Exercises BOOLEAN PROOFS Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Period\_\_\_\_

1) The logical AND is represented by the operator &&. The logical OR is represented by the operator ||. Evaluate

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| true && true | 🡪 |  |  | true || true | 🡪 |  |
| true && false | 🡪 |  |  | true || false | 🡪 |  |
| false && true | 🡪 |  |  | false || true | 🡪 |  |
| false && false | 🡪 |  |  | false || false | 🡪 |  |

2) Show the truth table to prove or disprove that (A && B) || A is the same as A || B

A B A && B (A && B) || A A || B

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| false | false |  |  |  |  |  |
| false | true |  |  |  |  |  |
| true | false |  |  |  |  |  |
| true | true |  |  |  |  |  |

3) Prove that both sides of De Morgan’s Laws are true:

!(A && B) == !A || !B !(A || B) == !A && !B

A B A && B !(A && B) !A !B !A || !B

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| false | false |  |  |  |  |  |
| false | true |  |  |  |  |  |
| true | false |  |  |  |  |  |
| true | true |  |  |  |  |  |

A B A || B !(A || B) !A !B !A && !B

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| false | false |  |  |  |  |  |
| false | true |  |  |  |  |  |
| true | false |  |  |  |  |  |
| true | true |  |  |  |  |  |

4) Simplify the following using DeMorgan’s Thm.

1. !(x <= 0 || y > 5)
2. !(x > 0 && y <= 5)
3. !(x == 0 || y >= 5)
4. !(x < 0 && y == x && z.equals(“N”))
5. !(x > 0 || (y >= 5 && z.equals(“N”))
6. !(x != 0 && (y == 0 || !z.equals(“Y”))

5) Prove or disprove that A && (B || C) is equivalent to (A && B) || (A && C)

For convenience, let’s use 0 to represent false and 1 for true.

A B C

0 0 0

0 0 1

0 1 0

0 1 1

1 0 0

1 0 1

1 1 0

1 1 1

6) State the logical problem with the following code, and how you would fix the problem:

System.out.println(“enter a positive number:”);

double n = input.nextDouble();

if(Math.sqrt(n) <= 12 && n>=0)

System.out.println(“good size for a multiplication table”);

else

System.out.println(“Negative or too large”);

# **Exercises** if-else construct Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Period\_\_\_\_\_\_

Remember the difference between OR(||) and AND(&&) as well as DIV (/) and MOD (%). Write the output to the following code:

1. int x=2, y=3;

if (x%2 == 0 || y%2==0)

System.out.println(“Lobster Magnet”);

else

System.out.println(“My spoon is too big”);

1. int x=2, y=3;

if (x%2 == 0 && y%2==0) //NOTE THE DIFFERENCE WITH #1

System.out.println(“I wasn’t born with enough middle fingers”);

else

System.out.println(“Uncle Meat and Joe’s Garage”);

1. int x=32767, y=100, z;

z = x%y;

if (z == 67)

System.out.println(“Captain Howdy”);

else

if (z == 327)

System.out.println(“Head A Splode”);

else

System.out.println(“Tony Bennet”);

1. int x=32767, y=100, z;

z = x%y;

if (z == 67)

System.out.println(“Captain Howdy”);

if (z == 327) //NOTE THE DIFFERENCE WITH #3

System.out.println(“Head A Splode”);

else

System.out.println(“Tony Bennet”);

1. int x=2, y=3;

if (((y/x) == (y%x)) && (x-y>0))

x++;

else

y++;

System.out.println(x + “, “ + y);

1. int x=2, y=3;

if ((Math.sqrt(x+y) > x) || (Math.sqrt(x+y) > y))

x = x +10;

else

y = y + 10;

System.out.println(x + “, “ + y);

1. int x=3, y=2, z=1;

if (y>z)

System.out.println(z);

else

if (x>z)

System.out.println(y);

else

System.out.println(x);

1. int x=3, y=2, z=1;

if (y>z)

System.out.println(z);

if (x>z) //NOTE THE DIFFERENCE WITH #7

System.out.println(y);

else

System.out.println(x);

1. int x = 3, y=2, z=1;

if(x == y-1)

if(y == z+1)

System.out.println(x);

else

System.out.println(“WRONG!”);

Exercises nested loops Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Period\_\_\_\_\_\_\_\_\_

Write the output for the following loops:

1) for(int r = 0; r < 3; r++)

{

for(int c = 0; c < 3; c++)

System.out.print(r + “:” + c + “ “);

System.out.println();

}

2) for(int r = 6; r >= 3; r--)

{

for(int c = 8; c < 10; c++)

System.out.print(r + “:” + c + “ “);

System.out.println();

}

3) for(int r = 2; r <=4; r++)

{

for(int c = 5; c >=3; c--)

System.out.print(r + “:” + c + “ “);

System.out.println();

}

4) for(int r = 8; r > 5; r--)

{

for(int c = 4; c > 2; c--)

System.out.print(r + “:” + c + “ “);

System.out.println();

}

5) for(int r = 0; r <= 4; r = r + 2)

{

for(int c = 0; c < 3; c++)

System.out.print(r + “:” + c + “ “);

System.out.println();

}

Write a nested loop to produce the following output:

6) 7:4 7:5 7:6

8:4 8:5 8:6

7) 2:7 2:8 2:9

1:7 1:8 1:9

8) 2,3 2,4 2,5

3,3 3,4 3,5

4,3 4,4 4,5

9) Consider the following method: What is returned from mystery(4)?

public static int mystery(int x)

{

while(x < 5)

{

int y = 1;

while(y < 5)

{

y++;

x += y;

}

}

return x;

}

10) Consider the following method: What is returned from stuff(5)?

public static int stuff(int n)

{

int temp = 0;

for(int x=1; x <= n; x++)

for(int y=0; y <= n/2; y++)

temp++;

return temp;

}

Binary is a number system consisting of only two symbols, 0 and 1. Any number in decimal, our native number system, can also be represented in binary. We teach it in computer science because it is the native language of computers. When computer code is processed down to the hardware level, all of those logical circuits work on the premise that there is either something going through a little wire (1) or there is nothing (0). We can use knowledge of binary numbers to write efficient code, even with advanced languages like C, C++ and Java. But before we get to that, we need a good working understanding of the binary number system.

