Template Week 2 – Logic

Student number: 590190

Assignment 2.1: Parking lot

Which gates do you need?

- AND

Complete this table

Parking lot 1	Parking lot 2	Parking lot 3	Result (full)
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	1	0	0
1	0	1	0
1	1	1	1

Assignment 2.2: Android or iPhone

Which gates do you need?

- XOR

Complete this table

Android phone	iPhone	Result (Phone in possession)
0	0	0
1	0	1
0	1	1
1	1	0

Assignment 2.3: Four NAND gates

Complete this table

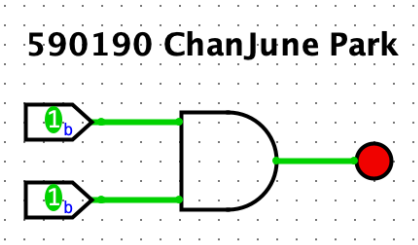
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

How can the design be simplified?

- Use XOR Gate

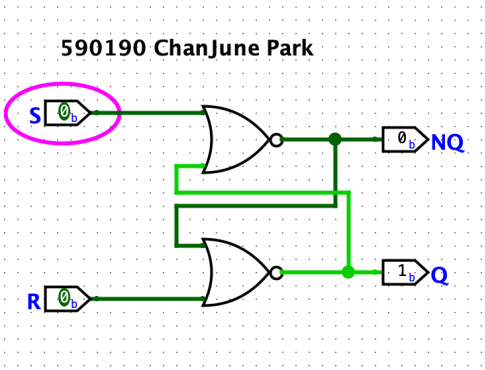
Assignment 2.4: Getting to know Logisim evolution

Screenshot of the design with your name and student number in it:



Assignment 2.5: SR Latch

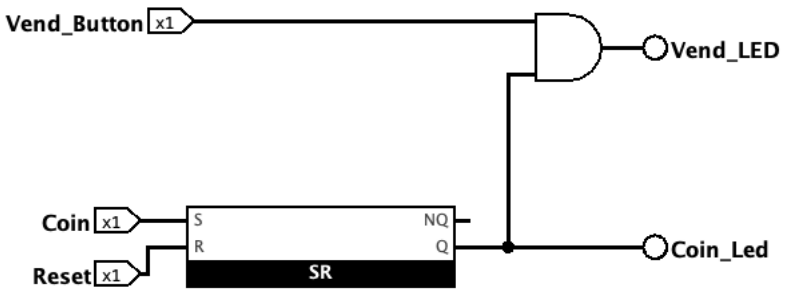
Screenshot SR Latch in Logisim with your name and student number:



Assignment 2.6: Vending Machine

Screenshot Vending Machine in Logisim with your name and student number:

590190 ChanJune Park



Assignment 2.7: Bitwise operators

Complete the java source code for bitwise operators. Put the source code here.

#1 even or odd

Examine the table on the right.

When does a binary number become odd?

Tip: Look at the Least Significant Bit (LSB)

- Every odd number ends with a binary 1.
- Every even number ends with a binary 0.

Could you write java code to check if a integer number is odd, without using the modulo % operator?

That is possible with the bitwise operators. But why would we use them? Bitwise operators are more efficient than the modulo operator because they operate directly on the binary representation of numbers, allowing for faster calculations without the overhead of division.

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111

```
public class Main {
  public static void main(String[] args) {
      int number = 5;
      if((number & 1) == 1)
      {System.out.println("number is odd"); }
      else {
      System.out.println("number is even"); }

}
}
```

Modify the condition of the if statement so that it actually checks if the number is odd or even.

Tip: use the bitwise & AND operator.

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#2 Power of 2

Examine the table on the right.

When does a binary number become a power of 2?

Tip: Look the numbers: 2, 4, 8 and 16

- When the binary representation has exactly one '1' bit and all other bits are '0's

Write java code to check if a integer number is a power of 2 by using the & bitwise operator.

Check the binary number pairs:

- 2 and 1
- 4 and 3
- 8 and 7
- 16 and 15

Decimal	Binary
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000

```
public class Main {

  public static void main(String[] args) {
      int number = 4;
      if(number > 0 && (number & (number – 1)) == 0)
      System.out.println("number is a power of 2"); else
      System.out.println("number isn't a power of 2");

}
}
```

