Building Java Programs

`Scanner`; `if`/`else`; `while` loops; random

reading: 3.3 – 3.4, 4.1, 4.5, 5.1, 5.6

Interactive Programs with `Scanner`

reading: 3.3 - 3.4

Interactive programs

interactive program: Reads input from the console.

- While the program runs, it asks the user to type input.
- The input typed by the user is stored in variables in the code.

- Can be tricky; users are unpredictable and misbehave.
- But interactive programs have more interesting behavior.

Scanner

- **`Scanner`**: An object that can read input from many sources.
  - Communicates with `System.in`
  - Can also read from files (Ch. 6), web sites, databases, …

- The `Scanner` class is found in the `java.util` package.

```
import java.util.*;   // so you can use Scanner
```

- Constructing a `Scanner` object to read console input:

```
Scanner name = new Scanner(System.in);
```

  - Example:

```
Scanner console = new Scanner(System.in);
```

`Scanner` methods

Method	Description
`nextInt()`	reads an `int` from the user and returns it
`nextDouble()`	reads a `double` from the user
`next()`	reads a one-word `String` from the user
`nextLine()`	reads a one-*line* `String` from the user

- Each method waits until the user presses Enter.
- The value typed by the user is returned.

```
System.out.print("How old are you? ");  // prompt
int age = console.nextInt();
System.out.println("You typed " + age);
```

- **prompt**: A message telling the user what input to type.

Scanner example

```
import java.util.*;   // so that I can use Scanner

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

        System.out.print("How old are you? ");
        int age = console.nextInt();

        int years = 65 - age;
        System.out.println(years + " years until retirement!");
    }
}
```

age	29
years	36

- Console (user input underlined):

```
How old are you? 29
36 years until retirement!
```

Scanner example 2

```
import java.util.*;   // so that I can use Scanner

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

        System.out.print("Please type two numbers: ");
        int num1 = console.nextInt();
        int num2 = console.nextInt();

        int product = num1 * num2;
        System.out.println("The product is " + product);
    }
}
```

- Output (user input underlined):

```
Please type two numbers: 8 6
The product is 48
```

  - The `Scanner` can read multiple values from one line.

Input tokens

- **token**: A unit of user input, as read by the `Scanner`.
  - Tokens are separated by *whitespace* (spaces, tabs, new lines).
  - How many tokens appear on the following line of input?

```
23  John Smith  42.0  "Hello world"  $2.50  "  19"
```

- When a token is not the type you ask for, it crashes.

```
System.out.print("What is your age? ");
int age = console.nextInt();
```

Output:

```
What is your age? Timmy
java.util.InputMismatchException
        at java.util.Scanner.next(Unknown Source)
        at java.util.Scanner.nextInt(Unknown Source)
        ...
```

The `if/else` statement

reading: 4.1, 4.4 - 4.6

The `if/else` statement

Chooses between outcomes using many tests

Java:

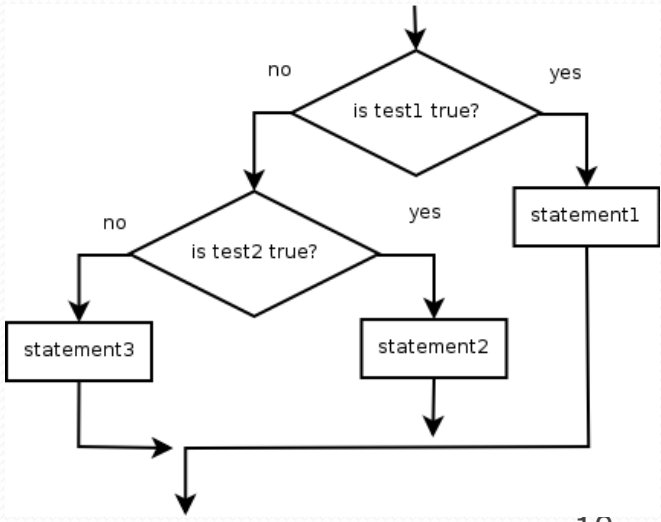
```
if (test) {
    statement(s);
} else if (test) {
    statement(s);
} else {
    statement(s);
}
```

Python:

```
if test:
    statement(s)
elif test:
    statement(s)
else:
    statement(s)
```

- Example:

```
if (x > 0) {
    System.out.println("Positive");
} else if (x < 0) {
    System.out.println("Negative");
} else {
    System.out.println("Zero");
}
```



Relational expressions

- `if` statements and `for` loops both use logical tests.

```
for (int i = 1; i <= 10; i++) { ...
if (i <= 10) { ...
```

  - These are `boolean` expressions, seen in Ch. 5.

