Review for Exam 1

1. **Algorithms**

   Algorithms are step by step instructions to solve a problem. They are expressed using the following key programming fundamentals:

   a. **Sequencing:** All instructions are carried out in the sequence they are written.

   ```
   do this
   then do this
   and then do this
   etc.
   ```

   b. **Selection:** Allows one to choose between a set of instructions depending on a statement.

   ```
   if something then
       do this
   otherwise
       do that
   ```

   c. **Repetition:** Allows one to repeat a set of statements. There are two kinds:

   ```
   1) repeat N times
       do this
   ```

   ```
   2) while something repeat the following
       do this
   ```

   d. **Abstraction:** Allows us to express computations abstractly:

   ```
   Let n ← $\sqrt{a}$
   ```

   Above, we abstract the algorithm to compute $\sqrt{a}$ into a single operation.

   e. **Variables:** Hold values that we compute on.

   ```
   Let $n$ ← $\frac{3}{4}n + c$
   ```

2. **Java Program Structure:** Every Java program has the following structure:

```
public class Name {
          public static void main(String[] args) {
                    statements
          } // main()
} // class Name
```

3. **Statements:** These are Java commands that implement variable assignment, sequencing, selection, repetition, library functions, etc.

   1. Variable declaration:

   type variable-name;

   e.g.

   ```
   int x, y, z;
   int a = 0;
   ```

   2. Assignment Statement: Sets the value of a variable to that of an expression.

   ```
   n = 3.0/4.0 * n + c;
   ```

   3. Data Types: In Java there are:

   `int` – positive or negative integer values (e.g, -89, 0, 897, etc.)
   `double` – there are floating point/real numbers (e.g. -2.34, 3.1415, etc).
   `boolean` – true, false
   `String` – "Hello there", "The number is:", etc.
   `char` – individual letters (e.g. 'A', '5', etc.)

   4. Operations: Each data type provides some operations:

   `int`: **+** (addition), **-** (subtraction, negation), ***** (multiply), **/** (divide), **%** (remainder)
   `double`: **+** (addition), **-** (subtraction, negation), ***** (multiply), **/** (divide)
   `String`: **+** (concatenation)
   `boolean`: **&&** (and, conjunction), **||** (or, disjunction), **!** (not, logical negation)

   5. Comparison Operations: Allow us to compare int or double values.

< (less than), **<=** (less than or equal to), **>** (greater than), **>=** (greater than or equal to)
== (equal to), **!=** (not equal to)

6. Command-Line Input Arguments: These enable inputs from the command line.

e.g.

```
$ java-introcs IsPrime 31
```

Above, 31 becomes the value of `args[0]`.

7. Output: These commands, from the **`System.out`** (or **`StdOut`**) library provide a meand to print out the results of our computations.

E.g.

```
System.out.println("Hello, world!");
```

- **`print(…)`**
- **`println(…)`**
- **`printf("…", …)`**

8. Data Type Conversions: To convert from one type to another. This can be:

Implicit: `int` to `double`, or `int`, `double`, `boolean` to `String`
Explicit:

e.g.

```
int n = (int) (Math.random() * 10) + );
```

9. Selection/Conditionals: Allow us to select one set of statements (pr another).

```
if ( condition ) {                if ( condition ) {
     do this                           do this
}                                 }
                                  else {
                                     do that
                                  }
```

10. **Repetition:** Allows us to repeat a set of statements.

```
while ( condition ) { {
     do this
}

for ( initialization ; condition ; update )
     do this
}
```

4. **Java Libraries:** These are collections of useful functions that we can use>

   ```
   System.out
   Std.Out
   Std.In
   Integer
   Double
   Math
   ```
   etc.

   Each library provides several useful futncions. See your text or notes for examples of things we have used.

5. **Java Compilation:** To compile a Java program, we use the command:

   ```
   $ javac-introcs ProgramName.java
   ```

   To run the program we use:

   ```
   $ java-introcs ProgramName …
   ```

6. **Linux Commands:** See Chapters 1-4 from Schott for an overview.