SWEN20003
Object Oriented Software Development

Input and Output - Questions

Bach Le
bach.le@unimelb.edu.au

University of Melbourne

The Road So Far

- Subject Introduction
- A Quick Tour of Java
- Classes and Objects
- Arrays and Strings

Lecture Objectives

Upon completion of this topic you will be able to:

- Accept input to your programs through:
  - Command line arguments
  - User input
  - Files
- Write output from your programs through:
  - Standard output (terminal)
  - Files
- Use files to store and retrieve data during program execution
- Manipulate data in files (i.e. for computation)

Which is the correct syntax to declare a Scanner class object?

1. `Scanner objectName = Scanner();`
2. `Scanner objectName = new Scanner();`
3. `Scanner objectName = Scanner(System.in);`
4. `Scanner objectName = new Scanner(System.in);`
5. `Scanner objectName = System.in;`

Answer:

4. `Scanner objectName = new Scanner(System.in);`

Which of the Scanner class methods consumes (or "eats") the newline character?

1. `next();`
2. `nextLine()`
3. `nextInt();`
4. `nextDouble()`
5. `nextBoolean()`

Answer:

2. `nextLine()`

Consider the following object declaration statement:

```
Scanner objectName= new Scanner(System.in);
```

What is `System.in` in this declaration?

1. A class which points to the input device
2. A reference to the Input stream
3. A reference to Computer System
4. None of the above
5. All of the above

Answer:

2. A reference to the Input stream

Using a Scanner

Name some of the useful methods in the scanner class.

Answer:

```
String l = scanner.nextLine();
String s = scanner.next();
boolean b = scanner.nextBoolean();
int i = scanner.nextInt();
double d = scanner.nextDouble();
float f = scanner.nextFloat();
```

Assess Yourself

```java
import java.util.Scanner;

public class TestScanner2 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter your input: ");
        double d = scanner.nextDouble();
        float f = scanner.nextFloat();
        int i = scanner.nextInt();

        System.out.format("%3.2f, %3.2f, %3d", d, f, i);
    }
}
```

What does the above program print when the following input in entered?

```
Enter your input:
3.2 4.5 8
```

Answer:

What are the drawbacks of the program in the previous slide?

Answer:

Does not tell the user what to enter; the number of inputs, their types etc.

If the user enters an incorrect data type the program crashes

Can you fix these problems?

```java
import java.util.Scanner;

public class TestScanner2Better {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter your input: ");
        System.out.println("First enter a double: ");
        while(!scanner.hasNextDouble()) {
            scanner.nextLine();
            System.out.println("Not a valid input" +    ": enter an double:
        }
        double d = scanner.nextDouble();
        scanner.nextLine();
        System.out.println("Now enter a float: ");
        while(!scanner.hasNextFloat()) {
            scanner.nextLine();
            System.out.println("Not a valid input" +
            ": enter an float: ");
        }
        float f = scanner.nextFloat();
        scanner.nextLine();
        System.out.println("Now enter a integer: ");
        while (!scanner.hasNextInt()) {
            scanner.nextLine();
            System.out.println("Not a valid input" +
            ": enter an integer: ");
        }
```

Assess Yourself

```java
import java.util.Scanner;

public class TestScanner3 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter your input: ");
        double d = scanner.nextDouble();
        String s1 = scanner.nextLine();
        String s2 = scanner.nextLine();

        System.out.format("%3.2f , %s , %s", d, s1, s2);
    }
}
```

Input: 5
6.7
7.2

Output: 5.00, ,6.7

Pitfall: Mixing nextXXX with nextLine

- `nextLine` is the **only** method that "eats" newline characters
- In some cases, you may have to follow `nextXXX` with `nextLine`, if your input is on multiple lines

Other Features

```
scanner.hasNext()
scanner.hasNextXXX()
```

Keyword

.hasNext: Returns `true` if there is *any* input to be read

Keyword

.hasNextXXX: Returns `true` if the next "token" matches *XXX*

Write a program that takes a single line of input, and counts the number of lowercase, uppercase and punctuation characters, and outputs them on the same line, separated by spaces.

Hint: Google how to check character case/category.

Input: "Winter is coming. I am Iron Man!"

Output: "20 4 2"

```java
import java.util.Scanner;
public class CountTypes {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter a string");
        String text = scanner.nextLine();
        int nLower = 0;
        int nUpper = 0;
        int nPunctuation = 0;
        for (int i = 0; i < text.length(); i++) {
            char c = text.charAt(i);
            if (Character.isLowerCase(c)) {
                nLower++;
            } else if (Character.isUpperCase(c)) {
                nUpper++;
            } else if (Character.getType(c) == Character.OTHER_PUNCTUATION) {
                        nPunctuation++;
            }
        }
        System.out.format("%d %d %d", nLower, nUpper, nPunctuation);
    }
}
```

What does the following line of code do?