- Tests use *relational operators*:

Operator	Meaning	Example	Value
==	equals	1 + 1 == 2	true
!=	does not equal	3.2 != 2.5	true
<	less than	10 < 5	false
>	greater than	10 > 5	true
<=	less than or equal to	126 <= 100	false
>=	greater than or equal to	5.0 >= 5.0	true

Logical operators

- Tests can be combined using *logical operators*:

Python	Java	Description	Example	Result
`and`	`&&`	and	`(2 == 3) && (-1 < 5)`	`false`
`or`	`||`	or	`(2 == 3) || (-1 < 5)`	`true`
`not`	`!`	not	`!(2 == 3)`	`true`

- "Truth tables" for each, used with logical values *p* and *q*:

p	q	p && q	p || q
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

p	!p
true	false
false	true

Nested `if/else` question

Formula for body mass index (BMI):

$$BMI = \frac{weight}{height^2} \times 703$$

BMI	Weight class
below 18.5	underweight
18.5 - 24.9	normal
25.0 - 29.9	overweight
30.0 and up	obese

- Write a program that produces output like the following:

```
This program reads height and weight data and
Computes body mass index (BMI).

Enter next person's information:
height (in inches)? 70.0
weight (in pounds)? 194.25

BMI = 27.868928571428572
overweight

Enter next person's information:
height (in inches)? 62.5
weight (in pounds)? 130.5

BMI = 23.485824
normal

Percentage of people normal weight = 0.5
```

Nested `if/else` answer

```
// This program computes two people's body mass index (BMI) and
// compares them.  The code uses Scanner for input, and parameters/returns.

import java.util.*;  // so that I can use Scanner

public class BMI {
    public static void main(String[] args) {
        System.out.println("This program reads height and weight data and");
        System.out.println("Computes body mass index (BMI).");

        Scanner console = new Scanner(System.in);
        int countNormal = 0;

        for (int i = 0; i < 2; i++) {
           System.out.println("Enter next person's information:");
           System.out.print("height (in inches)? ");
           double height = console.nextDouble();
           System.out.print("weight (in pounds)? ");
           double weight = console.nextDouble();

           double bmi = weight / (height * height) * 703;
           System.out.println("BMI = " + bmi);

           if (bmi < 18.5) {
              System.out.println("underweight");
           } else if (bmi < 25) {
              System.out.println("normal");
              countNormal++;
           } else if (bmi < 30) {
              System.out.println("overweight");
           } else {
              System.out.println("obese");
           }
        }
        System.out.println("Percentage of people normal weight = "
                           + countNormal/2.0);

    }
}
```

`while` loops

reading: 5.1

Categories of loops

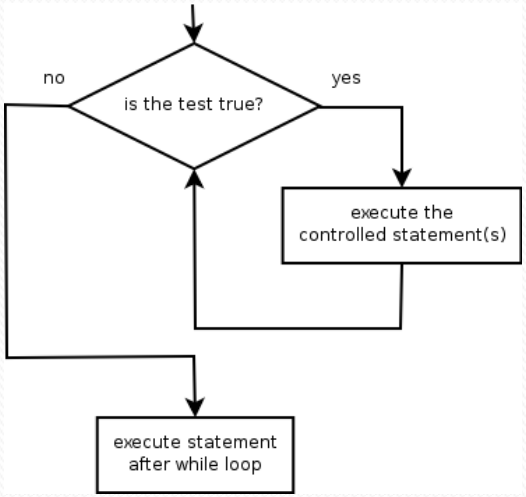
- **definite loop**: Executes a known number of times.
  - The `for` loops we have seen are definite loops.
    - Print "hello" 10 times.
    - Find all the prime numbers up to an integer *n*.
    - Print each odd number between 5 and 127.

- **indefinite loop**: One where the number of times its body repeats is not known in advance.
    - Prompt the user until they type a non-negative number.
    - Print random numbers until a prime number is printed.
    - Repeat until the user has typed "q" to quit.

The `while` loop

- **`while` loop**: Repeatedly executes its body as long as a logical test is true.

```
while (test) {
    statement(s);
}
```



- Example:

```
int num = 1;                          // initialization
while (num <= 200) {                  // test
    System.out.print(num + " ");
    num = num * 2;                    // update
}
// output:  1 2 4 8 16 32 64 128
```

Example `while` loop

```
// finds the first factor of 91, other than 1
int n = 91;
int factor = 2;
while (n % factor != 0) {
    factor++;
}
System.out.println("First factor is " + factor);

// output:  First factor is 7
```

- `while` is better than `for` because we don't know how many times we will need to increment to find the factor.

Sentinel values

- **sentinel**: A value that signals the end of user input.
  - **sentinel loop**: Repeats until a sentinel value is seen.

