Summary

1 Disclaimer

2 Variables
   2.1 Primitive Types
   2.2 Strings
   2.3 Suggestions, Warnings, and Resources
   2.4 FAQs

3 Printing
   3.1 Suggestions, Warnings, and Resources
   3.2 FAQs

4 Conditional Statements
   4.1 if-else if-else Statements
   4.2 Difference between if-if-if and if-else if-else
   4.3 Switch Statements
   4.4 Suggestions, Warnings, and Resources

5 Other Classes
   5.1 Integer
   5.2 Double
   5.3 Character
   5.4 Math

6 Scanners
   6.1 Suggestions, Warnings, and Resources
   6.2 FAQs

7 Practice Written Exam
   7.1 Short Answer
   7.2 Tracing Code

8 Programming Quiz Practice Exam

9 Suggestions for Studying and Test Taking
   9.1 Written
   9.2 Programming Quiz
   9.3 Common Errors
   9.4 Challenges
1 Disclaimer

This is a review of the material so far, but there may be material on the exam not covered in this study guide.

2 Variables

2.1 Primitive Types

The eight primitive types are: int, double, char, float, long, short, byte, and boolean. So far you’ve learned about two reference types, a String and a Scanner. These are NOT a primitive types, you will learn why later in the semester.

Below are a few examples of declaring a variable. Format: type var_name;

```java
int i;
double d;
char c;
String s;
```

Below are a few examples of assigning a variable. Format: var_name = value;

```java
i = 3;
d = 4.5;
c = 'a';
s = "Hey all!";
```

Below are a few examples of initializing a variable. Initializing is a combination of declaring and assigning. Format: type var_name = value;

```java
boolean b = true;
float f = 3.1415F;
long l = 32L;
String response = "Yes";
```

Remember: Every variable must always have a declaration (assignment can come later).

Below are a few examples of typecasting.

```java
int i = 3;
double d = (double) i;
char c = 'c';
int ascii_c = (int) c;
```

Remember: If you typecast you can lose information, for example if I typecast 6.8 to an integer I will get back 6.

2.2 Strings

Common String methods:

- length (ex. s.length();)
- charAt (ex. s.charAt(0);)
- indexOf (ex. s.indexOf('a');)
- substring (ex. s.substring(0, 3);)
- equals (ex. s.equals(s1);)
- toUpperCase (ex. s.toUpperCase();)
- toLowerCase (ex. s.toLowerCase();)

Below are a few examples of how to use the above methods.
```java
public class Review {
    public static void main ( String [] args ) {
        // Declaration
        String s;
        // Assigning
        s = "Hi there";
        // Declaring & Assigning
        String myString = "Okay then";
        String myString1 = new String ("Okay fine then");
        // Common Methods
        String s = "SmIlEy mOnKEy";
        char c0 = s.charAt(0); // assigns S to variable c
        int i0 = s.indexOf( 'z' ); // assigns -1 to variable i0
        // because there is no 'z' in s
        String sub = s.substring(0); // assigns s to sub
        // returns
        // (stringName.substring(#)
        // the character from #
        // to the end of the string
        String sl = s.substring(0,1); // only returns S because substring takes an
        // inclusive index as the first parameter and
        // an exclusive index for the second.
        // Meaning, this would only read the first letter.
        System.out.println(s.toUpperCase()); // does not change s!
        String newString = s.toLowerCase(); // does not change s!
        System.out.println(s.equals(newString)); // prints false
        boolean b = s.equals(sub); // assigns true to b
        int length = sub.length(); // assigns 13 to length
        // (1 based not 0 based
        // (like indexes)
    }
}
```

### 2.3 Suggestions, Warnings, and Resources

- **Warning:** Make sure you use the correct ticks/quotes for variables. For example, to assigninitialize a char you must use ‘ ’ (ticks or single quotes), to assigninitialize a String you must use “ ” (double quotes), and numbers don’t use ticks or quotes.

- **Warning:** Always use String’s .equals method (compared to ==).

- **Warning:** Indexing for Strings always starts at 0 (for example the first character is the same as the 0th character).

- **Warning:** String methods do NOT change the original string!

- **Resource:** Use Oracle’s documentation for more methods and examples

- **Resource:** Oracle’s Learning Paths

### 2.4 FAQs

1. **Q:** Why is there an ‘F’ or ‘L’ at the end of a long or float value? Is it case sensitive?
   
