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 10 questions on 15 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>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Points:</td>
<td>6</td>
<td>4</td>
<td>6</td>
<td>8</td>
<td>12</td>
<td>8</td>
<td>12</td>
<td>12</td>
<td>10</td>
<td>22</td>
<td>100</td>
</tr>
<tr>
<td>Score:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Score: 1
1. (6 points) Describe what function the ’+’ operator performs in the following statements:

(a) Given that: int a = 1, b = 2;
   
   a + b
   
   **Solution:** Arithmetic Operators (Additive operator), return the sum of the two operands.

(b) ’a’ + 1
   
   **Solution:** Arithmetic Operators (Additive operator), the character a will be casted/promoted to integer, the return value is an integer too.

(c) "a" + "b"
   
   **Solution:** Concatinate/join two strings "a" and "b"

2. (4 points) Describe what function the operators containing the symbol ’==’ perform in the following statements:

Given that: int a = 1, b = 2;

(a) a += b
   
   **Solution:** Assignment operator, equivalent to a = a + b; increment a on itself by b;

(b) a == b
   
   **Solution:** Equality operator, returns a boolean value; This expression returns true because a is equal to b.
3. (6 points) Convert the following binary numbers to decimal equivalents. (Show your work for partial credit)
   (a) $10001_2$

   (a) $17$

   (b) $111000_2$

   (b) $56$

4. (8 points) Convert the following decimal number to their binary. (Show your work for partial credit)
   (a) 57

   (a) $111001$

   (b) 62

   (b) $111110$
5. (12 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. The first row has been done for you.

```java
int i = 5, j = 3;
char c1 = 'A', c2 = 'B';
double d1 = 7.0, d2 = 4.0;
String s1 = "llama", s2="glama";
```

<table>
<thead>
<tr>
<th>Expression</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>i % j</td>
<td>2</td>
<td>int</td>
</tr>
<tr>
<td>i + d2</td>
<td>9.0</td>
<td>double</td>
</tr>
<tr>
<td>i + (int)d2</td>
<td>9</td>
<td>int</td>
</tr>
<tr>
<td>i + d2 &lt; j + d1</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>i-- != j++</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>c1 + 1</td>
<td>66</td>
<td>int</td>
</tr>
<tr>
<td>c1 &lt; c2</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>c1 + 1 == c2</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>s1 + c1</td>
<td>&quot;llamaA&quot;</td>
<td>String</td>
</tr>
<tr>
<td>i + c1 + s2</td>
<td>&quot;70glama&quot;</td>
<td>String</td>
</tr>
<tr>
<td>s1 + &quot;c1&quot;</td>
<td>&quot;llamacl&quot;</td>
<td>String</td>
</tr>
<tr>
<td>s1.charAt(1) == s2.charAt(1)</td>
<td>true</td>
<td>boolean</td>
</tr>
<tr>
<td>s1.charAt(2)+1 ==s2.charAt(2)+1.0</td>
<td>true</td>
<td>boolean</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.
6. (8 points) You are writing a program, trying to count how many letter ’g’ are in a string. The line numbers are given on the left.

   1:   public class Checking{
   2:     public static void main(String[] args){
   3:         
   4:         String check_word = "Debugging";
   5:         int count;
   6:         int i=0;
   7:         while(i < check_word.length()){
   8:             count=0;
   9:             if ( check_word.charAt(i)==’g’){
  10:                 count++;
  11:             }
  12:             i++;
  13:         }
  14:         System.out.println("There’re "+count+" ’g’ in this word");
  15:     }
  16: }
  17: }

(1) What is the output if check_word is "bugg"?

Solution: (1 point) There’re 1 ’g’ in this word

(2) What is the output if check_word is "glue"?

Solution: (1 point) There’re 0 ’g’ in this word

(3) From (1) and (2), is this result correct? If not, identify the bug and correct it. You do not need to rewrite the entire program, just the lines which contains the error.

