given parameters in a constructor, make sure you save them somehow
 - E.g. if you're given a file, you might save the contents in a HashMap/ArrayList
 - Might directly save some parameters as instance variables using the `this` keyword
- Look at the required getter/setter methods - make sure that each one corresponds to an instance variable
 - E.g. in `NameSurferEntry`, `getRank(int decade)` returned an element in a data structure that was stored in an instance variable
- Look at the functionality for other methods - what information do you need to do those correctly?

CLASSES EXAMPLE

Write a class that models a Dog. Our dog knows tricks! He can bark how old he is (one bark per year) when asked to speak, he can play, which turns his color from the original color to brown (because of the mud), he can shower, which restores his color, and he has a name, which he was born with, but which might be changed by a new owner. Every third time he showers, he ages one year.

```
Dog(String name, Color color)
```

```
String speak() //returns a string like "bark bark bark"
```

```
Color shower() //returns the original color
```

```
String getName(); String setName(String); Color getColor()
```

GOOD INSTANCE VARIABLES

`Dog(String name, Color color)`

`String speak() //returns a string like "bark bark bark"`

`Color shower() //returns the original color`

`String getName(); String setName(String); Color getColor()`

Name

Number of times the dog has showered (to get the age)

Current color

Original color

CLASSES EXAMPLE

```
public class Dog {  
    private String name;  
    private int numShowers;  
    private Color originalColor;  
    private Color currentColor;
```

CLASSES EXAMPLE

```
public class Dog {  
    private String name;  
    private int numShowers;  
    private Color originalColor;  
    private Color currentColor;  
  
    public Dog(String name, Color color) {  
        this.name = name;  
        originalColor = color;  
        currentColor = color;  
        numShowers = 0;  
    }  
}
```

CLASSES EXAMPLE

```
public class Dog {  
    private String name;  
    private int numShowers;  
    private Color originalColor;  
    private Color currentColor;  
  
    public Dog(String name, Color color) {  
        this.name = name;  
        originalColor = color;  
        currentColor = color;  
        numShowers = 0;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
  
    public Color getColor() {  
        return currentColor;  
    }  
}
```


CLASSES EXAMPLE CONTINUED

```
public Color shower() {  
    currentColor = originalColor;  
    numShowers++;  
    return currentColor;  
}
```

```
public String speak() {  
    String result = "";  
    for (int i = 0; i < numShowers / 3; i++) {  
        result += "bark ";  
    }  
    return result;  
}
```

```
}
```