   **A:** If you leave off the ‘F’ when assigning or initializing a float, Java will assume you want a double (instead of a float). A float can hold 32-bit number and a double can hold a 64-bit number. You only need to add an ‘L’ when assigninginitializing a long when the value won’t fit into an int. An int can hold numbers between $2^{-31}$ and $2^{31} - 1$ and a long can hold numbers between $2^{-63}$ and $2^{63} - 1$. The ‘F’ and the ‘L’ are not case sensitive, but be careful a lower case ‘l’ looks an awful like the number 1.
There are three ways to print to the console: `System.out.print()`, `System.out.println()`, and `System.out.printf()`. Below are examples of how and when you could use each.

```java
// Example code
int num1 = 3;
int num2 = 5;
int num3 = 9;
double avg = (num1 + num2 + num3) / 3;

// The following three print statements are equivalent
System.out.print("Average " + num1 + ", " + num2 + ", and " + num3 + ": " + avg + 
"n");
System.out.println("Average " + num1 + ", " + num2 + ", and " + num3 + ": " + avg);
System.out.printf("Average %d, %d, and %d: %.1f
n", num1, num2, num3, avg);
```

This outputs the following:

```
Average 3, 5, and 9: 5.0
Average 3, 5, and 9: 5.0
Average 3, 5, and 9: 5.0
```

```
2
0002
3.141593
3.142
```

- `System.out.print();` is the same as `System.out.println();` except it doesn’t have a new line character at the end of the line.
- `System.out.printf();` uses format specifiers as placeholders. A format specifier begins with a percent sign and ends with a converter.

```java
%d - int
%f - float
%.#f - (# is the number of decimal places) float
%c - char
%s - String
```

### 3.1 Suggestions, Warnings, and Resources

- Resource: [Here is a good printf resource](https://example.com/resource1)
- Resource: [Oracle’s Formatter Documentation](https://docs.oracle.com/javase/8/docs/api/java/util/Formatter.html)
- Note: If you don’t specify how many decimals to print when using printf, Java defaults to printing six decimal digits.
- Warning: Printf rounds!

### 3.2 FAQs

1. **Q:** Can you use any of the three ways to print? Is there a specific time you’d use one over the other?  
   **A:** You can use any of the three ways to print at any time, one might be easier in some situations (i.e. if you have a lot of variables to print, printf might be easier).

2. **Q:** Is it necessary to use printf or String.format if you need to specify decimal places?  
   **A:** Yes.
4 Conditional Statements

4.1 if-else if-else Statements

```java
public class Conditionals {
    public static void main(String[] args) {
        int i = 0;
        // if statement
        if (i == 0) {
            System.out.println("i == 0");
        }
        boolean b = true;
        // I don’t need brackets because I only have one line of code
        // under each statement. However, you can always have brackets
        // like the example above.
        if (b == false) { // the example above.
            System.out.println("b is FALSE");
        } else {
            System.out.println("b is TRUE");
        }
        int i1 = 32;
        // here I do need brackets because there is more than one
        // line of code nested in my if statement.
        if (i1 >= 15) {
            System.out.println("i1 is greater than or equal to 15");
            i1++;
        } else if (i < 3) {
            System.out.println("i is less than 3");
        } else if (i > 100) {
            System.out.println("i is greater than or equal to 15");
        } else {
            System.out.println("i must be between 3 and 14");
        }
        // sometimes you don’t need an else
        String name = "Waldo";
        // if I have found Waldo I want to say I found him but...
        if (name.equals("Waldo")) {
            System.out.println("Found him!");
        // if I didn’t find him I don’t want to do anything
        }
    }
}
```

The output from the above code is:
- i == 0
- b is TRUE
- i1 is greater than or equal to 15
- Found him!

4.2 Difference between if-if-if and if-else if-else

If you use an if-else if-else statement and multiple conditionals are true, the program executes the first true conditional and then exits.

If you use an if-if-if statement and multiple conditionals are true, the program will execute all of the true conditionals.

```java
public class Conditionals {
    public static void main(String[] args) {
        int i = 128;
        System.out.println("if-else if-else statement");
        if (i == 128)
            System.out.println("i == 128");
        else if (i > 100)
            System.out.println("i > 100");
    }
}
```
else if (i < 130)
    System.out.println("i < 130");

System.out.println("if-if-if statement");

if (i == 128)
    System.out.println("i == 128");
if (i > 100)
    System.out.println("i > 100");
if (i < 130)
    System.out.println("I < 130");
}
}