- Example: Alter the BMI program to ask the user if there is more data to input after each BMI is calculated. If the user doesn't wish to continue the user must enter the number 0.
  - (In this case, 0 is the sentinel value.)

```
Is there more information to enter? 1

Enter next person's information:
height (in inches)? 70.0
weight (in pounds)? 194.25

BMI = 27.868928571428572
overweight

Is there more information to enter? 42
...
```

Sentinel values answer

```
// This program computes two people's body mass index (BMI) and
// compares them.  The code uses Scanner for input, and parameters/returns.
import java.util.*;  // so that I can use Scanner

public class BMI {
    public static void main(String[] args) {
        System.out.println("This program reads height and weight data and");
        System.out.println("Computes body mass index (BMI).");

        Scanner console = new Scanner(System.in);
        int countNormal = 0;
        int total = 0;

        int again = 42;
        while (again != 0) {
           total++;
           System.out.println("Enter next person's information:");
           System.out.print("height (in inches)? ");
           double height = console.nextDouble();
           System.out.print("weight (in pounds)? ");
           double weight = console.nextDouble();

           double bmi = weight / (height * height) * 703;
           System.out.println("BMI = " + bmi);

           if (bmi < 18.5) {
              System.out.println("underweight");
           } else if (bmi < 25) {
              System.out.println("normal");
              countNormal++;
           } else if (bmi < 30) {
              System.out.println("overweight");
           } else {
              System.out.println("obese");
           }
          System.out.println("Is there more information to enter? ");
           again = console.nextInt();
        }
        System.out.println("Percentage of people normal weight = "
                           + countNormal / total);
    }
}
```

Random Numbers

reading: 5.1, 5.6

Randomness

- Lack of predictability: don't know what's coming next
- Random process: outcomes do not follow a deterministic pattern (math, statistics, probability)
- Lack of bias or correlation (statistics)
- Relevant in lots of fields
  - Genetic mutations (biology)
  - Quantum processes (physics)
  - Random walk hypothesis (finance)
  - Cryptography (computer science)
  - Game theory (mathematics)
  - Determinism (religion)

Pseudo-Randomness

- Computers generate numbers in a predictable way using a mathematical formula
- Parameters may include current time, mouse position
  - In practice, hard to predict or replicate
- True randomness uses natural processes
  - Atmospheric noise (http://www.random.org/)
  - Lava lamps (patent #5732138)
  - Radioactive decay

The `Random` class

- A `Random` object generates pseudo-random numbers.
  - Class `Random` is found in the `java.util` package.

```
import java.util.*;
```

Method name	Description
`nextInt()`	returns a random integer
`nextInt(`***max***`)`	returns a random integer in the range [0, *max*) in other words, 0 to *max*-1 inclusive
`nextDouble()`	returns a random real number in the range [0.0, 1.0)

  - Example:

```
Random rand = new Random();
int randomNumber = rand.nextInt(10);    // 0-9
```

Generating random numbers

- Common usage: to get a random number from 1 to *N*

```
int n = rand.nextInt(20) + 1;   // 1-20 inclusive
```

- To get a number in arbitrary range [*min*, *max*] inclusive:

name*.nextInt(*size of range*) + *min

  - Where ***size of range*** is (***max*** - ***min*** + 1)

  - Example: A random integer between 4 and 10 inclusive:

```
int n = rand.nextInt(7) + 4;
```

Random questions

- Given the following declaration, how would you get:

```
Random rand = new Random();
```

  - A random number between 1 and 47 inclusive?

```
int random1 = rand.nextInt(47) + 1;
```

  - A random number between 23 and 30 inclusive?

```
int random2 = rand.nextInt(8) + 23;
```

  - A random even number between 4 and 12 inclusive?

```
int random3 = rand.nextInt(5) * 2 + 4;
```

`Random` and other types

- `nextDouble` method returns a `double` between 0.0 - 1.0
  - Example: Get a random GPA value between 1.5 and 4.0:

```
double randomGpa = rand.nextDouble() * 2.5 + 1.5;
```

- Any set of possible values can be mapped to integers
  - code to randomly play Rock-Paper-Scissors:

```
int r = rand.nextInt(3);
if (r == 0) {
    System.out.println("Rock");
} else if (r == 1) {
    System.out.println("Paper");
} else {  // r == 2
    System.out.println("Scissors");
}
```

Random question

- Write a program that simulates rolling two 6-sided dice until their combined result comes up as 7.

```
2 + 4 = 6
3 + 5 = 8
5 + 6 = 11
1 + 1 = 2
4 + 3 = 7
You won after 5 tries!
```

Random answer

```
// Rolls two dice until a sum of 7 is reached.
import java.util.*;

public class Dice {
    public static void main(String[] args) {
        Random rand = new Random();
        int tries = 0;

        int sum = 0;
        while (sum != 7) {
            // roll the dice once
            int roll1 = rand.nextInt(6) + 1;
            int roll2 = rand.nextInt(6) + 1;
            sum = roll1 + roll2;
            System.out.println(roll1 + " + " + roll2 + " = " + sum);
            tries++;
        }

        System.out.println("You won after " + tries + " tries!");
    }
}
```