CSE 142 Final
Cheat Sheet

```
for (initialization; test; update) {
    statement(s);
    ...
}
```

```
while (condition) {
    statement(s);
}
```

```
if (test) {
    statement(s);
} else if (test) {
    statement(s);
} else {
    statement(s);
}
```

```
public static type name(parameters) {
    statement(s);
    ...
    return expression;
}
```

```
type name = value;                    // variable declaration and initialization
Type objectName = new Type(parameters);// object construction
```

Math Method	Description
`Math.abs(`*value*`)`	absolute value
`Math.min(`*v1, v2*`)`	smaller of two values
`Math.max(`*v1, v2*`)`	larger of two values
`Math.round(`*value*`)`	nearest whole number
`Math.sqrt(`*value*`)`	square root
`Math.pow(`*b, e*`)`	base to the exponent power

Random Method	Description
`nextInt(`*max*`)`	random integer from 0 to *max*-1

String Method	Description
`contains(`***str***`)`	true if this string contains the other's characters inside it
`endsWith(`***str***`)`, `startsWith(`***str***`)`	true if this string starts/ends with the other's characters
`equals(`***str***`)`	true if this string is the same as *str*
`equalsIgnoreCase(`***str***`)`	true if this string is the same as *str*, ignoring capitalization
`indexOf(`***str***`)`	index in this string where given string begins (-1 if not found)
`length()`	number of characters in this string
`replace(`***str1***`, `***str2***`)`	replace all occurrences in this string of *str1* with *str2*
`substring(`***i***`, `***j***`)`	characters in this string from index *i* (inclusive) to *j* (exclusive)
`toLowerCase()`, `toUpperCase()`	a new string with all lowercase or uppercase letters
`charAt(`***i***`)`	returns char at index *i*

Scanner Method	Description
`nextInt()`	reads/returns input token as `int`
`next()`	reads/returns input token as `String`
`nextDouble()`	reads/returns input token as `double`
`nextLine()`	reads/returns line as `String`
`hasNextInt()`	returns `true` if there is a next token and it can be read as an `int`
`hasNext()`	returns `true` if there is a next token to read
`hasNextDouble()`	returns `true` if there is a next token and it can be read as a `double`
`hasNextLine()`	returns `true` if there is a next line to read

Declaring and using Arrays

```
type[] name = new type[length];
type[] name = {VAL1, VAL2, VAL3, ...};
name[index] = value;

name.length   // number of elements in the array
```

Arrays Class Method	Description
`Arrays.fill(`***array***`, `***value***`)`	sets every element in the array to the given value
`Arrays.sort(`***array***`)`	arranges the elements in the array into ascending order
`Arrays.toString(`***array***`)`	returns a `String` for the array, such as: `"[10, 30, 17]"`
`Arrays.copyOf(`***array***`, `***length***`)`	returns a new copy of the given array of the given length
`Arrays.copyOfRange(`***array***`, `***i, j***`)`	returns a new copy of the given array from indices i (incl.) to j (excl.)
`Arrays.equals(`***array1***`, `***array2***`)`	returns `true` if the two arrays have the same elements

Classes

Field *(data inside each object)*

```
private type name;
```

Method *(behavior inside each object)*

```
public type name(parameters) {
    statements;
}
```

Constructor *(code to initialize new objects)*

```
public className(parameters) {
    statements;
}
```

`toString` method *(called when an object is printed)*

```
public String toString() {
    code that produces/returns a String;
}
```

Inheritance

```
public class name extends superclass {

}
```

Critter classes

```
public class name extends Critter {
    fields

    constructor

    public boolean eat() {
        statement(s) that return true (eat) or false (don't eat);
    }

    public Attack fight(String opponent) {
        returns either Attack.ROAR, Attack.POUNCE, or Attack.SCRATCH;
    }

    public Color getColor() {
        statement(s) that return a Color;
    }

    public Direction getMove() {
        statement(s) that return either Direction.NORTH, Direction.SOUTH,
          Direction.EAST, Direction.WEST, or Direction.CENTER;
    }

    public String toString() {
        statement(s) that return a String;
    }
}
```