Name (print): ________________________________

• **INSTRUCTIONS:**

  – Keep your eyes on your own paper and do your best to prevent anyone else from seeing your work.
  – Do NOT communicate with anyone other than the professor/proctor for ANY reason in ANY language in ANY manner.
  – This exam is closed notes, closed books, and no calculator.
  – Turn all mobile devices off and put them away now. You cannot have them on your desk.
  – Write neatly and clearly indicate your answers. What I cannot read, I will assume to be incorrect.
  – Stop writing when told to do so at the end of the exam. I will take 5 points off your exam if I have to tell you multiple times.
  – Academic misconduct will not be tolerated. Suspected academic misconduct will be immediately referred to the Emory Honor Council. Penalties for misconduct will be a zero on this exam, an F grade in the course, and/or other disciplinary action that may be applied by the Emory Honor Council.

• **TIME:** This exam has 6 questions on 11 pages including the title page. Please check to make sure all pages are included. You will have 50 minutes to complete this exam.

---

*I commit to uphold the ideals of honor and integrity by refusing to betray the trust bestowed upon me as a member of the Emory community. I have also read and understand the requirements and policies outlined above.*

Signature: ________________________________

<table>
<thead>
<tr>
<th>Question</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>12</td>
<td>13</td>
<td>7</td>
<td>8</td>
<td>5</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1. (12 points) Define each term below. You do not need to give a formal definition, just a good description. Include an example if it is helpful to your answer. Be brief.

(a) inline comment

**Solution:** A single line of text preceded by // and ignored by the compiler.
Ex: //comment

(b) Java virtual machine

**Solution:** The software that runs Java programs. The JVM compiles Java bytecode just-in-time into the machine language of the real (physical) computer.

(c) string

**Solution:** A sequence of characters
A reference type for storing sequences of characters
Ex: "a", "Hello", or "My name is ...", etc.
Common mistake: String literal vs. a string. String literals use double quotes “”, but you can initialize a string without them. Ex String line = input.next(); This is like int vs. the literal 5.

(d) datatype

**Solution:** Classification scheme for data in a programming language. Used to specify encoding of bits (information).
Ex: in Java, some datatypes are int, char, long etc

(e) keyword

**Solution:** A word reserved by the compiler.
Ex: main, static, or public, etc.

(f) casting

**Solution:** Method of converting a value of one type into a (possibly-)different type
Ex: x = (int) 4.5; etc.
2. (13 points) Evaluate each expression. Then give the result of the evaluation and the data type of the result. If the expression cannot be evaluated or is not proper Java syntax, you may simply write “error” for the value. Assume these expressions are not full statements, and that their execution is independent of each other. The first row has been done for you.

```java
String strK = "Kiwi", strB = "Birds", str2 = "25.6";
char charP = '+';
int intX = -5, intY = 3, intZ = 2;
double doubA = 7.5, doubB = 2.8;
```

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>4+1</td>
<td>5</td>
<td>int</td>
</tr>
<tr>
<td>strK.charAt(1)</td>
<td>'i'</td>
<td>char</td>
</tr>
<tr>
<td>intX / intZ + intY</td>
<td>1</td>
<td>int</td>
</tr>
<tr>
<td>str2.charAt(4)</td>
<td>Error</td>
<td>index out of range</td>
</tr>
<tr>
<td>doubA * intZ / intY</td>
<td>5.0</td>
<td>double</td>
</tr>
<tr>
<td>str2 + charP + intY * intZ</td>
<td>“25.6+6”</td>
<td>String</td>
</tr>
<tr>
<td>intX + intZ + strK</td>
<td>“-3Kiwi”</td>
<td>String</td>
</tr>
<tr>
<td>intZ * (int) doubB + intZ</td>
<td>6</td>
<td>int</td>
</tr>
<tr>
<td>intY &gt; intZ</td>
<td></td>
<td>intX = doubB</td>
</tr>
<tr>
<td>intX%3</td>
<td>-2</td>
<td>int</td>
</tr>
<tr>
<td>intX + doubleA</td>
<td>2.5</td>
<td>double</td>
</tr>
<tr>
<td>doubA &lt;= intY ^ intZ &gt; 0</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>(false ^ true)</td>
<td></td>
</tr>
<tr>
<td>doubB--</td>
<td>2.8</td>
<td>double</td>
</tr>
<tr>
<td>-5/3 &lt; -1</td>
<td>false</td>
<td>boolean</td>
</tr>
<tr>
<td></td>
<td>(-5/3 == -1 and -1 is not &lt; -1)</td>
<td></td>
</tr>
</tbody>
</table>

Notes:

- The parenthesis operator has the highest precedence of all operators.
- The logical not operator and the casting operator have higher precedence than arithmetic operators, relational/comparison operators, and the other logical operators.
- Arithmetic operators have higher precedence than relational/comparison or logical operators.
- Relational/comparison operators have higher precedence than logical operators.
- Assignment operators have the lowest precedence of all operators.
3. (7 points) Assume the statements below are part of a Java program which compiles and runs. What is the output if the user types 8?

