Introduction to Computer Science I

Scanner, Increment/Decrement, Conversion

Janyl Jumadinova
September 19, 2016
The `Scanner` class in the `java.util` package is a simple text scanner which can parse primitive types and strings.

We can use the `Scanner` class to get the input from the terminal.

We must create an instance of the `Scanner` as:

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

where `name` is the name you choose for your instance of the `Scanner`
Scanner Methods

- `next()` : get the next word (token) as a String
- `nextLine()` : get a line of input as a String
- `nextInt()` : get an integer
- `nextDouble()` : get a double value
Memory Concepts

- Variable names such as first, second and sum correspond to locations in the computer’s memory.
Variable names such as `first`, `second` and `sum` correspond to locations in the computer’s memory.

Every variable has:
- a name,
- a type,
- a size,
- a value
Memory Concepts

- Variable names such as first, second and sum correspond to locations in the computer’s memory.
- Every variable has:
  - a name,
  - a type,
  - a size,
  - a value
- `first = input.nextInt();` stores whatever the user types into the location associated with first.
Memory Concepts

- Whenever a value is placed in a memory location, it replaces whatever was there before.
- This includes keyboard input (such as `first = input.nextInt()`) and assignment statements (such as `sum = first + second`).
Increment and Decrement Operators

- **++**: adds 1 to any value
  
  `count ++` same as `count = count + 1`
  
  prefix form: `++count`
  
  postfix form: `count++`

- **--**: subtracts 1 from any value
  
  `count --` same as `count = count - 1`
  
  prefix form: `--count`
  
  postfix form: `count--`
Increment and Decrement Operators

- **++**: adds 1 to any value
  - `count ++` same as `count = count + 1`
  - prefix form: `++count`
  - postfix form: `count++`

- **--**: subtracts 1 from any value
  - `count --` same as `count = count - 1`
  - prefix form: `--count`
  - postfix form: `count--`

- `total = count ++;` vs. `total = ++count;`
Increment and Decrement Operators

- `+= (-=, *=, ..): combine basic operation with assignment
  
  count += num same as count=count+num
Conversion

- **Widening**
  - from `byte` to `short`, `int`, `long`, `float` or `double`
  - from `short` to `int`, `long`, `float`, `double`
  - from `char` to `int`, `long`, `float`, `double`
  - from `int` to `long`, `float`, `double`
  - from `long` to `float`, `double`
  - from `float` to `double`
Conversion

- Narrowing - should be avoided!
  - from byte to char
  - from short to byte, char
  - from char to byte, short
  - from int to byte, short, char
  - from long to byte, short, char, int
  - from float to byte, short, char, int, long
  - from double to byte, short, char, int, long, float
Conversion

- **Assignment**: grade = ‘A’
- **Promotion**: total/count, where total is a floating point value and count is an integer
  - Occurs automatically, count is *promoted* to a floating point value
- **Casting**: grade = (int) total
  - Java operator: type name in parentheses
  - Casting converts floating point value total into an integer, truncating any fractional part.