Strings, I/O, & String Comparison Variables – Instance and Local

January 26, 2004

Outline

- Strings
- String Operators
- Printing Strings
- Converting Strings to ints and doubles
- Formatting Output
- Reading Input
- Variables
- Local Variables
- Instance Variables

Strings

- Strings are a sequence of characters
- String is a Java class
  - Has methods that perform actions on the String
- Syntax:

  String stringName = “This is a string”;

String Operations

- Length: int len = stringName.length();
- Substring:

  String sub = stringName.subString(0,5);
  - The first integer is the starting position
  - The second integer is the ending position + 1
  - So, sub = “This ”

  String sub2 = stringName.subString(1);
  - sub2 = “his is a string”
  - String starts at specified index and goes to end of string
  - Note: All indexes in Java start at 0!

String Operators (2)

- Case Conversion

  String upper = stringName.toUpperCase();

  String lower = stringName.toLowerCase();
- Concatenation – joins two strings together

  String s = “This is” + “ a concatenated string”;
  - s = “This is a concatenated string”

  String s2 = stringName + “ example.”;
  - s2 = “This is a string example.”

Printing Strings

- Use System.out.print(<*string goes here*>); or System.out.println(<*string goes here*>); to print strings to the standard output
- Ex:

  System.out.println(“A string”);

  System.out.println(stringName);

Printing Strings (2)

- If you wish to print out a number you need to concatenate the number to a string.
- Numbers are automatically cast to a String.
- Ex:

  System.out.println(“Some text “ + stringName + “ “ + stringName.length());

  System.out.println(124);
  - Like System.out.println(Integer.toString(124));

Converting Strings to ints and doubles

- String to int

  int age = Integer.parseInt(“19”);
- String to double

  double money = Double.parseDouble(“2.45”);
- If the string cannot be converted to the specified data type a NumberFormatException will occur

Formatting Output

- Use the NumberFormat class
- Import java.text.NumberFormat
- Define NumberFormat object:
  NumberFormat nf = NumberFormat.getNumberInstance();
- Define a NumberFormat currency object:
  NumberFormat cnf = NumberFormat.getCurrencyInstance();
- Define a NumberFormat percent object:
  NumberFormat pnf = NumberFormat.getPercentInstance();

Formatting Output (2)

- Minimum number of decimal digits
  nf.setMinimumFractionDigits(3);
- Maximum number of decimal digits
  nf.setMaximumFractionDigits(3);
- Don't need to set number of digits for a currency number formatter

Formatting Output (3)

- Formatting Output for NumberFormat
  double money = 1.35573456;
  System.out.println(nf.format(money));
  - The printed output is “1.356”
- Formatting Output for currency NumberFormat
  double money = 1.35573456;
  System.out.println(cnf.format(money));
  - The printed output is “$1.36”

Reading Input

- User BufferedReader and InputStreamReader classes
- System.in specifies that input comes from standard input
- Syntax:

```
BufferedReader console =

new BufferedReader(new InputStreamReader(System.in));
```

Reading Input (2)

- Use BufferedReader's readLine() method to read input from standard in
- Ex:

```
String input = console.readLine();
```


Reading Input (3)

- This can cause exceptions, so wrap the call to the readLine() method in a try-catch block
- Ex:

```
try {
    String input = console.readLine();
}
catch (IOException e) {
    System.out.println("Error: " + e);
}
```

Variables

- Holds a value
- Two types of variables:
  - Local Variables
  - Instance Variables
- Access Modifiers – assign the level of access of variables, methods, and classes
  - Public – anyone can use
  - Private – accessible only to instances of a class

Local Variables

- Declared inside of a method
- Exist only when the method is running
- Independent of variables in other methods (even those with the same name)
- Should be given an initial value when declared
- No access modifier

Instance Variables

- Declared inside a class
- Exist while an object exists
- Can be accessed by all methods in a class
- Can have an access modifier (mostly private)
- Should have an initial value given – in Constructor

Variables

- You can have local and instance variables of the same name
- In a method, the local variable is used automatically when the variable name is called
- To call an instance variable use the this keyword
  - Ex:

  this.variableName

References

- Jason Schwarz's Lecture 4 and 5 slides: http://courses.ncsu.edu/csc116/
- Java API for Integer, Double, NumberFormat, BufferedReader, InputStreamReader, and String classes
  http://java.sun.com/j2se/1.4.2/docs/api/