Strings

CSE 1310 – Introduction to Computers and Programming
Vassilis Athitsos & Chris Conly
University of Texas at Arlington

The String Type

- In the same way that **int** and **double** are designed to store numerical values, the **String** type is designed to store text.
- Text for strings must be enclosed in double quotes.
- Examples:

```
String name = "George";

String phone_number = "310-123-987";
```

A Simple Program Using Strings

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Hi, my name is Java.\n");
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    System.out.printf("Hello %s %s, nice to meet you!\n",
            first_name, last_name);
  }
}
```

```
Example Output:

???
```

A Simple Program Using Strings

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Hi, my name is Java.\n");
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    System.out.printf("Hello %s %s, nice to meet you!\n",
            first_name, last_name);
  }
}
```

```
Example Output:

Hi, my name is Java.
What is your first name? Mary
What is your last name? Smith
Hello Mary Smith, nice to meet you!
```

String Input from the User

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Hi, my name is Java.\n");
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    System.out.printf("Hello %s %s, nice to meet you!\n",
            first_name, last_name);
  }
}
```

- As you see above, to read a string from user input, you use the Scanner.next() method.
- Note: although the code calls in.next(), the name of the method is Scanner.next(), because **in** is just an arbitrary variable name.

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.next();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

???
```

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.next();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary
```

- What is wrong with the output here?

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.next();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary
```

- What is wrong with the output here?
- The user types that the name is “Mary Smith”.
- However, the program does NOT print “Hello Mary Smith”.
  - It prints “Hello Mary”.

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.next();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary
```

- The Scanner **next** method returns a single word that the user entered.
- It stops at the first space character. The rest of what the user entered remains in the keyboard buffer and can still be accessed.

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String firstName = in.next();
    String lastName = in.next();
    System.out.printf("Hello %s %s\n", firstName, lastName);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary Smith
```

- The Scanner **next** method returns a single word that the user entered.
- It stops at the first space character. The rest of what the user entered remains in the keyboard buffer and can still be accessed.

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.next();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary
```

- To get a single word from the user, use the Scanner **next** method.
- To get an entire line of text from the user, use the Scanner **nextLine** method.

next and **nextLine**

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your name? ");
    String name = in.nextLine();
    System.out.printf("Hello %s\n", name);
  }
}
```

```
Example Output:

What is your name? Mary Smith
Hello Mary Smith
```

- To get a single word from the user, use the Scanner **next** method.
- To get an entire line of text from the user, use the Scanner **nextLine** method.
- Here we changed in.next to in.nextLine, and now it works!

Length of a String

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Hi, my name is Java.\n");
    System.out.printf("What is your name? ");
    String name = in.next();
    int length = name.length();
    System.out.printf("Your name has %d letters!\n", length);
  }
}
```

```
Example Output:

Hi, my name is Java.
What is your name? Vassilis
???
```

- To obtain the length of a string, we use the String.length() method.

Length of a String

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("Hi, my name is Java.\n");
    System.out.printf("What is your name? ");
    String name = in.next();
    int length = name.length();
    System.out.printf("Your name has %d letters!\n", length);
  }
}
```

```
Example Output:

Hi, my name is Java.
What is your name? Vassilis
Your name has 8 letters!
```

- To obtain the length of a string, we use the String.length() method.

String Concatenation Using +

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    String name = first_name + last_name;
    System.out.printf("Hello %s!\n", name);
  }
}
```

```
Example Output:

What is your first name? Mary
What is your last name? Smith
???
```

- string1 + string2 returns the result of putting those strings together. This is what we call **"string concatenation"**.

String Concatenation Using +

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    String name = first_name + last_name;
    System.out.printf("Hello %s!\n", name);
  }
}
```

```
Example Output:

What is your first name? Mary
What is your last name? Smith
Hello MarySmith!
```

- string1 + string2 returns the result of putting those strings together. This is what we call **"string concatenation"**.

String Concatenation Using +

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    String name = first_name + " " + last_name;
    System.out.printf("Hello %s!\n", name);
  }
}
```

```
Example Output:

What is your first name? Mary
What is your last name? Smith
Hello Mary Smith!
```

- When you concatenate strings, make sure that you put spaces where they are needed.

Bonus Material: String.format()

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    System.out.printf("What is your last name? ");
    String last_name = in.next();
    String name = String.format("%s %s", first_name, last_name);
    System.out.printf("Hello %s!\n", name);
  }
}
```

```
Example Output:

What is your first name? Mary
What is your last name? Smith
Hello Mary Smith!
```

- String.format() gives you a printf()-like syntax to create a string.
- Similar to sprintf() in C.