Solution: The variable count is initialized inside the while loop. Thus at each loop it will be initialized with 0 again. The solution is to move it in front of the while loop.
... 
7: count=0;
8: while(i < check_word.length()) {
   ...

grading:
2 points: identify the bug
4 points: fix the bug correctly
7. (12 points) In Emory we have two swimming pools, one in Woodruff PE center, the other is in Clairmont Campus. Here is a program for query swimming pool open hour. Suppose this program compiles and runs without error.
We use 1-7 to represent Sun, Mon, ..., Sat
We use 0 - 23 to represent 24 hours in a day (0=midnight, 12=noon).

```java
import java.util.Scanner;
public class Pool {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("Welcome to Emory Pool Hour Query System!");
        System.out.print("Please input a day of week in integer(1-7): ");
        int day = in.nextInt();
        System.out.print("please input a time of day in integer(0-23): ");
        int time = in.nextInt();

        boolean woodruff=false;
        boolean clairmont=false;

        switch ( day ) {
            case 2:
            case 4:
            case 6:
                if ( time >=12 && time <=20 )
                    clairmont = true;
                if (time >= 7 && time <=8 || time >= 12 && time <=13)
                    woodruff = true;
                break;

            case 3:
            case 5:
                if ( time >=12 && time <=20 )
                    clairmont = true;
                if ( time >= 11 && time <=13)
                    woodruff = true;
                break;

            case 1:
            case 7:
                if (time >=10 && time <=17 )
                    clairmont = true;
                if (time >=13 && time <=16 )
                    woodruff = true;
                break;
        }
    }
}
```
default:
    System.out.println("invalid day!");
    break;
}

System.out.println("On day "+day+" at the time "+time+";");
if(clairmont)
    System.out.println("Swimming pool in Clairmont campus is open!");
if(woodruff)
    System.out.println("Swimming pool in Woodruff PE center is open!");
if(!(clairmont || woodruff))
    System.out.println("no pool is open!");

}

(1) Suppose we want to check if any pool is open on Wednesday morning, 11am. Write down the output result if we enter 4 for day and 11 for time

(2) Suppose we want to check if any pool is open on Friday afternoon, 1pm. Write down the output result if we enter 6 for day and 13 for time

Solution:

(1)
On day 4 at the time 11:
no pool is open!

(2)
On day 6 at the time 13:
Swimming pool in Clairmont campus is open!
Swimming pool in Woodruff PE center is open!

grading:
6 points for each question; within each question:
2 points for the output format first message
4 points for the correctness of workflow
8. (12 points) Our task is to form a triangle using matchsticks. Suppose all the matchsticks are of the same length, and we ignore the width of the matchstick.

You want to write a program that reads in two integers as the length of two edges, (e.g. if we input 3 and 4, then we use 3 matchsticks to form one edge, and 4 matchsticks to form another edge) then print out all the possible lengths for the third edge (an integer, representing the number of matchsticks) to form the 3rd side.

Hint: For the 2 given edge, suppose \(a\) is the length of the shorter edge, \(b\) is the longer edge. then the length of the third edge \(c\) should satisfy: \(b - a < c < b + a\).

```java
import java.util.Scanner;
public class Triangle {
    public static void main(String[] args) {
        Scanner myScanner = new Scanner(System.in);
        System.out.print("Enter length of the first edge: ");
        int a = myScanner.nextInt();
        System.out.print("Enter length of the second edge: ");
        int b = myScanner.nextInt();

        /*------- Your code here -------*/
    }
}
```
Solution:

```java
if ( a > b ){
    int c = a;
    a = b;
    b = c;
}

int i=b-a+1;
while (i < a+b){
    System.out.println(i);
    i++;
}
```

grading:
4 points: swap a,b; or handling both a < b and a > b
2 points: variable initialization
2 points: loop terminate correctly
2 points: output
2 points: Format, programming style, naming convention, etc.
9. (10 points) The Chinese Remainder Theorem (CRT) will determine a number n that when divided by some given divisors leaves given remainders. For example:

“If we know the following remainders of an integer:

• remainder a if divided by 3
• remainder b if divided by 5
• remainder c if divided by 7

Then we can find the smallest possible such number using CRT”.

