# 1.00/1.001
## Spring 2012
### Quiz 1 Review

#### Course Topics Overview

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><strong>Control and scope</strong></td>
<td>2.</td>
<td><strong>Classes and objects</strong></td>
</tr>
<tr>
<td>3.</td>
<td><strong>Arrays, ArrayLists</strong></td>
<td>4.</td>
<td><strong>Recursion</strong></td>
</tr>
<tr>
<td>5.</td>
<td><strong>Inheritance</strong></td>
<td>6.</td>
<td><strong>Graphical user interfaces</strong></td>
</tr>
<tr>
<td>7.</td>
<td><strong>Numerical methods</strong></td>
<td>8.</td>
<td><strong>Input/output streams</strong></td>
</tr>
<tr>
<td>9.</td>
<td><strong>Sensors and threads</strong></td>
<td>10.</td>
<td><strong>Data structures</strong></td>
</tr>
</tbody>
</table>

*Quiz 1*

*Quiz 2*

*Final*
1. Control and Scope

1. Data types
   - Promotion and casting

2. Operators
   - Precedence
   - Numerical problems

3. Control structures
   - Branching (if/else)
   - Iteration (while/do while/for)

4. Methods
   - Argument passing
   - Variable scope

<table>
<thead>
<tr>
<th>Type</th>
<th>Size (bits)</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>boolean</td>
<td>1</td>
<td>true or false</td>
</tr>
<tr>
<td>char</td>
<td>16</td>
<td>ISO Unicode character set</td>
</tr>
<tr>
<td>byte</td>
<td>8</td>
<td>-128 to 127</td>
</tr>
<tr>
<td>short</td>
<td>16</td>
<td>-32,768 to 32,767</td>
</tr>
<tr>
<td>int</td>
<td>32</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
<tr>
<td>long</td>
<td>64</td>
<td>-9,223,372,036,854,775,808L to 9,223,372,036,854,775,807L</td>
</tr>
<tr>
<td>float</td>
<td>32</td>
<td>1.4E-45F to 3.4E+38F (6-7 significant digits)</td>
</tr>
<tr>
<td>double</td>
<td>64</td>
<td>4.9E-324 to 1.8E+308 (15 significant digits)</td>
</tr>
</tbody>
</table>
1. Control and Scope

1. Data types
   • Promotion and casting
2. Operators
   • Precedence
   • Numerical problems
3. Control structures
   • Branching (if/else)
   • Iteration (while/do while/for)
4. Methods

---

<table>
<thead>
<tr>
<th>Promotion example</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 / 3 – 4 / 3.0</td>
</tr>
<tr>
<td>int</td>
</tr>
<tr>
<td>int</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>double</td>
</tr>
<tr>
<td>– 0.333</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Casting example</th>
</tr>
</thead>
<tbody>
<tr>
<td>(double) 4 / 3 – 4 / 3.0</td>
</tr>
<tr>
<td>double</td>
</tr>
</tbody>
</table>

---

1. Control and Scope

1. Data types
   • Promotion and casting
2. Operators
   • Precedence
   • Numerical problems
3. Control structures
   • Branching (if/else)
   • Iteration (while/do while/for)
4. Methods
   • Argument passing
   • Variable scope

---

<table>
<thead>
<tr>
<th>Arithmetic operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
</tr>
<tr>
<td>--</td>
</tr>
<tr>
<td>* (unary)</td>
</tr>
<tr>
<td>- (unary)</td>
</tr>
<tr>
<td>/</td>
</tr>
<tr>
<td>%</td>
</tr>
<tr>
<td>+</td>
</tr>
<tr>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Boolean operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equal</td>
</tr>
<tr>
<td>Less than</td>
</tr>
<tr>
<td>Greater than</td>
</tr>
<tr>
<td>Logical and</td>
</tr>
</tbody>
</table>
1. Control and Scope

1. Data types
   - Promotion and casting
2. Operators
   - Precedence
   - Numerical problems
3. Control structures
   - Branching (if/else)
   - Iteration (while/do while/for)
4. Methods
   - Argument passing
   - Variable scope

Common problems
- Integer divide
- Divide by zero
- $0 / 0 = \text{NaN}$
- Exceeding capacity of data type
- Decimal imprecision and error

if / else example
boolean b = 3 > 4;
String s = "hello";
if (b) {
    ...
} else if (s.equals("bye")) {
    ...
} else {
    ...
}
1. Control and Scope

1. Data types
   • Promotion and casting

2. Operators
   • Precedence
   • Numerical problems

3. Control structures
   • Branching (if/else)
   • **Iteration (while/do while/for)**

4. Methods
   • Argument passing
   • Variable scope

---

### Writing a method

**public static double power(double x, int n)**

- **Access**
- **Object or class-based**
- **Return type**
- **Name**
- **Arguments (input)**
1. Control and Scope

