Lecture 15

Enhanced For Loop

Dr. Martin O'Connor
CA166
www.computing.dcu.ie/~moconnor

Enhanced for Statement when using arrays

- The for statement also has another form designed for iteration (looping) through Collections (to be discussed later) and arrays
- This form is sometimes referred to as the *enhanced for* statement, and can be used to make your loops more compact and easy to read
- To demonstrate, consider the following array, which holds the numbers 1 through 10:

```
int[] numbers = {1,2,3,4,5,6,7,8,9,10};
```

Enhanced for Statement when using arrays

- The following program, *EnhancedForDemo*, uses the enhanced for to loop through the array

```
class EnhancedForDemo {
    public static void main(String[] args){
        int[] numbers =
            {1,2,3,4,5,6,7,8,9,10};
        for (int item : numbers) {
            System.out.println("Count is: " + item);
        }
    }
}
```

- In this example, the variable *item* holds the current value from the numbers array

Enhanced for Statement when using arrays

- The output is the same as before.

```
Count is: 1
Count is: 2
Count is: 3
Count is: 4
Count is: 5
Count is: 6
Count is: 7
Count is: 8
Count is: 9
Count is: 10
```

Enhanced for Statement when using arrays

- Old format

```
for (int i = 0; i < numbers.length; i++) {
    System.out.println(numbers[i]);
}
```

- New format

```
for (int item : numbers) {
    System.out.println(item);
}
```

Enhanced for Statement when using arrays

- An enhanced for statement can loop through an array of any type
- An example of a string array:

```
class EnhancedForLoop {
  public static void main(String[] args) {
    String[] languages = { "C", "C++", "Java", "Python", "Ruby"};

    for (String sample: languages) {
      System.out.println(sample);
    }
  }
}
```

Enhanced for Statement when using arrays

When to use Enhanced For Loop

Enhanced for loops may be simple but they are inflexible. You don't have access to a LoopIndex. Thus, you should use an Enhanced For Loop:

- If you have to iterate through all the elements
- If you don't need to know the loop index of the current element
- If you need to iterate through the Collection/Array in first-to-last order.(In other words, if the order doesn't matter to you)

- In all other cases, the "standard" for loop should be preferred