Review of Chapter 1&2

Cheng, Wei   COMP110-001   May 16, 2014
Today’s Contents

• Review of Chapter 1 & 2
• Review programs in lectures & labs
Measuring Data

- **1 bit** (binary digit): one of two values (0 or 1).
- **1 byte**: 8 bits.
  - 00000000 ~ 11111111
  - \(2^8 = 256\) possible states

- A sample byte:
  - 01010010
  - As decimal number: \(82 = 2^1 + 2^4 + 2^6\)
Measuring Data

• 4 bytes: $4 \times 8 = 32$ bits
  – $2^{32}$ possible states (counting from 1)

• Size of int type in Java

• If we use 4 bytes to represent an integer, what is the range?
  – Unsigned: $0 \sim 2^{32}-1$ (counting from 0)
  – Signed: $-2^{31} \sim 2^{31}-1$
## Primitive Types (In This Course)

<table>
<thead>
<tr>
<th>Type</th>
<th>Size</th>
<th>Example</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>4 bytes</td>
<td>3443, -1024</td>
<td>Integer only. Smaller range +, -, *, /, %</td>
</tr>
<tr>
<td>double</td>
<td>8 bytes</td>
<td>-0.4, 3.2, 3.343*10^100</td>
<td>Much larger range. Limited precision +, -, *, /, %</td>
</tr>
<tr>
<td>boolean</td>
<td>1 bit in theory</td>
<td>true, false</td>
<td>And or negation &amp; &amp;</td>
</tr>
</tbody>
</table>
Variables

• Container of Data
  – Data can be of Class type or Primitive type

• Declaration of Variable:

  type variable_name;

  type variable_name = initial_value;

  e.g.:

  int i = 1;
  boolean passedTest = true;
  Polygon triangle = new Polygon();

  int age;
  double length;
  char letter;

main memory
Using Variables

• Specify the **type** only once at declaration
• In assignment operation, the right side is evaluated first. The value is then stored into the left side.
• Swap values of two integer variables:

```c
int a = 10; int b = 5;
```

**Method 1:**

```c
int c = a;
```

```c
a = b;
```

```c
b = c;
```

**Method 2**

```c
a = a + b;
```

```c
b = a - b;
```

```c
a = a - b;
```
Compatibility and Type Casting

- Variable of “Bigger” type can hold values of “smaller” type.
  ```java
  int a = 5;
  double b = a;
  ```

- One can cast one type into another type (at the risk of losing information)
  ```java
  double a = 10.0 / 3.0; // a = 3.3333333333333333
  int b = (int)a; // b = 3
  ```
Object Oriented Programming (OOP)

- Object: Attributes + Methods
- Class: the blueprint of objects of the same type

![Diagram of Object-Oriented Programming]

**Class**
- Person
  - name, contact

**Subclass**
- Student
  - student ID, program, year

**Objects**
- S1: name="Alan", contact="919-.....", program = biostat, year = 1st
- S2: name="Anna", contact="919-.....", program = CS, year = 1st
- T1: name="Yi", contact="919-.....", program = CS, dept = CS, rank = no rank
- T2: name="Yun", contact="919-.....", program = biostat, dept = CS, rank = ast prof
OOP in Practice

• Import class if necessary:
  
  import java.util.*;

• Create object:
  
  Class_Type variable_name = new Class_Type( ... );
  e.g.:
  Scanner s = new Scanner(System.in);
  Polygon treeTop = new Polygon();

• Access object members (attribute or method):
  
  int inputNumber = s.nextInt();
  treeTop.setColor( Color.green );
String

• A special class. Object of String class can be defined using shortcut:

```java
String str = "UNC is Great";
```

• Each String object consists of:
  – A sequence of characters (char)

<table>
<thead>
<tr>
<th>String Indices</th>
<th>U</th>
<th>N</th>
<th>C</th>
<th>i</th>
<th>s</th>
<th>G</th>
<th>r</th>
<th>e</th>
<th>a</th>
<th>t</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>

  – A set of methods that can process the sequence of characters
String

• Concatenation by “+”
  – “My name is “ + “Yi” → “My name is Yi”

• Mixed operations:
  – “The sum is ” + 5 + 6 → “The sum is 56”
  – “The sum is ” + (5 + 6) → “The sum is 11”

• A lot of methods. See Java API for reference:
  – length(), substring(), charAt(), toLowerCase() .....
Console I/O

• Two built-in Java objects:
  System.out
  System.in

• Console output:
  System.out.print(...);
  System.out.println(....);

• Console input – use Scanner class:
  Scanner s = new Scanner(System.in);
  int inputNumber = s.nextInt();
  String name = s.next();
SecondProgram.java

```java
import java.util.*;
public class SecondProgram {
    public static void main(String[] args) {
        System.out.println("Hi, What's your name?\n");
        Scanner s = new Scanner(System.in);
        String name = s.next();
        System.out.println(name + ", welcome to COMP110\n");
    }
}
```

<terminated> SecondProgram [Java Application] C:\Program Files\Java\jre: Hi, What's your name?
Yi
Yi, welcome to COMP110
```
import java.util.*;

public class lab0 {

    public static void main(String[] args) {
        int n1, n2;
        System.out.println ("Hello out there! I made you a simple Math program");
        System.out.println ("Please enter two integers on the line");

        Scanner keyboard = new Scanner(System.in);
        n1= keyboard.nextInt();
        n2= keyboard.nextInt();

        System.out.println("The sum is:");
        System.out.println(n1 + n2);

        System.out.println("The difference is:");
        System.out.println(n1 - n2);

        System.out.println("The product is:");
        System.out.println(n1*n2);

        System.out.println("The quotient is:");
        System.out.println(n1/n2);
    }
}
```
import java.awt.Color;
public class Lab1 {
    public static void main(String[] args) {
        // Create the main canvas object and set its background color
        LabCanvas canvas = new LabCanvas();
        canvas.setBackgroundColor( new Color( 0.8f, 0.9f, 1.0f ) );

        // Create a triangle (3 vertices) of green color. Then add it to the canvas
        Polygon treeTop = new Polygon();
        treeTop.addVertex(200, 200);
        treeTop.addVertex(300, 400);
        treeTop.addVertex(100, 400);
        treeTop.setColor( new Color(0.0f, 0.4f, 0.0f) );
        canvas.addCanvasItem(treeTop);

        // Create a rectangle (4 vertices) of brown color. Then add it to the canvas
        Polygon treeStem = new Polygon();
        treeStem.addVertex(160, 400);
        treeStem.addVertex(240, 400);
        treeStem.addVertex(240, 600);
        treeStem.addVertex(160, 600);
        treeStem.setColor( new Color(0.59f, 0.29f, 0.0f) );
        canvas.addCanvasItem(treeStem);

        // Create a yellow oval. Then add it to the canvas
        Oval sun = new Oval();
        sun.setColor(Color.yellow);
        sun.setLocation(600, 100);
        sun.setWidth(100);
        sun.setHeight(100);
        canvas.addCanvasItem(sun);
Next Class

• String operations

• Review slides on String

• Go through the list of methods of String in Java API