Write the output on this side of the page:

Scanner input = new Scanner(System.in);
int num = input.nextInt();

System.out.println("Food: ");
if(num % 9 == 2){
   System.out.println("fruit");
} else if (num % 4 == 1){
   System.out.println("worms");
}
if(num % 2 == 4){
   System.out.println("seeds");
} else {
   System.out.println("grubs");
}

System.out.println("Attributes:");
if(num <= 8){
   System.out.println("chicken-sized");
} else if(num == 8){
   System.out.println("flightless");
} else if(num >= 8){
   System.out.println("bipedal");
} else {
   System.out.println("nocturnal");
}

System.out.println("Predators:");
if(num > 8){
   System.out.println("cats");
   if(num > 10){
      System.out.println("ferrets");
   } else {
      System.out.println("stoats");
   }  
} else {
   System.out.println("dogs");
   if(num != 9){
      System.out.println("humans");
   } else {
      System.out.println("pigs");
   } 
}
**Solution:**

The parenthetical words are the solution for the other version.

**Food:**
- grubs (fruit)

**Attributes:**
- chicken-sized (flightless)
- flightless (bipedal)

**Predators:**
- dogs (dogs)
- humans (pigs)

4. For each of the code fragments below, give the output. If the code results in an infinite loop, write the first few outputs, and then indicate that it is an infinite loop.

(a) (2 points)
```java
int x = 4;
while(x != 0){
    System.out.println(x);
    x -= 3;
}
```

**Solution:**
- 4
- 1
- -2
- -5
- (infinite loop)

(b) (3 points)
```java
int iter = 0;
while(iter < 10){
    System.out.println(iter);
    if(iter%3 == 0){
        iter = iter* 2-1;
    } else {
```
iter += 4;
}
}
System.out.println(iter);

Solution:
0
-1
3
5
9
17
(c) (3 points)

boolean isWednesday = false;
int day = 7;
while(!isWednesday){
    day%= 7;
    System.out.println("It’s day " + day);
    if(day >= 6){
        day-= 5;
    }
    day++;
    isWednesday = day == 3;
}
System.out.println("It’s Wednesday!");

Solution:
It’s day 0
It’s day 1
It’s day 2
It’s Wednesday!
5. (5 points) Saanvi is debugging (fixing the errors in) the following code snippet. The intended results are shown below.

<table>
<thead>
<tr>
<th>Input</th>
<th>Prints</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>Please enter larger input.</td>
</tr>
<tr>
<td>4</td>
<td>Four is an even number.</td>
</tr>
<tr>
<td>5</td>
<td>Five is a prime number.</td>
</tr>
<tr>
<td>Other input</td>
<td>Try again later.</td>
</tr>
</tbody>
</table>

On the next page, fill in the table for the current output of the program. Then, correct the code by adding lines of code and/or crossing out incorrect code and writing the corrected code above. Assume input is a Scanner reading System.in.

```java
x = input.nextInt();

switch(x-1){
    case -1:
        System.out.println("Please enter larger input.");
        break;
    case 4:
        System.out.println("Four is an even number.");
    case 5:
        System.out.println("Five is a prime number.");
        break;
    default:
        System.out.println("Try again later.");
}
```
### Solution:

switch(x-1) → switch(x),
add break; to case 4.

<table>
<thead>
<tr>
<th>Input</th>
<th>Prints</th>
</tr>
</thead>
</table>
| 5     | Four is an even number.  
       | Five is a prime number. |
| 4     | Try again later.        |
| -1    | Try again later.        |
6. (5 points) Complete the program below. It should take a line of input from a user. If that input has odd length, it should print out every odd-indexed character in the input on one line. If the input has even length, it should print out every even-indexed character in the input on one line. For example, the input Kiwis are adorable. should yield output ii r drbe

```java
import java.util.Scanner;

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

        /* --------- Write your code below this line ----------- */

        /* --------- Write your code above this line ------------ */

        System.out.println();
        }
    }
```
Solution:

```java
import java.util.Scanner;

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

        /* --------- Write your code below this line --------- */
        String line = input.nextLine();

        int index = 0;
        int length = line.length();
        while(index < length){
            // Odd length -> print odd indices
            // Even length -> print even indices
            if(index%2 == length%2){
                System.out.print(line.charAt(index));
            }
            index++;
        }

        /* --------- Write your code above this line --------- */
        System.out.println();
    }
}
```