Algorithms and Java basics: pseudocode, variables, assignment, and interactive programs

CSC 1051 – Algorithms and Data Structures I

Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website: www.csc.villanova.edu/~map/1051/

Some slides in this presentation are adapted from the slides accompanying:
• Java Software Solutions by Lewis & Loftus
• Introduction to Programming in Java: An Interdisciplinary Approach by Robert Sedgewick and Kevin Wayne

Algorithms and Java basics: pseudocode, variables, assignment, and interactive programs

Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website: www.csc.villanova.edu/~map/1051/

Some slides in this presentation are adapted from the slides accompanying:
• Java Software Solutions by Lewis & Loftus
• Introduction to Programming in Java: An Interdisciplinary Approach by Robert Sedgewick and Kevin Wayne

Algorithms

An algorithm is a specific set of instructions for carrying out a procedure or solving a problem, usually with the requirement that the procedure terminate at some point. Specific algorithms sometimes also go by the name method, procedure, or technique. The word "algorithm" is a distortion of al-Khwārizmī [named after Muhammad ibn al-Khwārizmī], a Persian mathematician who wrote an influential treatise about algebraic methods.

Sources: http://mathworld.wolfram.com/Algorithm.html and Wikipedia (http://en.wikipedia.org/wiki/Muhammad_ibn_M%C4%81s%C4%81_M%C5%ABs%C4%81_al-Khw%C4%81rizm%C4%AB)

Algorithm Example: Input-Compute-Output pattern

GPA problem: Write a program that computes and outputs the GPA, given the credits and quality points earned.

Variables: qp, credits, gpa

Algorithm:
1. qp = input from user
2. credits = input from user
3. gpa = qp / credits
4. Print gpa
// GPA.java  Author: Joyce/Papalaskari
// Demonstrates the use of Scanner.
******************************************************************************
import java.util.Scanner;
public class GPA {
    public static void main(String[] args) {
        // Inputs quality points and credits and calculates GPA.
        double qp, credits, gpa;
        Scanner scan = new Scanner(System.in);
        // input qp
        System.out.print("Enter Quality Points > ");
        qp = scan.nextInt();
        // input credits
        System.out.print("Enter Credits > ");
        credits = scan.nextInt();
        // calculate GPA
        gpa = qp / credits;
        // print GPA
        System.out.println("GPA:  "+ gpa);
    }
}

Java Program

Algorithm

Variables: qp, credits, gpa

Algorithm:
1. qp = input from user
2. credits = input from user
3. gpa = qp / credits
4. Print gpa

Interactive Programs – Input/Output

• Programs can use data obtained during runtime, eg:
  int age;
  String name;

  Scanner scan = new Scanner(System.in);

  System.out.print("Enter your name");
  name = scan.nextLine();
  System.out.print("Enter your age");
  age = scan.nextInt();

• In Java, you first need to create a Scanner object

  Scanner scan = new Scanner(System.in);

  System.out.print("Enter your name");
  name = scan.nextLine();
  System.out.print("Enter your age");
  age = scan.nextInt();

Interactive Programs – Input/Output

• The Scanner class is part of the java.util class library, and must be imported into a program in order to be used

• The import statement goes at beginning of your program (above class definition)

  import java.util.Scanner;
Interactive Programs – Input/Output

Summary:
1. import the Scanner class, i.e., add this before the class
definition of your program:
   ```java
   import java.util.Scanner;
   ```
2. In your main method, before doing any input, declare and
   initialize the Scanner object
   ```java
   Scanner scan = new Scanner(System.in);
   ```
3. Input away!
   ```java
   System.out.print("Enter your name");
   name = scan.nextLine();
   System.out.print("Enter your age");
   age = scan.nextInt();
   ```

Variables & Assignment

- Variable. A name that refers to a value of declared type.
- Assignment statement. Associates a value with a variable.

```java
// declaration statement
int age;
age = 18;

// assignment statement
double x = 3.2, y = -0.80;

// combined declaration and assignment statement
final int INCHES_PER_FOOT = 12;
String name = scan.nextLine();
```

Scanner methods

- `nextInt()` → input an int
- `nextDouble()` → input a double
- `nextLine()` → input a String (until end of line)
- `next()` → input a String token (one word or
  other delimited “chunk” of text)
- White space (space, tab, new line) are used to
  separate input tokens

Example

```java
public class TellMeAboutYou {
    public static void main(String[] args) {
        int age;
        String name;
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter your name");
        name = scan.nextLine();
        System.out.print("Enter your age");
        age = scan.nextInt();
        System.out.println("Pleased to meet you, " + name + "!");
        System.out.println("Your age in dog years: " + age*10.5);
    }
}
```

Enter your name: Fiona
Enter your age: 17
Pleased to meet you, Fiona!
Your age in dog years is 178.5

Inspired by: http://www.onlineconversion.com/dogyears.htm
Variable Declaration

• A variable is a name for a location of data in memory
• A variable must be declared by specifying the variable’s name and the type of information that it will hold

int age;
double x, y;
String name;

Assignment Statement

• Changes the value of a variable
• The assignment operator is the = sign

total = 55 - discount;

• The expression on the right is evaluated and the result is stored in the variable on the left