YOUR JOB: Write a program in order to validate the CRT. In this program, it well read in the following numbers from user:

• a: the remainder if the number is divided by 3
• b: the remainder if the number is divided by 5
• c: the remainder if the number is divided by 7
• n: the maximum integer we want to try

The program will print all numbers between 0 and n(inclusive) which satisfy the following rules:

• The remainder when the number is divided by 3 is a;
• The remainder when the number is divided by 5 is b;
• The remainder when the number is divided by 7 is c;

Hint: Use while loop, and try all the integers from 1 to n. if this integer satisfies: (1) has remainder a if divided by 3; (2) has remainder b if divided by 5; (3) has remainder c if divided by 7, then print them out.

import java.util.Scanner;
public class CRT {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("Input remainder a, if divided by 3:");
        int a = in.nextInt();
        System.out.println("Input remainder b, if divided by 5:");
        int b = in.nextInt();
        System.out.println("Input remainder c, if divided by 7:");
        int c = in.nextInt();
        System.out.println("Input maximum integer n:");
        int n = in.nextInt();
        for (int i = 1; i <= n; i++) {
            if (i % 3 == a && i % 5 == b && i % 7 == c) {
                System.out.println(i);
            }
        }
    }
}
System.out.println("Input remainder c, if divided by 7:");
int c = in.nextInt();
System.out.print("Please input the maximum positive integer: ");
int n = in.nextInt();

System.out.println("\nNow, find all the integers that satisfies:");
System.out.println(" *has remainder of "+a+" if divided by 3");
System.out.println(" *has remainder of "+b+" if divided by 5");
System.out.println(" *has remainder of "+c+" if divided by 7");

/*------- Your code here -------*/

Solution:

int i=0;
while(i <= n){
    if( (i % 3 ==a) && (i % 5 == b) && (i % 7 ==c)){
        System.out.println(i);
    }
    i++;
}

grading:
2 points: use a,b,c,n correctly
3 points: loop terminates correctly
3 points: conditions on a, b and c
2 points: output
10. (22 points) You want to write a program that reads in an email address from the user, and checks:

- how many upper case letter(s) are in the user name
- how many lower case letter(s) are in the user name
- how many numbers are in the username

*Hint:*

- user name is the part before character '@'
- You might need the ASCII table (attached at the end) to identify the ranges for letters and numbers:

Examples of running this program:

```java
>>> java CheckUserName
Enter your email address: Txu28@emory.edu
# of upper case letter: 1;
# of lower case letter: 2;
# of numbers: 2

>>> java CheckUserName
Enter your email address: xin.chen2@emory.edu
# of upper case letter: 0;
# of lower case letter: 7;
# of numbers: 1
```
import java.util.Scanner;
public class CheckUserName {
    public static void main(String[] args) {
        Scanner myScanner = new Scanner(System.in);
        System.out.print("Enter your email address: ");
        String s = myScanner.next();
        /*------ Your code here -------*/
    }
}

Solution:

    int i = 0;
    int numOfUpper = 0;
    int numOfLower = 0;
    int numOfNum = 0;
    while(s.charAt(i) != '@'){
        int ascii = (int)s.charAt(i); //or char ascii=s.charAt(i);
        if( ascii <= 57 && ascii >=48){
            numOfNum++;
        }else if( ascii >=65 && ascii <=90){
            numOfUpper++;
        }else if( ascii >=97 && ascii <=122){
            numOfLower++;
        }else if( ascii >=32 && ascii <=39){
            numOfNum++;
        }else if( ascii >=41 && ascii <=57){
            numOfNum++;
        }else if( ascii >=57 && ascii <=64){
            numOfNum++;
        }else if( ascii >=64 && ascii <=90){
            numOfUpper++;
        }else if( ascii >=91 && ascii <=126){
            numOfNum++;
        }else if( ascii >=126 && ascii <=126){
            numOfNum++;
        }
        i++;
    }
else if( ascii >=97 && ascii <=122){
    numOfLower++;
}
    i++;
}
System.out.println("# of upper case letter: " + numOfUpper);
System.out.println("# of lower case letter: " + numOfLower);
System.out.println("# of number: " + numOfNum);

grading:
3 points: variable initialization
3 points: loop ends at symbol "@"
12 points: 4 points for each of the counter
2 points: iter++ (or using for loop correctly)
2 points: output