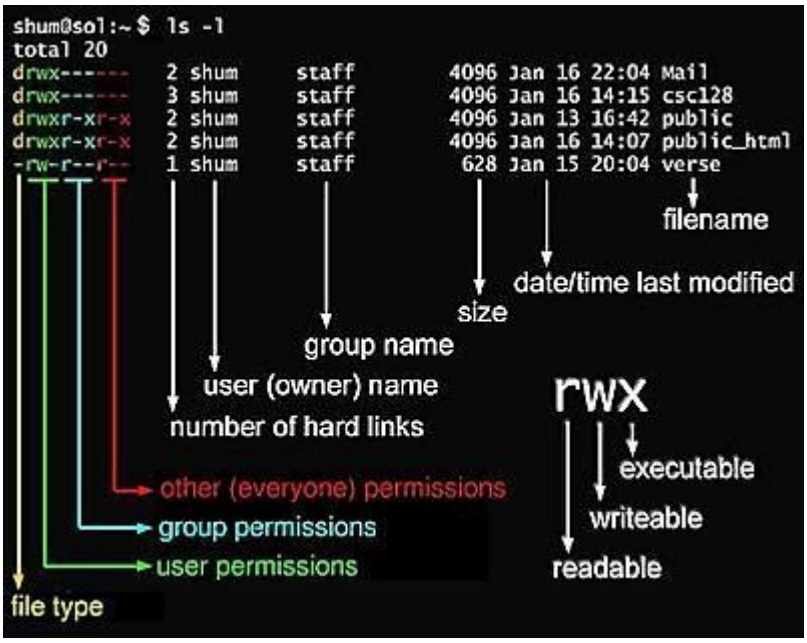
Modify the condition of the if statement so that it actually checks if the number is a power of 2.

Tip: use the bitwise & AND operator.

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#3 Check permissions

As learned in the module 1.1 introduction to infrastructures Linux uses file permissions. In Linux, file permissions are represented in octal format using three digits, each ranging from 0 to 7. Each digit corresponds to a set of permissions for the owner, group, and others, respectively. The permissions are defined as follows: read (4), write (2), and execute (1). For example, an octal value of 755 means the owner has read, write, and execute permissions (7), while the group and others have read and execute permissions (5). The sum of these values determines the permissions assigned to each category.



What are the file permissions on the file verse in the above picture?
Write the answer as an octal value.

- User: rw- -> Read(4) + Write(2) + No Execute(0) = 6
- Group: r-- -> Read(4) + No Write(0) + No Execute(0) = 4
- Other: r-- -> Read(4) + No Write(0) + No Execute(0) = 4
- Answer: 644

	User			Group			Other		
Permission	r	w	x	r	w	x	r	w	x
Octal	4	2	1	4	2	1	4	2	1
Setting Value	Sum of above			Sum of above			Sum of above		

```
public class Main {
  public static void main(String[]
args) {     final int READ = 4;
```

```
final int WRITE = 2;     final int
EXECUTE = 1;

    int userPermissions = 7;

    if((userPermisson & READ) == READ) System.out.println("User
has read permissions");     else System.out.println("User can't
read. No permissions.");

}
}
```

Modify the condition of the if statement so that it actually checks the read permissions.
Use the bitwise & AND operator for this assignment.

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#4 Assign permissions

In Linux, you want to be able to assign permissions to a user.

```
public class Main {
  public static void main(String[]
args) {     final int READ = 4;
final int WRITE = 2;     final int
EXECUTE = 1;

    int userPermissions = READ | EXECUTE;
    System.out.println("User permissions: "+userPermissions);


}
}
```

Modify the userPermissions so that this user gets READ and EXECUTE permissions. You have to use the bitwise | OR operator and the given constants.

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#5 Update permissions

In Linux, if a user has certain permissions, you want to be able to change those permissions.

```
 public class Main {   public static
void main(String[] args) {     final
int READ = 4;     final int WRITE = 2;
final int EXECUTE = 1;
```

```
    int userPermissions = 6;
userPermissions = userPermissions ^
WRITE;
    System.out.println("User permissions: "+userPermissions);


}
}
```

The user now has READ and WRITE permissions. Modify the userPermissions so that this user only gets READ permissions. Remove the WRITE permissions. You have to use the bitwise ^ XOR operator and the given constants.

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#6 Two's complement

Implement the two's complement method so that we can change a positive number into a negative one, and back again. You have to use the bitwise ~ NOT operator.

```
 public class Main {
  public static void main(String[] args)
{    int number = 5;     number = ~number
+ 1;
    System.out.println("Number: "+number);


}
}
```

Compile and run your modified code in this website: Online Java Compiler. Or use IntelliJ IDEA.

#7 Display binary, octal and hexadecimal values

This example will show you how to convert an integer value to binary, octal or hexadecimal values.

```
public class Main {     public static
void main(String[] args) {

        int number = 10;
        System.out.println("Decimal integer: "+number);

        String binary = Integer.toBinaryString(number);
```

```
        String octal = Integer.toOctalString(number);
        String hexadecimal = Integer.toHexString(number);

        System.out.println("Binary representation: " + binary);
        System.out.println("Octal representation: " + octal);
        System.out.println("Hexadecimal representation: " + hexadecimal);

    }
}
```

Assignment 2.8: Java Application Bit Calculations

Create a java program that accepts user input and presents a menu with options.

1. Is number odd?
2. Is number a power of 2?
3. Two's complement of number?

Implement the methods by using the bitwise operators you have just learned.