String Concatenation Using +=

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    String message = "Hello ";
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    message += first_name;
    System.out.printf("%s!\n", message);
  }
}
```

```
Example Output:

What is your first name? Mary
???
```

- The following two lines do the EXACT SAME THING:

variable_name += *value*;

variable_name = *variable_name* + *value*;

String Concatenation Using +=

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    String message = "Hello ";
    System.out.printf("What is your first name? ");
    String first_name = in.next();
    message += first_name;
    System.out.printf("%s!\n", message);
  }
}
```

```
Example Output:

What is your first name? Mary
Hello Mary!
```

- The following two lines do the EXACT SAME THING:

variable_name += *value*;

variable_name = *variable_name* + *value*;

Escape Sequences

- If you want to put a " character in a string: use \"
- If you want to put a \ character in a string: use \\
- If you want to put a newline character in a string: use \n

```
public class example1 {
  public static void main(String[] args) {
    String a = "He said \"Hello\"";
    String b = "C:\\users\\jane\\note.txt";
    String c = "*\n**\n***";
    System.out.println(a);
    System.out.println(b);
    System.out.println(c);
  }
}
```

```
Output:

???
```

Escape Sequences

- If you want to put a " character in a string: use \"
- If you want to put a \ character in a string: use \\
- If you want to put a newline character in a string: use \n

```
public class example1 {
  public static void main(String[] args) {
    String a = "He said \"Hello\"";
    String b = "C:\\users\\jane\\note.txt";
    String c = "*\n**\n***";
    System.out.println(a);
    System.out.println(b);
    System.out.println(c);
  }
}
```

```
Output:

He said "Hello"
C:\users\jane\note.txt
*
**
***
```

Characters and Substrings

- The position of string characters are numbered starting from 0.
- To get the character at position *p*: use **charAt(p)**;
- To get the substring from position *s* up to and **not including** position *t*, use **substring(s, t)**
- If you leave off the *t*, and use **substring(s)** instead, it takes the substring from position **s** to the end of the string.

```
public class example1 {
  public static void main(String[] args) {
    String a = "Hello, world!";
    char first = a.charAt(0);
    char fifth = a.charAt(4);
    String sub = a.substring(2, 9);
    System.out.println(first);
    System.out.println(fifth);
    System.out.println(sub);
  }
}
```

```
Output:

???
```

Characters and Substrings

- The position of string characters are numbered starting from 0.
- To get the character at position *p*: use **charAt(p)**;
- To get the substring from position *s* up to and **not including** position *t*, use **substring(s, t)**
- If you leave off the **t** and use **substring(s)** instead, it takes the substring from position **s** to the end of the string.

```
public class example1 {
  public static void main(String[] args) {
    String a = "Hello, world!";
    char first = a.charAt(0);
    char fifth = a.charAt(4);
    String sub = a.substring(2, 9);
    System.out.println(first);
    System.out.println(fifth);
    System.out.println(sub);
  }
}
```

```
Output:

H
o
llo, wo
```

Printing Characters with printf

- To print a value of type **char** with System.out.printf, you should use %c.
  - %s will also work, but it is really meant to be used for strings. Do not use %s with characters.

Example: Printing Name Initial

- Write a program that:
  - Asks the user:
    What is your name?
  - Gets the name from user input.
  - Prints:
    Your initial is *X*
    - where *X* is the first letter of the name that the user typed.

Example: Printing Name Initial

```
import java.util.Scanner;

public class example1 {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);

    System.out.printf("What is your name? ");
    String name = in.next();
    char initial = name.charAt(0);
    System.out.printf("Your initial is %s\n", initial);
  }
}
```

```
Example Output:

What is your name? Mary
Your initial is M
```

```
Example Output:

What is your name? John
Your initial is J
```

Converting Numbers to Strings

- To convert an integer to a string, use:
  - **Integer.toString** method.
  - **String.valueOf** method.
  - Concatenation.

```
public class example1 {
  public static void main(String[] args)
  {
    int a = 25;
    String s1 = Integer.toString(a);
    String s2 = String.valueOf(a);
    String s3 = "" + a;
    System.out.printf("s1 = %s\n", s1);
    System.out.printf("s2 = %s\n", s2);
    System.out.printf("s3 = %s\n", s3);
  }
}
```

```
Output:

???
```

Converting Numbers to Strings

- To convert an integer to a string, use:
  - **Integer.toString** method.
  - **String.valueOf** method.
  - Concatenation.

```
public class example1 {
  public static void main(String[] args)
  {
    int a = 25;
    String s1 = Integer.toString(a);
    String s2 = String.valueOf(a);
    String s3 = "" + a;
    System.out.printf("s1 = %s\n", s1);
    System.out.printf("s2 = %s\n", s2);
    System.out.printf("s3 = %s\n", s3);
  }
}
```

```
Output:

s1 = 25
s2 = 25
s3 = 25
```

