Intro to Java

MadLibs

Getting Started

Download Java:

https://java.com/download

Download an IDE:

https://www.bluej.org/

Hello, world!

Every Java file has one public class named after the file.

Main is the first function.

We can print text to the screen.

```
public class Madlibs {
    public static void main(String args[]) {
        System.out.println("Hello, world!");
    }
}
```

Functions

Have inputs and outputs, known as parameters and return values.

returnVariableName = functionName(parameterName);

objectName.functionName();

Scanner Class

Use a Scanner object to read text into a program.

Calling the .next() function on the Scanner gets the next word.

```
import java.util.Scanner;
```

```
public class Madlibs {
    public static void main(String args[]) {
        Scanner scanny = new Scanner(System.in);
        System.out.println(scanny.next());
    }
}
```

Imports

Sometimes your programs will use classes from extra packages, so you must specify where they come from.

Changing `import java.util.Scanner;`

to `import java.util.*;`

gives access to everything in the util package.

Variables

Used to refer to objects and values.

Must first be declared by giving the type, then just the name is needed.

Type variableName = ;

```
int myNumber = 3;

myNumber = myNumber + 2;
```

Loops

Used to do the same tasks many times.

The while loop runs while the loop condition is true - until there are no more words to scan.

```
import java.util.Scanner;

public class Madlibs {
    public static void main(String args[]) {
        Scanner scanny = new Scanner(System.in);
        while (scanny.hasNext()) {
            System.out.println(scanny.next());
        }
    }
}
```

String Class

"Hello, world." Strings are used to represent words.

They can be added together. \n is the new line character.

Let's make a String of our story.

```
public static void main(String args[]) {
    Scanner scanny = new Scanner(System.in);

    //A = adjective N = noun V = verb AV = adverb C = character
    String myStory = (
            "Once upon a time, there was a A1 A2 N1 named C1 who\n"
            + "always thought about N2 even while V1 or V2 .\n"
            + "But AV1 , C1 was changed by a A3 event.\n" +
            "A N2 V3 AV2 so C1 could never\n" +
            "think about N1 in the same way again.\n" +
            "The end.\n");
        System.out.print(myStory);
```

Comments

Comments are notes for people to read that are ignored by the program. They help with organization and understanding what a program does.

```
//This is a one line comment.

/* This is a

 *  multiline comment.

 */
```

HashMap Class

Stores a pair of objects - a key and an associated value.

```
HashMap<String,String> myBlanks = new HashMap<>();
```

When a new HashMap is declared, you must specify the type of all the keys and their values, which can both be any object.

Put Keys and Values in a HashMap

```
"Once upon a time, there was a A1 A2 N1 named C1 who\n"
+ "always thought about N2 even while V1 or V2 .\n"
+ "But AV1 , C1 was changed by a A3 event.\n" +
"A N2 V3 AV2 so C1 could never\n" +
"think about N1 in the same way again.\n" +
"The end.\n");
```

```
myBlanks.put("A1", "Adjective: ");
myBlanks.put("A2", "Adjective: ");
myBlanks.put("N1", "Noun: ");
myBlanks.put("C1", "Character name: ");
myBlanks.put("N2", "Noun: ");
myBlanks.put("V1", "ING verb: ");
myBlanks.put("V2", "ING verb: ");
myBlanks.put("AV1", "Adverb: ");
myBlanks.put("A3", "Adjective: ");
myBlanks.put("N2", "Noun: ");
myBlanks.put("V3", "Past tense verb: ");
myBlanks.put("AV2", "Adverb: ");
```

For Loop

Like the while loop, but iterates over some values.

First type: do a task for every value in a set.

For every blank (key) in the story, print the prompt (value) then replace the prompt with the user input.

```
for (String key : myBlanks.keySet()) {
    System.out.print(myBlanks.get(key));
    myBlanks.replace(key, scanny.next());
}
```

Arrays

An indexed list of objects. All objects must be the same type.

Calling the split function on the story returns an array of single words.

```
String[] splitStory = myStory.split(" ");
```

Other arrays: int[] myNumbers = {1, 5, 4};

String[] emptyArraySizeThree = new String[3];

For Loops Part 2

You can iterate over the objects in an array using an integer to track the index of each object.

3 parts to loop condition: initial condition, run while, do after each loop.

```
for (int i = 0; i < splitStory.length; i++) {
```

This loopstarts with i = 0, runs while it's less than the length of the array, and increases i by one each time through.

If Statements

The code inside the block will only execute if the condition is true. If the word in the array is in the HashMap (aka it's a blank), replace it with the value for that key (the user input).

```
for (int i = 0; i < splitStory.length; i++) {
    if (myBlanks.containsKey(splitStory[i])) {
        splitStory[i] = myBlanks.get(splitStory[i]);
    }
}
```

```
if (a < b) {

} else if (b > a) {

} else {

}
```

Put the story back together!

Just like we made an array from a String, we can make a String from an array using the join function. It takes two parameters - which characters to join with and the array.

```
myStory = String.join(" ", splitStory);
```

The story is finished!

Print and close up shop!

```
        System.out.println(myStory);
    }
}
```

Extra Resources

https://docs.oracle.com/javase/7/docs/api/

https://www.codecademy.com/learn/learn-java

The End