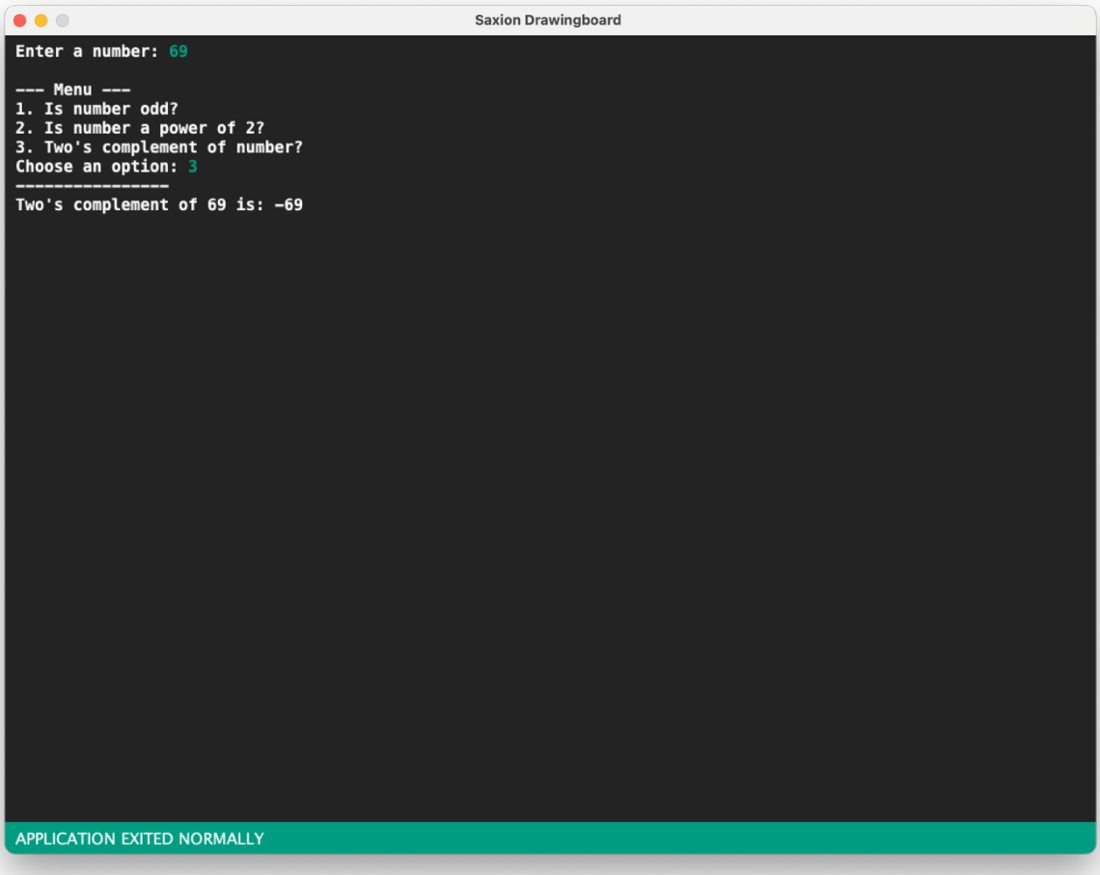
Organize your source code in a readable manner with the use of control flow and methods.

Keep this application because you need to expand it in week 6 for calculating network segments.

Paste source code here, with a screenshot of a working application.

```
import nl.saxion.app.SaxionApp;

public class Application implements Runnable {

    public static void main(String[] args) {
        SaxionApp.start(new Application(), 1024, 768);
    }

    public void run() {
        SaxionApp.print("Enter a number: ");
        int number = SaxionApp.readInt();

        SaxionApp.printLine("\n--- Menu ---");
        SaxionApp.printLine("1. Is number odd?");
        SaxionApp.printLine("2. Is number a power of 2?");
        SaxionApp.printLine("3. Two's complement of number?");
        SaxionApp.print("Choose an option: ");
        int choice = SaxionApp.readInt();
        SaxionApp.printLine("----------------");

        if (choice == 1) {
            boolean odd = isOdd(number);
            SaxionApp.printLine("Is " + number + " odd? " + odd);
        }
        else if (choice == 2) {
            boolean powerOfTwo = isPowerOfTwo(number);
            SaxionApp.printLine("Is " + number + " a power of 2? " +
powerOfTwo);
        }
        else if (choice == 3) {
```

```
            int twosComplement = getTwosComplement(number);
            SaxionApp.printLine("Two's complement of " + number + " is: " +
twosComplement);
        }
        else {
            SaxionApp.printLine("Invalid option.");
        }
    }
    public boolean isOdd(int n) {
        return (n & 1) == 1;
    }

    public boolean isPowerOfTwo(int n) {
        return (n > 0) && ((n & (n - 1)) == 0);
    }

    public int getTwosComplement(int n) {
        return ~n + 1;
    }
}
```



Ready? Then save this file and export it as a pdf file with the name: **week2.pdf**