Some types of data in Java

<table>
<thead>
<tr>
<th>Type</th>
<th>Set of Values</th>
<th>Literal Values</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>char</td>
<td>characters</td>
<td>&quot;A&quot;, &quot;@&quot;</td>
<td>compare</td>
</tr>
<tr>
<td>String</td>
<td>sequences of characters</td>
<td>&quot;Hello World&quot;, &quot;jackie123&quot;</td>
<td>concatenate</td>
</tr>
<tr>
<td>int</td>
<td>integers</td>
<td>17, 12345</td>
<td>add, subtract, multiply, divide</td>
</tr>
<tr>
<td>double</td>
<td>floating-point numbers</td>
<td>3.1415, 6.022e23</td>
<td>add, subtract, multiply, divide</td>
</tr>
<tr>
<td>boolean</td>
<td>truth values</td>
<td>true, false</td>
<td>and, or, not</td>
</tr>
</tbody>
</table>

Combined declaration and assignment

A variable can be given an initial value in the declaration

int age = 18;
double x = 3.2, y = -0.80;
String name = scan.nextLine();
Combined declaration and assignment

A variable can be given an initial value in the declaration - a new value can be assigned later:

```java
int age = 18;
double x = 3.2, y = -0.80;
String name = scan.nextLine();
age = 19;
x = x + 0.5;
name = scan.nextLine();
```

Combined declaration and assignment – Note: CANNOT declare twice

A variable can be given an initial value in the declaration - a new value can be assigned later:

```java
int age = 18;
double x = 3.2, y = -0.80;
String name = scan.nextLine();
int age = 19;
```

Error: declaring variable age again

Example

Computing the total number of seconds

```java
int hours = 1;
int minutes = 25;
int seconds = 31;
int totalMinutes = (hours * 60) + minutes;
int totalSeconds = (totalMinutes * 60) + seconds;
```

Example

Computing the total number of seconds

Another alternative:

```java
int hours = 1;
int minutes = 25;
int seconds = 31;
int totalSeconds = (hours * 3600) + (minutes * 60) + seconds;
```
Arithmetic Operators

- Addition +
- Subtraction -
- Multiplication *
- Division /
- Remainder %

- If either or both operands used by an arithmetic operator are floating point (e.g., type double), then the result is a floating point.

Division and Remainder

- If both operands are integers (e.g., type int), the division result is an integer (the fractional part is discarded):
  
<table>
<thead>
<tr>
<th>Expression</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>14 / 3</td>
<td>4</td>
</tr>
<tr>
<td>143 / 60</td>
<td>2</td>
</tr>
<tr>
<td>20 / 16</td>
<td>1</td>
</tr>
<tr>
<td>8 / 12</td>
<td>0</td>
</tr>
</tbody>
</table>

% gives the remainder of the division:

Example

Extracting hours, minutes seconds from total number of seconds

```c
int totalSeconds = 7222;
int hours = totalSeconds / 3600;
int remainingSeconds = totalSeconds % 3600;
int minutes = remainingSeconds / 60;
int seconds = remainingSeconds % 60;
```

Operator Precedence

What is the order of evaluation of sub-expressions?

1. Multiplication, division, remainder
2. Addition, subtraction, string concatenation
   - Operators with the same precedence: left -> right
   - Use parentheses to override default order

More examples:

```
result = total + count / max - offset;
```

```
a + b + c + d + e
a / (b + c) - d % e
a - b / c + d * e
a / (b * (c + (d - e)))
```
Tracing the values of variables after each statement.

```java
int age = 18;
double x;
String name = "Sherlock";
age = 19;
x = 0.5;
x = x + 0.2;
name = name + "Holmes";
```

Trace: A table of variable values after each statement.

<table>
<thead>
<tr>
<th>age</th>
<th>x</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>18</td>
<td>undefined</td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>undefined &quot;Sherlock&quot;</td>
</tr>
<tr>
<td>0.5</td>
<td>19</td>
<td>&quot;Sherlock&quot;</td>
</tr>
<tr>
<td>0.7</td>
<td>19</td>
<td>&quot;SherlockHolmes&quot;</td>
</tr>
</tbody>
</table>

Final values:

Trace: TRY THIS:

```java
int a, b;
a = 3;
b = 4;
a = b;
double pi = 3.14;
```

Final values:

a  b  pi
---
---
---

Trace: TRY THIS:

```java
int a, b;
a = 3;
b = 4;
int c = a;
a = b;
b = 5;
b = c;
```

Final values:

a  b  c
---
---
---
Assignment operator

- Assignment ( = ) copies the value of the right side into the memory location associated with the left side
- It does not set up an ongoing equivalence

```java
int davesAge = 21;
int susesAge = davesAge;
davesAge = 22;
System.out.println (davesAge); // prints 22
System.out.println (suesAge); // prints 21
```

Increment and Decrement

- The increment operator (++) adds one to its operand
- The decrement operator (--) subtracts one from its operand
- The statement
  ```java
count++;
```
is functionally equivalent to
  ```java
count = count + 1;
```

CONSTANTS: like variables, but value cannot change – declare using `final` modifier:

```java
final int INCHES_PER_FOOT = 12;
final double LBS_PER_KG = 2.2;
```

Variables & Assignment

- `Variable`. A name that refers to a value of declared type.
- `Assignment statement`. Associates a value with a variable.

```
type variable declaration statement
int age;            // declaration statement
age = 18;           // assignment statement
double x = 3.2, y = -0.80;   // combined declaration and assignment statement
final int INCHES_PER_FOOT = 12; // constant declaration (always initializes value)
String name = scan.nextLine();     // input from user
```