Overloading Methods

Hui Chen

Department of Computer & Information Science

Brooklyn College

Objectives

- To use method overloading and understand ambiguous invocation (§6.8).

Outline

- Discussed
    - Defining and invoking value-returning methods
    - Defining and invoking void methods
    - Parameter passing and passing by value
    - Pitfalls and errors
    - Using method to modularize several example problems (including converting hexadecimal to decimal)
- To discuss
    - Method overload
    - Ambiguous method invocation

Overloading Methods

- We can write multiple methods with the same name but different parameter list

Method Overloading Example

- Overloading the `max` Method
  - The max method we implement is to find the maximum between two integers.
  - How about we want a method to find the maximum between two double values?

Overload the max method

```
public static double max(double
 num1, double num2) {
   if (num1 > num2)
      return num1;
   else
      return num2;
}
```

Questions?

Ambiguous Invocation

- Sometimes there may be two or more possible matches for an invocation of a method, but the compiler cannot determine the most specific match.
- This is referred to as *ambiguous invocation*. Ambiguous invocation is a compile error.
- Can you think of an example?

Example of Ambiguous Invocation

```
public class AmbiguousInvocation {
  public static void main(String[] args) {
    System.out.println(max(1, 2));
  }

  public static double max(int num1, double num2) {
    if (num1 > num2)
      return num1;
    else
      return num2;
  }

  public static double max(double num1, int num2) {
    if (num1 > num2)
      return num1;
    else
      return num2;
  }
}
```

How do we fix the problem?

Questions