# CS1101 Programming and Problem Solving

Dr. Gina Bai

Spring 2023

# Updated Schedule

- Wednesday, March 8
  - **Review of Exam 2**
- Friday, March 10
  - No Class
- Monday, March 20
  - Guest Lecture on Arrays
    - Don't know which section yet
- Wednesday, March 22
  - Exam 2
- Friday, March 24
  - No Lecture
    - Instructor's out of town for an AP CSA meeting
  - "Office Hours" in Stevenson 5326
    - **4 TAs for 10:10am – 11:00am**
    - **3 TAs for 11:15am – 12:05pm**

# Logistics

- **ZY-5B** on zyBook > Assignments
  - Due: **Wednesday, March 8**, at 11:59pm
- **PA07 – W, A, B** on zyBook > Chap 11
  - Due: **Thursday, March 9**, at 11:59pm
- **ZY-6** on zyBook > Assignments
  - Due: **Monday, March 20**, at 11:59pm
- **PA08 – A, B** on zyBook > Chap 11
  - Due: **Saturday, March 25**, at 11:59pm

**NO Office Hours during Spring Break**

# More File Input

## Validating the Input File Name

zyBook Chap 6.4

# Recap – `throws` Clause

A `throws` clause is a declaration that a method **may exit unexpectedly** due to a **particular type of exception**.

- It is an explicit acknowledgment that a statement in the method may throw the exception.
- A waiver of liability: "I hereby agree that this method might throw an exception, and I accept the consequences (crashing) if this happens."

Can we prevent the program from crashing due to FileNotFoundException?

# Validating the Input File Name

```java
// Prompt the user for a file name
System.out.print("Enter file name: ");
String fileName = console.nextLine().trim();

// Construct the File object given the file name
File inputFile = new File(fileName);

// while the input file doesn't exist given the file name
while (!inputFile.exists()) {

    // Reprompt the user for a file name
    System.out.print("File does not exist, try again: ");
    fileName = console.nextLine().trim();

    // Reconstruct the File object with the new file name
    inputFile = new File(fileName);
}

// ASSERT: the file exists
Scanner fileScnr = new Scanner(inputFile);
```

# Midterm Exam 2

# Midterm Exam 2 – Learning Objectives

- All materials correspond to zyBook **Chapters 4 – 6**
- The concepts in these chapters **build on earlier concepts**, hence the exam will be **cumulative**, but with a focus on the chapters above

# Midterm Exam 2 – Format

- **Paper-based**, closed book, closed notes
  - Tip: practice writing code on paper
- A combination of short answer, multiple choice, true/false, code reading and writing **(45~55%)**
- Regular class time (50 minutes)
  - Arrive early!!!
- Includes a *reference guide*

# Midterm Exam 2 – Programming Style

- You *do not* need to comment your code
- You *do not* need to keep track of line length
- You *do not* need to use meaningful names for identifiers
- You *NEED* to use **proper indentation**
- Your answer must be ***LEGIBLE*** (exams are scanned, and then graded)

# Midterm Exam 2 – Practice Exam

Practice Exam

- **Brightspace > Content > Course Documents**
- Solutions will be posted
- *Will not* demonstrate *all* the kinds of problems you can see
- Not a full view of the exam, *just a snapshot*

# Midterm Exam 2 – Preparation Strategies

- Review the learning objectives
- Review the lecture slides, including the in-class activities
- Review zyBook, and the activities
- Review (and possibly rewrite) the lab exercises
- If you do not fully understand a topic, read the related textbook section
- Attend office hours to ask additional questions/clarifications
- Complete the Practice Exam

# Learning Objectives – Chap 4

- Decision statement structure – if statements
- Relational and logical operators
- The `boolean` type - boolean expressions, boolean methods
- String methods
  - `length, equals, equalsIgnoreCase, indexOf, charAt, substring, contains, replace, toLowerCase, trim…`
- The `char` type and the `Character` wrapper class
  - `isDigit, isLetter, isLowerCase, isUpperCase, toLowerCase, toUpperCase…`

# Learning Objectives – Chap 5

- `while` loops
- `do-while` loops
- `for` loops
- Nested loops
- Fencepost problems, Sentinel values (while loop)

# Learning Objectives – Chap 5

- Variable scope
- `Scanner` object methods
  - `nextInt, nextDouble, next, nextLine`
- The `Random` object and generating random numbers
- Assertions

# Learning Objectives – Chap 6

- Output formatting using `printf`
  - Details in **Lecture 6**
- File Input
  - `File` object
  - `throws` clause
  - `Scanner` object methods
    - `hasNextInt, hasNextDouble, hasNext, hasNextLine`

## Q1: True/False

a. The while loop will always execute at least once.

   **False**

b. The do-while loop will always execute at least once.