1. Data types
   • Promotion and casting

2. Operators
   • Precedence
   • Numerical problems

3. Control structures
   • Branching
   • Iteration

4. Methods
   • *Argument passing*
   • Variable scope

---

```java
public class Scope {
    private int num;

    public Scope(int n) {
        num = n;
    }

    public double calcSomething(int a) {
        double result = 0;
        for (int i = 1; i < a; i++) {
            result += num * i;
        }
        return result;
    }

    public static void main(String[] args) {
        int n = 4;
        Scope s = new Scope(n);
        double d = s.calcSomething(n);
    }
}
```

Scope error!
2. Classes and Objects

1. Making a class
   - Data members
   - Constructor
   - Methods
2. Creating objects
   - Memory allocation
3. Calling methods
   - Passing object arguments
4. Access
5. Static vs. non-static

public class Pie {
    private int apples;
    private double cupsSugar;
    private Crust myCrust;

    public Pie(int a, double s, Crust c) {
        apples = a;
        cupsSugar = s;
        myCrust = c;
    }

    public double calcWeight() {
        double appleWt = apples * 0.5;
        double sugarWt = cupsSugar * 0.3;
        double crustWt = myCrust.getWt();
        return appleWt + sugarWt + crustWt;
    }
}
2. Classes and Objects

1. Making a class
   • Data members
   • Constructor
   • Methods

2. Creating objects
   • Memory allocation

3. Calling methods
   • Passing object arguments

4. Access

5. Static vs. non-static

```java
public class Test{
    public static void main(String[] args){
        Crust lattice = new Crust(...);
        Crust flat = new Crust(...);
        Pie p1 = new Pie(4, 1.5, lattice);
        Pie p2 = new Pie(8, 3, flat);
    }
}
```

```java
public class Test{
    public static void main(String[] args){
        Crust lattice = new Crust(...);
        Crust flat = new Crust(...);
        Pie p1 = new Pie(4, 1.5, lattice);
        Pie p2 = new Pie(8, 3, flat);
        double w1 = p1.calcWeight();
    }
}
```

```java
public class Test{
    public static void main(String[] args){
        Crust lattice = new Crust(...);
        Crust flat = new Crust(...);
        Pie p1 = new Pie(4, 1.5, lattice);
        Pie p2 = new Pie(8, 3, flat);
        double w1 = p1.calcWeight();
    }
}
```
2. Classes and Objects

1. Making a class
   - Data members
   - Constructor
   - Methods
2. Creating objects
   - Memory allocation
3. Calling methods
   - **Passing object arguments**
4. Access
5. Static vs. non-static

```
public class Test{
    public static void main(String[] args){
        Crust lattice = new Crust(...);
        Crust flat = new Crust(...);
        Pie p1 = new Pie(4, 1.5, lattice);
        p1.setCrust(flat);
    }
}
```

```
public class Pie {
    ...
    public void setCrust(Crust c){
        myCrust = c;
    }
}
```

- **Private**: Access only within current class
- **Public**: Access from all classes in project
- **Package (default)**: Access from all classes in same package
- **Protected**: Used with inheritance (covered later)
2. Classes and Objects

1. Making a class
   • Data members
   • Constructor
   • Methods
2. Creating objects
   • Memory allocation
3. Calling methods
   • Passing object arguments
4. Access
5. *Static vs. non-static*

```java
public class Ticket {
    private static int total = 0;
    private int num;

    public Ticket() {
        num = 100 + total;
        total++;
    }

    //... total = 0
}
```

![Ticket object 1 Ticket object 2 Ticket object 3](num 100 num 101 num 102)

3. Arrays and ArrayLists

1. Differences between array and ArrayList
2. Declaration and initialization
3. Assigning / adding an element
4. Accessing an element
5. Retrieving the number of elements
6. Looping over elements
3. Arrays and ArrayLists

1. Differences between array and ArrayList
2. Declaration and initialization
3. Assigning / adding an element
4. Accessing an element
5. Retrieving the number of elements
6. Looping over elements

**Array**

All about brackets [ ]!

**ArrayList**

All about methods!

---

Array example

```java
double[] d = new double[5];
d[0] = 32.0; d[1] = 67.0; ...
System.out.println(d[1]);
int len = d.length;
for (double elem : d)
    System.out.println(elem);
for (int i=0; i<len; i++)
    System.out.println(d[i]);
```

ArrayList example

```java
ArrayList<Double> d = new ArrayList<Double>();
d.add(32.0); d.add(67.0); ...
System.out.println(d.get(1));
int len = d.size();
for (double elem : d)
    System.out.println(elem);
for (int i=0; i<len; i++)
    System.out.println(d.get(i));
```
1.00 / 1.001 / 1.002 Introduction to Computers and Engineering Problem Solving
Spring 2012

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.