Answer

```
try (BufferedReader br =
new BufferedReader(new FileReader("test.txt"))) {
```

What do the following lines of code do?

Answer

```
} catch (Exception e) {
    e.printStackTrace();
}
```

Assess Yourself

What does the following program do?

Assess Yourself

```java
import java.io.FileReader;
import java.io.BufferedReader;
import java.io.IOException;

public class ReadCSV {
    public static void main(String[] args) {

        try (BufferedReader br =
            new BufferedReader(new FileReader("recipe.csv"))) {
            String text;
            int count = 0;

            while ((text = br.readLine()) != null) {
                String cells[] = text.split(",");

                String ingredient = cells[0];
                double cost = Double.parseDouble(cells[1]);
                int quantity = Integer.parseInt(cells[2]);

                System.out.format("%d %s will cost $%.2f\n", quantity,
                ingredient, cost*quantity);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
}
```

Reading CSV files

- CSV = *Comma Separated Value*
- Somewhat equivalent to a spreadsheet
- Usually contains a header row to explain columns
- Example:

```
Ingredient,Cost,Quantity
Bananas,9.2,4
Eggs,1,6
```

- **Required knowledge for Projects!**

File Output

```
1   import java.io.FileWriter;
2   import java.io.PrintWriter;
3   import java.io.IOException;
4
5   public class FileWrite1 {
6       public static void main(String[] args) {
7           try (PrintWriter pw =
8               new PrintWriter(new FileWriter("testOut.txt"))) {
9
10              pw.println("Hello World");
11              pw.format("My least favourite device is %s and its price is $%d",
12                  "iPhone", 100000);
13
14          } catch (IOException e) {
15              e.printStackTrace();
16          }
17      }
18  }
```

File Output - Methods

```
pw.print("Hello ");
pw.println("World");
pw.format("My least favourite device is %s and its price is $%d",
    "iPhone", 100000);
```

- `pw.print` - Outputs a String
- `pw.println` - Outputs a String with a new line
- `pw.format` - Outputs a String, and allows for format specifiers

Assess Yourself

What does the following program write to the file?

Assess Yourself

```java
import java.io.PrintWriter;
import java.io.IOException;
import java.util.Random;

public class FileWrite2 {
    public static void main(String[] args) {
        final int MAX_NUM = 10000;
        final int ITERATIONS = 1000000;

        Random rand = new Random();

        try (PrintWriter pw =
            new PrintWriter(new FileWriter("testOut2.txt"))) {

            int nums[] = new int[MAX_NUM];

            for (int i = 0; i < ITERATIONS; i++) {
                nums[rand.nextInt(MAX_NUM)] += 1;
            }
            for (int i = 0; i < nums.length; i++) {
                pw.format("%4d: %4d\n", i, nums[i]);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Assess Yourself

What does the following program write to the file?

Assess Yourself

```java
import java.io.FileWriter;
import java.io.PrintWriter;
import java.io.IOException;
import java.util.Scanner;

public class FileWrite3 {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        try (PrintWriter pw =
            new PrintWriter(new FileWriter("test.html"))) {

            pw.println("<h1>The Chronicles of SWEN20003</h1>");

            while (scanner.hasNext()) {
                String text = scanner.nextLine();

                pw.println("<p>" + text + "</p>");
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Assess Yourself

Implement a rudimentary survey/voting system, by writing a program that continuously expects a single input from the user. This input will be one of three options, in response to the question "Which is your favourite Star Wars trilogy?"

The valid responses are `0` (for the "Original" trilogy), `1` ("New"), and `2` ("The other one").

Once the input has ended, your program should output the results of the survey, one option per line, as below.

Execution:

```
0
0
1
1
1
2
Original Trilogy: 2
New Trilogy: 3
Other Trilogy: 1
```

Assess Yourself

```java
import java.util.Scanner;

public class Survey1 {
    public static void main(String[] args) {

        final int N_OPTIONS = 3;

        final int ORIGINAL = 0;
        final int NEW = 1;
        final int OTHER = 2;

        int results[] = new int[N_OPTIONS];

        Scanner scanner = new Scanner(System.in);

        while (scanner.hasNextInt()) {
            int vote = scanner.nextInt();
            results[vote] += 1;
        }

        System.out.println("Original Trilogy: " + results[ORIGINAL]);
        System.out.println("New Trilogy: " + results[NEW]);
        System.out.println("Other Trilogy: " + results[OTHER]);

    }
}
```

