AP Computer Science A
Class Notes 5

Converting Between Strings and Character Arrays

The point of converting from a String to a character array is to allow us to have character-by-character control over the contents. Since a String is immutable, and cannot be changed, that limits how we are able to use them in a game such as WordGuess, where that control of individual characters is paramount.

However, we do need to be able to convert our character array back into a string so that we can display it.

toCharArray()

This function, which exists in the behavior of the String data type, takes the contents of a String and *returns* to us a character array which contains all of the individual characters in the String in array format, of the same length as the String.

```
String name = "Mark";
char[] letters = name.toCharArray();
for( char c : letters )
{
    System.out.print( c );
}
```

charAt()

This function, also part of the String data type, allows us to get a specific character from a String at a specific index position.

```
String word = "Gorilla";
char character = word.charAt( 3 );
System.out.print( character );
output: i
```

Converting a character array back into a string requires the following:

```
char[] letters = { 'f', 'i', 's', 'h' };

// elegant solution
String word = String.valueOf( letters );

// brute force solution
String word = "";
for( int i = 0; i < letters.length; i++ )
{
    word += letters[ i ];
}
```

The Empty String

An empty string is a declared and instantiated piece of data.

```
String blank = "";
```

It is not the same as just declaring a String and leaving it null.

```
String nothing;
String nothing = null;
```

The marker *null* indicates that no data exists at this memory location. When you declare a variable, its value is null until it is instantiated. This means that integers must be set to some value, even if it is 0, and Strings have to be set to at least the empty string "".

When you declare an Empty String, **DO NOT PUT A SPACE IN IT.**

WordGuess Revised

Take your WordGuess program, and modify it to do the following:

1. Limit the number of guesses that a player is able to make before the game stops. If the player makes too many wrong guesses, the game ends

and notifies the player what the secret word is. *Hint: you might want to set the max number of guesses as a multiplier of the number of letters in the secret word.*

2. Create an option so that the user can try and guess the entire word at once. If they get it wrong, it simply counts as one guess.

Typecasting

Typecasting is the act of *casting* one kind of data as another. This is usually used in two ways:

1. When multiplying integers by floating point values, converting the integer to floating point so that we have specific decimal results.

```
int value = 6;
double num = 2.521351237 * (double) value;
```

2. When multiplying integers by floating point values, setting the final value (casting) it as an integer, such that it truncates the decimal portion of the number.

```
int value = 6;
int num = (int) (2.521351237 * value);
```

Procedural Programming

Up to this point, we have engaged in a programming paradigm (or way of doing things) called Structured Programming. Structured programming works in a sequential, line-by-line process. This is demonstrated by our writing of programs completely contained within a single main() function. While this makes it easy to put all of our code in a single place, as you noticed with WordGuess, when we generate enough code, it becomes difficult to keep track of all of it.

We are going to engage in the next programming paradigm (or way of doing things) called *Procedural Programming*, and it doesn't matter if you call it a procedure, a function, or a *method*, which is Java's preference. A method is a way of *encapsulating* a block of code within a unique name, that can be called by other parts of the program. This is part of the *divide and conquer strategy* of programming, which seeks to take large, complex, tasks, and divide them into smaller, easier-to-solve subtasks.

Methods

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

The first word in a method declaration is the **access level**. It can either be **public** or **private**. Methods that are public can be called and used by anyone, including callers outside of the class. Methods that are private can only be seen within the class that the method is written.

The second word is the **static** or non-static (the word is not present) reference. This determines whether a method can be called directly from the class (like the Arrays.equal() method you used in the last lab) or whether an object has to be created from the class in order to be used, like Scanner and Random.

The third word is the **return type.** This indicates the data type of the information that the method can send back to the caller. If you write a method that calculates something, it is usually proper to hand the caller back the solution that they were looking for. If a method does not return any data, we use the type **void**. Otherwise, we choose any kind of primitive or abstract data type as the return type, like int, double, String, etc.

The fourth word is the **name**. Method names should be *verbs*, because they describe the **behavior** that a program has. Consequently, your variable names should always be *nouns*, because they describe the **state** of your program.

Within the parenthesis that follow the name are the **parameters**. This is where you can pass to the method any number of declared variables to be used in the method's calculations.

Compound Boolean Expression

```
&&   and
||   or
```

Method Examples

See code from today.

Quadratic Formula

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

```
// useful functions

// absolute value
Math.abs()

// square root
Math.sqrt()
```

Reading and Homework

Please see eLearning for a separate post with method practice for this week, as well as new textbook chapter that covers methods in great detail.

Practice programming for 10-15 minutes a day, maximum.

Reminder:

How do you ask Mr. Wie questions?

1. Use the Remind app.
2. Text me at (949) 542-6858.
3. Slowest way: email me...okay, actually, don't!