The output from the above code is:

if-else if-else statement
i == 128
if-if-if statement
i == 128
i > 100
I < 130

4.3 Switch Statements

public class Conditionals {
    public static void main (String [] args) {
        int i = 32;
        switch (i) {
            case 10: System.out.println("i = 10"); break;
            case 32: System.out.println("i = 32"); break;
            case 101: System.out.println("i = 101"); break;
            default: System.out.println("i is not 10, 32, or 101");
        }
        // beware of your breaks!
        char c = 'b';
        switch (c) {
            case 'a': System.out.println("c = a");
            case 'b': System.out.println("c = b");
            case 'c': System.out.println("c = c");
            default: System.out.println("c is not a, b, or c");
        }
        // times you might want to not have breaks
        int dogs = 0, cats = 0;
        String breed = "German Shepard";
        // spacing doesn't matter
        // you aren't required to have a default
        // like you aren't required to have an else
        // just be aware that there are possible cases
        // that aren't being handled.
        switch (breed) {
            case "Boxer":
            case "Great Dane":
            case "German Shepard":
            case "Lab":
                dogs++; break;
            case "Calico": case "British Shorthair": case "Siamese":
                cats++; break;
        }
        System.out.printf("Dogs: %d, Cats %d\n", dogs, cats);
    }
}
The output from the above code is:
```java
i = 32
c = b
c = c
c is not a, b, or c
Dogs: 1, Cats 0
```

4.4 Suggestions, Warnings, and Resources

- Warning: Don’t add a semicolon after your conditional. For example if (32 == 32);
- Warning: Be aware of when you should and shouldn’t have breaks in your switch statement.

5 Other Classes

5.1 Integer

- **parseInt** (ex. `Integer.parseInt("3");`
- **MAX_VALUE** (constant) (ex. `int max = Integer.MAX_VALUE;`
- **MIN_VALUE** (constant) (ex. `int min = Integer.MIN_VALUE;`

5.2 Double

- **parseDouble** (ex. `Double.parseDouble("3.3");`
- **MAX_VALUE** (constant) (ex. `double max = Double.MAX_VALUE;`
- **MIN_VALUE** (constant) (ex. `double min = Double.MIN_VALUE;`

5.3 Character

- **isUpperCase** (ex. `Character.isUpperCase('a');`
- **isLowerCase** (ex. `Character.isLowerCase('a');`
- **isLetter** (ex. `Character.isLetter('4');`
- **isDigit** (ex. `Character.isDigit('4');`

5.4 Math

- **sqrt** (ex. `Math.sqrt(64);`
- **pow** (ex. `Math.pow(1.0, 3.0);`
- **sin** (ex. `Math.sin(1.5);`
- **cos** (ex. `Math.cos(1.5);`
- **exp** (ex. `Math.exp(3.3);`
- **log** (ex. `Math.log(2.1);`)
• min (ex. Math.min(1, 3);
• max (ex. Math.max(1, 3);
• round (ex. Math.round(3.3))
• floor (ex. Math.floor(3.3);
• ceil (ex. Math.ceil(3.3))
• PI (constant) (ex. double num = Math.PI;)

6 Scanners

Below is an example of reading from the keyboard/console.

```java
// importing necessary methods
import java.util.Scanner;

public class ScannerPractice {  
  public static void main (String [] args) {
    // Initializes a Scanner object called reader
    Scanner reader = new Scanner(System.in);
    // Prompting user for an integer
    System.out.print("Enter your favorite integer: ");
    // Read and store an int
    int favorite_number = reader.nextInt();
    // Prompting user for a decimal
    System.out.print("Enter your favorite decimal number: ");
    // Read and store a double
    double favorite_decimal = reader.nextDouble();
    // Prompting user for a character
    System.out.print("Enter your favorite character: ");
    // Read and store a letter
    char favorite_character = reader.next().charAt(0);

    // printing summary
    System.out.printf("Your favorite integer is %d\nYour favorite decimal number to 6 decimals
    is %f\nYour favorite character is %c\n", favorite_number,
    favorite_decimal, favorite_character);
    reader.close();

    // NOTE: You can also use Java 8’s try–catch resource. This guarantees your Scanner is
    // cleaned up (Scanner closing, etc.). Example below:
    try (Scanner scan = new Scanner(new File(filename))){
      // your code here
    }
  }
}
```

On the console, it looks like this:

Enter your favorite integer: 11
Enter your favorite decimal number: 3.14
Enter your favorite character: a
Your favorite integer is 11
Your favorite decimal number to 6 decimals is 3.140000
Your favorite character is a

6.1 Suggestions, Warnings, and Resources

• Suggestion: Get in the habit of closing your Scanners
• Warning: When you go from line based processing (.nextLine()) to token based processing (.next(), .nextDouble(), .nextInt()) you MUST discard the new line character before you continue parsing.
• Resource: Use Oracle’s documentation for more methods and examples
6.2 FAQs

1. **Q**: My code still works even though I didn’t close my Scanner object, why do you guys teach so you have to close it?