   **True**

c. The for loop is sometimes called a fixed repetition loop because the loop is repeated a fixed number of times.

   **True**

Q2: What is the output when the following code is executed?

```
for (int i = -2; i <= 2; i += 2) {
    for (int j = i; j < i + 3; j++) {
        System.out.print( j + " " );
    }
}
```

A. -2 -1 0 1 2

B. 0 1 2 0 1 2 0 1 2

C. -2 0 2

D. -2 -1 0 0 1 2 2 3 4

E. -2 -1 0

Q3: Which of the following code segments produces the output when executed? Select ALL.

6 5 4 3 2 1

5 4 3 2 1

4 3 2 1

3 2 1

2 1

1

A

```
for ( int i = 0; i < 6 ; i++ ) {
    for ( int j = 6 - i; j > 0; j-- ) {
        System.out.print( j + " " );
    }
    System.out.println();
}
```

B

```
for ( int i = 0; i < 6 ; i++ ) {
    for ( int j = 6 - i; j >= 1; j-- ) {
        System.out.print( j + " " );
    }
    System.out.println();
}
```

C

```
for ( int i = 6; i > 0 ; i-- ) {
    for ( int j = i; j > 0; j-- ) {
        System.out.print( j + " " );
    }
    System.out.println();
}
```

Q4: Given the following method

```
public static void mystery(int x) {
    int y = 1;
    int z = 0;
    while (2 * y <= x) {
        y = y * 2;
        ++z;
    }
    System.out.println(y + " " + z);
}
```

Write the output of each of the following method calls.

- `mystery(1)` **1 0**
- `mystery(6)` **4 2**
- `mystery(19)` **16 4**
- `mystery(39)` **32 5**
- `mystery(74)` **64 6**

Q5: Write an expression that generates a random integer between 0 and 10 inclusive.

```
Random rand = new Random();
int n = rand.nextInt(11);
```

Q6: What is the range of the result of integers a, b, c, and d? Be specific with inclusive vs. exclusive.

```
Random rand = new Random();
int a = rand.nextInt(100);
int b = rand.nextInt(20) + 50;
int c = rand.nextInt(20 + 50);
int d = rand.nextInt(100) - 20;
```

**[0, 99]**

**[50, 69]**

**[0, 69]**

**[-20, 79]**

Q7: Given the double **1234567.89**, what format specifier would you use in a printf statement to print it as **1,234,567.9**?

`%,.1f`

Q8: What is the output when the following code is executed?

```
String s1 = "Go Commodore!";
String s2 = "Anchor Down!";
char c = 'a'; //ASCII (integer) value is 97
System.out.println(s2.substring(7) + s1.charAt(9) + c + 1);
```

**Down!oa1**

Q9: In the following input file,

```
Hello there,how are you?
I am "very well", thank you.
12 34 5.67 (8 + 9) "10"
```

- How many tokens could be read by a Scanner? **17**
- What are they?
- # of tokens that can be read as an integer **2**
- # of tokens that can be read as a double **3**
- # of tokens that can be read as a String **17**

Q10: For each point in the code, choose (**A**)lways true, (**N**)ever true, or (**S**)ometimes true

| | `n > b` | `a > 1` | `b > a` |
|---|---|---|---|
| Point A | **S** | **S** | **S** |
| Point B | **A** | **S** | **S** |
| Point C | **S** | **A** | **A** |
| Point D | **S** | **A** | **N** |
| Point E | **N** | **S** | **S** |

```java
public static int mystery (int n) {
    Random r = new Random();
    int a = r.nextInt(3) + 1;
    int b = 2;

    // Point A
    while (n > b) {
        // Point B
        b = b + a;

        if (a > 1) {
            --n;

            // Point C
            a = r.nextInt(b) + 1;
        } else {
            a = b + 1;
            // Point D
        }
    }

    // Point E
    return n;
}
```

Q11: Reproduce the following matrix with nested for loops

j →

i ↓

| | | | | |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| 2 | 4 | 6 | 8 | 10 |
| 3 | 6 | 9 | 12 | 15 |
| 4 | 8 | 12 | 16 | 20 |

Hint: Represent each value with i and j

| | **j = 1** | **j = 2** | **j = 3** | **j = 4** | **j = 5** |
|---|---|---|---|---|---|
| **i = 1** | | | | | |
| **i = 2** | | | | | |
| **i = 3** | | | | | |
| **i = 4** | | | | | |

# Sample Solution

```
1    2    3    4    5
2    4    6    8    10
3    6    9    12   15
4    8    12   16   20
```

```java
for (int i = 1; i <= 4; ++i) {
    for (int j = 1; j <= 5; ++j) {
        System.out.print((i * j) + "\t");
    }
    System.out.println();
}
```

Try to rewrite the code to make the matrix resizable (e.g., having a method with the size as parameter)