Consider that when you count, you are expressing quantities with numeric words, and those words are comprised of symbols. The decimal number system has 10 symbols, thus, the name decimal. Imagine an odometer, where mileage is shown with a series of dials, and each dial has all of the symbols in decimal: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, and after 9, we loop back around to 0. Driving that next mile means that a dial advances from one number to the next. But if we advance past 9, it loops back around to 0 and the digit to the left advances one.

This all may be considered too elementary to describe, but now perform that same thought experiment with a twist: imagine that each dial only has a 0 and a 1 on it. Compare how you would count on a binary odometer to the way we do in decimal:

Decimal (10 symbols) Binary (2 symbols)

1. 0000
2. 0001
3. 0010
4. 0011
5. 0100
6. 0101
7. 0110
8. 0111
9. 1000
10. 1001
11. 1010
12. 1011
13. 1100
14. 1101
15. 1110
16. 1111

A few items to note: each binary number is represented with 4 binary digits (bits). If we wanted to represent the decimal number 16, we would need 5 bits. Also notice the pattern you see in the binary column: the 1st digit alternates every 1 row (20). The 2nd digit alternates every 2 rows (21). The 3rd digit alternates every 4 rows (22) and the most significant digit alternates every 8 rows (23). Is it a Coincidence?

No, it is not a coincidence. When we consider a decimal number, you learn that the least significant digit is in the 1’s place (100), the 2nd digit is in the 10’s place (101), the 3rd digit is in the 100’s place (102) and the 4th digit is in the 1000’s place (103). With binary, we have the same relationship, but the place for each digit is represented with a power of 2.

8439 = (8 \* 103) + (4 \* 102) + (3 \* 101) + (9 \* 100) = 8000 + 400 + 30 + 9 = 8439

Now consider a binary number: the least significant digit is in the 1’s place (20), the 2nd digit is in the 2’s place (21), the 3rd digit is in the 4’s place (22), the 4th digit is in the 8’s place (23), etc.

11012 = (1 \* 23) + (1 \* 22) + (0 \* 21) + (1 \* 20) = 8 + 4 + 0 + 1 = 13.

These are not different numbers. 11012 and 1310 are two different words that represent the same quantity.

Translating a decimal number into binary is just as easy. Divide 2 into the number and store the remainder. Your first remainder will be the least significant digit in your answer. Divide 2 into the resulting quotient and store the remainder, which will be the second digit. Repeat the process until the quotient is 0, and your last remainder will be the most significant digit.

So for 87, 2 goes into 87 forty three times with a remainder of 1 (least significant digit).

2 goes into 43 twenty one times with a remainder of 1.

2 goes into 21 ten times with a remainder of 1.

2 goes into 10 five times with a remainder of 0.

2 goes into 5 two times with a remainder of 1.

2 goes into 2 one time with a remainder of 0.

2 goes into 1 zero times with a remainder of 1 (most significant digit).

So, 8710 = 10101112.

We can reverse the process to check accuracy:

10101112 = (1 \* 26) + (0 \* 25) + (1 \* 24) + (0 \* 23) + (1 \* 22) + (1 \* 21) + (1 \* 20) = 64 + 16 + 4 + 2 + 1 = 8710

A few items to note: the process of translating between binary and decimal (and back) can be achieved with any number system. If the number system has more than ten symbols, we use capital letters.

The hexadecimal number system has sixteen symbols: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

The number A16 represents the same quantity as 1010. B16 represents the same quantity as 1110.

F16 = 1510. 1016 = 1610. Confused?

Try counting using the odometer system, but imagine that each dial contains all of the numbers in the hexadecimal system: after 9 comes the number A. It is not until we move past the number F that we loop back around to 0.

Exercises Number Systems Name\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Period\_\_\_\_\_

Translate the following binary numbers into decimal:

1. 101101
2. 100001
3. 110000
4. 101100
5. 111111

Translate the following decimal numbers into binary:

1. 53
2. 26
3. 31
4. 16
5. 61

11) Consider that you have a base-5 number system. You only have the symbols 0, 1, 2, 3 and 4 to express quantities. Write the first twenty numbers in base 5.

12) You have seen how to translate a binary number into decimal by adding up each digit multiplied by the base (2) to a certain power. Now translate 3025 into decimal.

13) Given the means of translating binary into decimal, consider how you would translate a number that has digits to the right of the decimal point: what place does each digit have when it is on the right side of the decimal point? Translate 101.112 into decimal.

14) Translate 3A16 into decimal.

15) Translate D4.F16 into decimal.

16) Consider your first name. What base number system would you need in order to have the digits 0 through 9 and then all of the letters in your first name? Assume, like with Hex, A would equate to 10, B would equate to 11, etc.

17) Consider your name represented as a number in that number system. Represent your name as its base 10 equivalent by expanding it into a sum of products.