Converting Numbers to Strings

- To convert an double to a string, use:
  - **Double.toString** method.
  - **String.valueOf** method.
  - Concatentation

```
public class example1 {
  public static void main(String[] args)
  {
    double b = 8.12;
    String s1 = Double.toString(b);
    String s2 = String.valueOf(b);
    String s3 = "" + b;
    System.out.printf("s1 = %s\n", s1);
    System.out.printf("s2 = %s\n", s2);
  }
}
```

```
Output:

???
```

Converting Numbers to Strings

- To convert an double to a string, use:
  - **Double.toString** method.
  - **String.valueOf** method.
  - Concatentation

```
public class example1 {
  public static void main(String[] args)
  {
    double b = 8.12;
    String s1 = Double.toString(b);
    String s2 = String.valueOf(b);
    String s3 = "" + b;
    System.out.printf("s1 = %s\n", s1);
    System.out.printf("s2 = %s\n", s2);
  }
}
```

```
Output:

s1 = 8.12
s2 = 8.12
s3 = 8.12
```

Converting to Upper and Lower Case

- To convert a string to upper case, use the **toUpperCase** method.
- To convert a string to lower case, use the **toLowerCase** method.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "January has 31 days and is COLD!!!";
    String s2 = s1.toUpperCase();
    System.out.printf("%s\n", s2);
    String s3 = s1.toLowerCase();
    System.out.printf("%s\n", s3);
  }
}
```

```
Output:

???
```

Converting to Upper and Lower Case

- To convert a string to upper case, use the **toUpperCase** method.
- To convert a string to lower case, use the **toLowerCase** method.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "January has 31 days and is COLD!!!";
    String s2 = s1.toUpperCase();
    System.out.printf("%s\n", s2);
    String s3 = s1.toLowerCase();
    System.out.printf("%s\n", s3);
  }
}
```

```
Output:

JANUARY HAS 31 DAYS AND IS COLD!!!
january has 31 days and is cold!!!
```

(Very) Common Mistake

- What is wrong with this code?
- What will it print?

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    s1.toUpperCase();
    System.out.printf("%s\n", s1);
    s1.toLowerCase();
    System.out.printf("%s\n", s1);
  }
}
```

```
Output:

???
```

(Very) Common Mistake

- What is wrong with this code?
- What will it print?
- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    s1.toUpperCase();
    System.out.printf("%s\n", s1);
    s1.toLowerCase();
    System.out.printf("%s\n", s1);
  }
}
```

```
Output:

Hello
Hello
```

One Way to Fix the Mistake

- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.
- In this example, we store the results of toUpperCase and toLowerCase into s2.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    String s2 = s1.toUpperCase();
    System.out.printf("%s\n", s2);
    s2 = s1.toLowerCase();
    System.out.printf("%s\n", s2);
  }
}
```

```
Output:

???
```

One Way to Fix the Mistake

- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.
- In this example, we store the results of toUpperCase and toLowerCase into s2.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    String s2 = s1.toUpperCase();
    System.out.printf("%s\n", s2);
    s2 = s1.toLowerCase();
    System.out.printf("%s\n", s2);
  }
}
```

```
Output:

HELLO
hello
```

A Second Way to Fix the Mistake

- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.
- In this example, we directly call s1.toUpperCase and s1.toLowerCase in the second argument of printf.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    System.out.printf("%s\n", s1.toUpperCase());
    System.out.printf("%s\n", s1.toLowerCase());
  }
}
```

```
Output:

???
```

A Second Way to Fix the Mistake

- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.
- In this example, we directly call s1.toUpperCase and s1.toLowerCase in the second argument of printf.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    System.out.printf("%s\n", s1.toUpperCase());
    System.out.printf("%s\n", s1.toLowerCase());
  }
}
```

```
Output:

HELLO
hello
```

A Third Way to Fix the Mistake

- **s1.toUpperCase DOES NOT CHANGE s1.**
  - Strings are immutable; they can't be changed in place.
  - It creates a new string, that must be stored in a variable.
  - Same goes for s1.toLowerCase.
- In this example, we want to keep the s1 variable name and make it appear as if the string has changed.

```
public class example1 {
  public static void main(String[] args)
  {
    String s1 = "Hello";
    s1 = s1.toUpperCase();
    System.out.printf("%s\n", s1);
    s1 = s1.toLowerCase();
    System.out.printf("%s\n", s1);
  }
}
```

```
Output:

HELLO
hello
```