   **A**: We teach it so you close your Scanners so you get in the habit of closing objects (better coding practice and when we get to PrintWriters you must close your object).
7 Practice Written Exam

7.1 Short Answer

1. Declare an integer named \texttt{i0}.

2. Initialize a float named \texttt{f} to 4.0.

3. Initialize a Scanner to read from the keyboard, called \texttt{read}.

4. Assign \texttt{i0} to an integer entered from the user.

5. Close the Scanner.

6. Change the pre-assigned String variable called \texttt{s} to be all upper case.

7. Name the eight primitive types.

8. Question 8 might be better written as: Given the following code, write what would be printed to the console (write error if the code throws an error):

   ```java
   String s = new String("Bob Marley");
   String m = "mummies";
   System.out.println(m.substring(0,3) + s.substring(4));
   System.out.println(m.charAt(1) + "−" + s.charAt(1));
   System.out.println(s.indexOf('q') + m.indexOf(3));
   System.out.println(s.length());
   System.out.println(s + m);
   System.out.println(m.indexOf("Ma"));
   System.out.println(s.length() - m.length());
   System.out.println(s.substring(0,11));
   System.out.println(s.length() / m.length());
   ```

9. What is 18 \% 3?

10. Write a switch statement based off of the character variable \texttt{c}. If \texttt{c} is 'a' print “hi”, if \texttt{c} is 'b' print “okay”, if \texttt{c} is 'c' print “then”, if \texttt{c} is 'd' print “bye", if it is not 'a', 'b', 'c', or 'd' print “fine then”.

11. Write an if statement that does the same thing as question 10.

12. Declare a Scanner called \texttt{scan}. Read in one word and store it into the predefined String variable called \texttt{word}. Print out the second through fourth character of \texttt{word}. Close your scanner.

13. Check to see if the predefined String variables \texttt{s0} and \texttt{s1} are equal. Print the result.

14. Check to see if the predefined int variables \texttt{i0} and \texttt{i1} are equal. Print the result.
7.2 Tracing Code

For each question below, write what is printed.

```java
import java.util.Scanner;

public class Tracing {
    public static void main(String[] args) {
        int i = 3;
        double d = (double) i;
        // Question 1
        System.out.printf("%d %f\n", i, d);
        Scanner keys = new Scanner(System.in);
        d = keys.nextInt();
        i = (int) keys.nextDouble();
        // Question 2 (assume the user enters 8 and 98.5 respectively)
        System.out.printf("%d %.4f\n", i, d);
        String result = "initial result";
        // Question 3
        if (d > 98)
            result = "greater than 98";
        else
            result = "less than or equal to 98";
        System.out.println(result);
        char grade = keys.next().charAt(0);
        // Question 4 (assume the user enters 'a' (without the quotes))
        switch (grade) {
            case 'a': System.out.println("Passing!");
            case 'b': System.out.println("Passing!");
            case 'c': System.out.println("Passing!");
            case 'd': System.out.println("Kinda sorta passing")
            case 'f': System.out.println("Not really passing")
            default: System.out.println("Not a grade letter recognized");
        }
    }
}
```
8  Programming Quiz Practice Exam

1. Declare a class with a main method
2. Import the Scanner class
3. Declare four String variables called name, city, state, school
4. Declare an int variable called age
5. Declare a double variable called gpa
6. Declare a Scanner that reads from the keyboard called reader
7. Prompt the user to enter their name
8. Using the Scanner read in their name and assign the value to name
9. Prompt the user to enter their age
10. Using the Scanner read in their age and assign the value to age
11. Prompt the user to enter their GPA
12. Using the Scanner read in their GPA and assign the value to gpa
13. Prompt the user to enter their school
14. Using the Scanner read in their school’s name and assign the value to school
15. Prompt the user to enter their city
16. Using the Scanner read in their city’s name and assign the value to city
17. Prompt the user to enter their state
18. Using the Scanner read in their state’s name and assign the value to state.
19. Print a summary of the entered information in the format by printing:
   (a) Print their first name (ONLY the first name)
   (b) Print their age
   (c) Print their GPA with two decimal point accuracy
   (d) Print their school name
   (e) Print their city then their state

An example run would look like:

Enter your name: Bob Stevey Joe
Enter your age: 18
Enter your GPA: 3.13897984251
Enter your school: Colorado State University
Enter your city: Fort Collins
Enter your state: Colorado

Summary:
First name: Bob
Age: 18
GPA: 3.14
School Name: Colorado State University
Fort Collins, Colorado

20. Close reader
9 Suggestions for Studying and Test Taking

9.1 Written

When reading through code and writing the output: Write your variables on the side and as your variables change, you change your list on the side.

