Small Text Programs

— Elements of a simple Java program
— Prompting the user
— Getting user input
— Being robust

Elements of a simple Java program

```
import java.util.NameOfClassUsed;

public class ProgramName {

    public static void main(String[] args) {

        // Program starts here

    }

}
```

Prompting the user

```
public static void main(String[] args) {

    System.out.print("What is you name? ");   // Cursor stays on line, OR
    System.out.println("What is you name? "); // Cursor advances

}
```

Getting user input

```
public static void main(String[] args) {

    Scanner in = new Scanner(System.in);
    System.out.print("What is you name? ");
    String name = in.next();                    // Grabs next token, OR
    String name = in.nextLine();                // Grabs to next newline
    System.out.println("Hello " + name + "!");

}
```

Example: Linux text filters

```
% cat test.txt
grape
apple
banana
% cat test.txt | sort
apple
banana
grape
```

Example: Linux text filters

```
public static void main(String[] args) {
    String[] lines = new String[8];  // Bad. Can only handle 8.
    int numLines = 0;
    Scanner in = new Scanner(System.in);
    while (in.hasNextLine()) {
        lines[numLines] = in.nextLine();
        numLines += 1;
    }
    Arrays.sort(lines,0,numLines);
    for (int i=0; i<numLines; i++) {
        System.out.println(lines[i]);
    }
}
```

Being robust

Program should act sensibly no matter what is input.

Previous would throw exception if more than 8 lines.

Robust: Increase size of array if more lines.

```
if (numLines == lines.length) {
    String[] tmp = new String[2*lines.length]
    for (int i=0; i<lines.length; i++) {
        tmp[i] = lines[i];
    }
    lines = tmp;
}
```

Example: Robust Linux text filters

```
public static void main(String[] args) {
    String[] lines = new String[8];  // Bad. Can only handle 8.
    int numLines = 0;
    Scanner in = new Scanner(System.in);
    while (in.hasNextLine()) {
        if (numLines == lines.length) {
            String[] tmp = new String[2*lines.length]
            for (int i=0; i<lines.length; i++) {
                tmp[i] = lines[i];
            }
            lines = tmp;
        }
        lines[numLines] = in.nextLine();
        numLines += 1;
    }
    Arrays.sort(lines,0,numLines);    // Sorts indices 0..numLines-1
    for (int i=0; i<numLines; i++) {
        System.out.println(lines[i]);
    }
}
```