### Building Java Programs Chapter 5 Lab Handout

### While Loop Mystery

1. Consider the following method. For each call below, indicate what output is produced.

|  |  |
| --- | --- |
| public static void mystery1(int x) {  int y = 1;  int z = 0;  while (2 \* y <= x) {  y \*= 2;  z++;  }  System.out.println(y + " " + z);  } | **Method Call Output Produced**  mystery1(1); 1 0  mystery1(6); 4 2  mystery1(19); 16 4  mystery1(39); 32 5  mystery1(74); 64 6 |

2. Consider the following method. For each call below, indicate what output is produced.

|  |  |
| --- | --- |
| public static void mystery2(int x) {  int y = 0;  while (x % 2 == 0) {  y++;  x /= 2;  }  System.out.println(x + " " + y);  } | **Method Call Output Produced**  mystery2(19); 19 0  mystery2(42); 21 1  mystery2(48); 3 4  mystery2(40); 5 3  mystery2(64); 1 6 |

3. Consider the following method. For each call below, indicate what value is returned.

|  |  |
| --- | --- |
| public static int mystery3(int x, int y) {  while (x != 0 && y != 0) {  if (x < y) {  y -= x;  } else {  x -= y;  }  }  return x + y;  } | **Method Call Value Returned**  mystery3(3, 3); 3  mystery3(5, 3); 1  mystery3(2, 6); 2  mystery3(12, 18); 6  mystery3(30, 75); 15 |

### While Loop Programming

4. Write a static method named showTwos that shows the factors of 2 in a given integer. For example, the following calls produce the following output:

Call Output

showTwos(7); 7 = 7

showTwos(18); 18 = 2 \* 9

showTwos(68); 68 = 2 \* 2 \* 17

showTwos(120); 120 = 2 \* 2 \* 2 \* 15

The idea is to express the number as a product of factors of 2 and an odd number. The number 120, for example, has 3 factors of 2 multiplied by the odd number 15. For odd numbers (as in the first example of 7), there are no factors of 2, so you just show the number itself. Assume that your method is passed a number greater than 0.

Test your code with the following code file:

public class TestShowTwos {

public static void main(String[] args) {

showTwos(7); // 7 = 7

showTwos(18); // 18 = 2 \* 9

showTwos(68); // 68 = 2 \* 2 \* 17

showTwos(120); // 120 = 2 \* 2 \* 2 \* 15

}

// your code goes here

}

Text

Description automatically generated

IS THIS OKAY?????

5. Write a static method named gcd that accepts two integers as parameters and returns the greatest common divisor (GCD) of the two numbers. The greatest common divisor of two integers *a* and *b* is the largest integer that is a factor of both *a* and *b*. The GCD of any number and 1 is 1, and the GCD of any number and 0 is that number. One efficient way to compute the GCD is to use Euclid's algorithm, which states the following:

*GCD(a, b) = GCD(b, a % b)*

*GCD(a, 0) = Absolute value of a*

For example, gcd(24, 84) returns 12, gcd(105, 45) returns 15, and gcd(0, 8) returns 8.

Test your code with the following code file:

public class TestGCD {

public static void main(String[] args) {

System.out.println("GCD of 27 and 6 is " + gcd(27, 6)); // 3

System.out.println("GCD of 24 and 84 is " + gcd(24, 84)); // 12

System.out.println("GCD of 38 and 7 is " + gcd(38, 7)); // 1

System.out.println("GCD of 45 and 105 is " + gcd(45, 105)); // 15

System.out.println("GCD of 1 and 25 is " + gcd(1, 25)); // 1

System.out.println("GCD of 25 and 1 is " + gcd(25, 1)); // 1

System.out.println("GCD of 0 and 14 is " + gcd(0, 14)); // 14

System.out.println("GCD of 14 and 0 is " + gcd(14, 0)); // 14

}

// your code goes here

}

### Text Description automatically generatedBoolean Logic

6. Write a method named season that takes two integers as parameters representing a month and day, and that returns a String indicating the season for that month and day. Assume that months are specified as an integer between 1 and 12 (1 for January, 2 for February, and so on) and that the day of the month is a number between 1 and 31.

For this problem, winter occurs between 12/16 and 3/15. Spring is between 3/16 and 6/15. Summer is between 6/16 and 9/15. Fall is between 9/16 and 12/15.

Text

Description automatically generated

IS THIS RIGHT???

### Random Numbers

7. Write a method randomWalk that performs a random one-dimensional walk, reporting each position reached and the maximum position reached during the walk. The random walk should begin at position 0. On each step, you should either increase or decrease the position by 1 (with equal probability). The walk stops when 3 or -3 is hit. The output should look like this:

position = 0

position = 1

position = 0

position = -1

position = -2

position = -1

position = -2

position = -3

max position = 1

Text

Description automatically generated

8. Write code that prints a random number of lines between 2 and 10 lines inclusive, where each line contains a random number of 'x' characters between 5 and 20 inclusive. For example:

xxxxxxx

xxxxxxxxxxxxxxxxxx

xxxxxxxxxxxx

xxxxxx

xxxxxxxxxxx

xxxxxxxxxxxxxxxx

A screenshot of a computer

Description automatically generated with medium confidence

CHECK BOUNDS???

9. Write an interactive program that prompts for a desired sum, then repeatedly rolls two six-sided dice until their sum is the desired sum. Here is the expected dialogue with the user:

Desired dice sum: 9

4 and 3 = 7

3 and 5 = 8

5 and 6 = 11

5 and 6 = 11

1 and 5 = 6

6 and 3 = 9

Finish the following code file:

import java.util.\*;

public class TestDiceRoll {

public static void main(String[] args) {

Scanner console = new Scanner(System.in);

System.out.print("Desired dice sum: ");

int sum = console.nextInt();

// your code here

}

}

A screenshot of a computer

Description automatically generated with medium confidence