Combining Reading and Writing

```
1   import java.io.FileReader;
2   import java.io.BufferedReader;
3   import java.io.FileWriter;
4   import java.io.PrintWriter;
5
6   public class FileReadWrite {
7       public static void main(String[] args) {
8
9           try (BufferedReader br = new BufferedReader(new FileReader("input.txt"));
10              PrintWriter pw = new PrintWriter(new FileWriter("output.txt"))) {
11
12              String text;
13
14              while ((text = br.readLine()) != null) {
15                  pw.println(text.toLowerCase());
16              }
17          } catch (Exception e) {
18              e.printStackTrace();
19          }
20      }
21  }
```

Application #1: Data Storage/Retrieval

```
1   /** Using files to store intermediate data during computation */
2
3   final int MAX_DATA = 1000;
4
5   try (BufferedReader br = new BufferedReader(new FileReader("input.txt"));
6       PrintWriter pw = new PrintWriter(new FileWriter("output.txt", true))) {
7
8       // Recover data from previous run
9       String oldData[] = loadPreviousData(br);
10
11      String newData[] = new String[MAX_DATA];
12
13      int count = 0;
14
15      while (magicalComputationNeedsDoing()) {
16          newData[count] = magicalComputation(oldData);
17
18          count += 1;
19
20          // Once we do enough computation, store the results just in case
21          if (count == MAX_DATA) {
22              writeData(pr, newData);
23              count = 0;
24          }
25      }
26
27  }
```

Application #2: Data Manipulation

```
1   /** Using Java to parse/manipulate/convert/etc. files */
2
3   try (BufferedReader br = new BufferedReader(new FileReader("input.txt"));
4       PrintWriter pw = new PrintWriter(new FileWriter("output.txt"))) {
5
6       String text;
7
8       while ((text = br.readLine()) != null) {
9           // Manipulate the input file
10          String newText = magicalComputation(text);
11
12          // Write to output file
13          pw.println(newText);
14      }
15
16  }
```

Assess Yourself

1. Write a program that accepts a `filename` from the user, which holds the marks for students in SWEN20003. Your program must then process this data, and output a histogram of the results
2. Extend your program so that it accepts two more inputs for the `min` and `max` values for the data
3. Extend your program so that it accepts one more input for the `width` of each "bin" in the histogram

Assess Yourself

```java
import java.util.Scanner;
import java.io.File;

public class MarkHist {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter filename: ");
        String filename = scanner.nextLine();

        System.out.print("Enter min value: ");
        int min = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter max value: ");
        int max = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter bin width: ");
        int width = scanner.nextInt();
        scanner.nextLine();

        int data[] = new int[max-min + 1];

        int total = 0;

        ...

    }
}
```

Assess Yourself

```
1   try (Scanner file = new Scanner(new File(filename))) {
2       // Skip the first line
3       file.nextLine();
4
5       while (file.hasNext()) {
6           String line[] = file.nextLine().split(",");
7
8           int d = Integer.parseInt(line[1]);
9
10          data[d - min] += 1;
11          total += 1;
12      }
13
14      ...
15
16  } catch (Exception e) {
17      e.printStackTrace();
18  }
```

Assess Yourself

```
1    // Print out graph
2    for (int i = 0; i < data.length; i += width) {
3        int sum = 0;
4
5        // Bundle into *width* sized blocks
6        for (int j = 0; j < width && i + j < data.length; j++) {
7            sum += data[i+j];
8        }
9
10       int percentage = (int) (100 * (1.0 * sum)/total);
11       String bar = "";
12
13       if (percentage > 0) {
14           bar = String.format("%" + percentage + "s", " ")
15           .replace(" ", "=");
16       }
17
18       int lower = i + min;
19       int upper = lower + width - 1;
20
21       // Print the block
22       System.out.format("%03d-%03d: %s\n", lower, upper, bar);
23   }
```

Assess Yourself

Write a program that takes three inputs from the user:

- `String`, a unit of measurement
- `int`, the number of units
- `String`, an ingredient in a recipe

Your code should write in the following format to a file called `"recipe.txt"`:
`"- Add 300 grams of chicken"`

Bonus Task:
Open the file in "append" mode; this means the file will be added to, rather than overwritten, each time you run your code.

Assess Yourself

Write a program that accepts a `filename` from the user, and then processes that file, recording the frequency with which **words** of different lengths appear.

Assess Yourself

Write a program that accepts a HTML `filename` from the user, and then takes continuous user input and writes it to the file; essentially a Java based HTML writer.

Bonus #1: add validation to detect valid HTML tags (`<p>`, `<h1>`, etc.).

Bonus #2: add "shortcuts"; for example, entering `{text}` might make *text* automatically bold.

Lecture Objectives

Upon completion of this topic you will be able to:

- Accept input to your programs through:
  - Command line arguments
  - User input
  - Files
- Write output from your programs through:
  - Standard output (terminal)
  - Files
- Use files to store and retrieve data during program execution
- Manipulate data in files (i.e. for computation)