Practice writing code in Eclipse and before you run your program guess what the output would be. This is good practice for testing your own programs and also for the code tracing part of the exam.

If you need more tracing examples (or more coding examples in general), there is a “Programs” tab on the CS163/164 homepage. There are also examples on the Progress page.

Write the variables, their values, and any changes made to those values to keep track of what is changing throughout the program.

9.2 Programming Quiz

Go back to past recitations and assignments and redo them without using the Internet, friends, or past recitations/assignments.

Practice writing code in Eclipse. Make up projects and problems or ask a TA and they can give you some challenges.

Look at code, the more exposure you get to code (whether it’s your own code or not) the easier it is to understand. Some sample code is under the “Programs” tab and the Progress Page.

9.3 Common Errors

- Scanner token processing(.next, .nextInt(), .nextDouble()) to line processing (.nextLine()). Need an extra .nextLine() when going from token processing to line processing.
- Use .equals when comparing Strings, NOT == !
- Incorrect brackets around conditional statements

9.4 Challenges

[CodingBat](https://codingbat.com) and [Hackerrank](https://www.hackerrank.com) offer good extra coding practice.
ANSWERS ON THE NEXT PAGE
10 Answers to Practice Written and Programming Problems

10.1 Written

10.1.1 Short Answer

1. int i0;
2. float f = 4.0F;
3. Scanner read = new Scanner(System.in);
4. i0 = read.nextInt();
5. read.close();
6. s = s.toUpperCase();
7. int, double, char, float, long, boolean, byte
8. mumMarley
   u-o-2
   10
   Bob Marley mummies
   -1
   3
   error
   1
9. 0
10. switch (c) {
    case 'a': System.out.println("hi"); break;
    case 'b': System.out.println("okay"); break;
    case 'c': System.out.println("then"); break;
    case 'd': System.out.println("bye"); break;
    default: System.out.println("fine then"); break;
   }
11. if (c == 'a')
    System.out.println("hi");
    else if (c == 'b')
    System.out.println("okay");
    else if (c == 'c')
    System.out.println("then");
    else if (c == 'd')
    System.out.println("bye");
    else
    System.out.println("fine then");
12. Scanner scan = new Scanner(System.in);
    word = scan.next();
    System.out.println(word.substring(1, 4));
    scan.close();
13. System.out.println(s0.equals(s1));
14. System.out.println(i0 == i1);
10.1.2 Tracing

1. 3 3.000000
2. 98 8.0000
3. less than or equal to 98
4. Passing!
   Passing!
   Passing!
   Kinda sorta passing
   Not really passing
   Not a grade letter recognized

10.2 Programming

NOTE: Your solution may be different from the code below, as long as your output matches my output (in all cases), you are answer is fine.

```java
import java.util.Scanner;

public class PracticeProgramming {
    public static void main(String[] args) {
        Scanner reader = new Scanner(System.in);
        String name = "";
        int age;
        double gpa;
        String city = "", state = "", school = "";

        System.out.print("Enter your name: ");
        name = reader.nextLine();
        // Checks if there is a space (i.e. between first and last name)
        int space_index = name.indexOf(' ');
        // there is a space
        if (space_index != -1)
            name = name.substring(0, space_index);

        System.out.print("Enter your age: ");
        age = reader.nextInt();
        System.out.print("Enter your GPA: ");
        gpa = reader.nextDouble();
        reader.nextLine();
        System.out.print("Enter your school: ");
        school = reader.nextLine();
        System.out.print("Enter your city: ");
        // Could be multiple words (i.e. Colorado Springs)
        city = reader.nextLine();
        System.out.print("Enter your state: ");
        state = reader.nextLine();

        System.out.println("Summary:");
        System.out.println("First name: " + name);
        System.out.println("Age: " + age);
        System.out.printf("GPA: %.2f\n", gpa);
        System.out.println("School Name: " + school);
        System.out.println(city + "", " + state);
        reader.close();
    }
}
```