Lecture 6.4
The char and String Types

char and String

char is for storing single characters

• primitive type
• constants: a printable character enclosed in single quotes
  ( ‘Y’ ‘y’ ‘#’ ‘7’ )
• char widens to int, long, float or double
• operators: ++ and --
  caution: (char)7 is not the same as ‘7’

String is for storing a sequence of characters

• built-in Java class (java.lang.String)
• constants: zero or more characters enclosed in double quotes
  ( “Hi mom.” “$###!!” )
• a String object is immutable
• operators: +
  caution: new not used with constant notation.
Examples

**variable declarations**

```java
private char initial, digit;
private String name, message, str;
```

- `initial = 'D';`
- `digit = '0';`
- `digit++;`
- `initial = (char)(initial + 3);`
- `System.out.println(digit);`
- `System.out.println(initial);`
- `name = "Riley";`
- `str = "";`
- `message = name + " assigns programs.";`
- `System.out.println( initial + ". " + message );`
- `System.out.println("digit value is " + digit );`

**Class Diagram**

```
String

«constructor»
+ String( String )
...

«query»
+ char charAt( int )
+ int length()
+ String substring(int)
+ String substring(int, int)
...

«translate»
+ String toLowerCase( )
+ String toUpperCase( )
...

«infix operator»
+ ...
```
## Examples

### variable declarations

```java
private String cheese, theory;

cheese = "limburger";
System.out.println( cheese.substring(3) );

cheese = cheese.substring(6);
System.out.println( cheese );

cheese = "gouda".substring(2);
System.out.println( cheese );

theory = "quantum mechanics";
System.out.println( theory.substring(4,10) );
System.out.println( theory );

cheese = "gouda";
theory = "hadrons";
System.out.println( cheese.substring(3,5) + " " + theory.charAt(0) + cheese.substring(4) + 't' );
```

## Differences Between Types

Strings and primitive types are largely incompatible...

- You can add, subtract, multiply or divide most primitives, but not Strings
  ```java
  String product = "75.2" * "3";
  ```

- You can extract individual characters from Strings, but not primitives
  ```java
  char decimalPoint = ("38.25").charAt(2);
  char nonSymbol = (38.25).charAt(2);
  ```

- Because they are a reference type, Strings are instantiated via `new`, but not primitives.
  ```java
  char letter = new char('D');
  ```

- Even if a String stores a single character it is incompatible with char.
  ```java
  char questionMark = "?";
  String asterisk = "*";
  ```
Exercises

1. Write a method, called toPigLatin, that returns its String parameter translated into Pig-Latin, assuming the parameter consists of a single non-blank word.

Examples:

toPigLatin( "cow" ) returns "ow-cay"
toPigLatin( "penguin" ) returns "enguin-pay"

2. Complete the method below so that it concatenates s1 less its last character with s2 less its first character. (You may assume s1 & s2 are one or more characters long.)

Examples:

mergedWithoutEnds( "cow", "pig" ) returns "coig"
mergedWithoutEnds( "chip", "joke" ) returns "chioke"

private String mergedWithoutEnds(String s1, String s2) {
}

Type Conversion Expressions

Any primitive => String

concatenate to the empty string
String realValue = 41.75 + "";
String logicalNegative = false + "";

String => char

use charAt
char initial = ("7").charAt(0);

String => Any primitive excepting char

1) Instantiate Scanner passing the String to be converted
2) Call nextType() upon this Scanner object.

String closeToPi = "3.14159";
Scanner converter = new Scanner(closeToPi);
double approximationOfPi = converter.nextDouble();
which can be abbreviated...
double approximationOfPi = (new Scanner(closeToPi)).nextDouble();