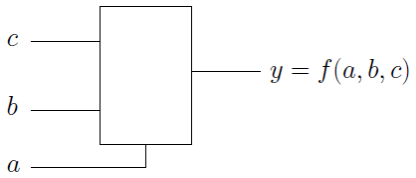
This is an optional homework assignment. Sorry that it is quite late, but I haven't had a stable internet connection yet, in order to have a look at what Bryant discussed this week. I tried to come up with some interesting questions and I am quite unsure, whether you are able to do them / whether you covered the material in class. So don't worry, if you have no idea what so ever when trying to solve one of the problems. The questions that I came up with also only cover a small portion of the previous week's material.

1. Simplify the following formula.
   - $NOT\ \big((a\ AND\ (NOT\ b))\ OR\ ((NOT\ a)\ AND\ b)\ OR\ (a\ AND\ b)\big)$
     - $\overline{\big((a \wedge \bar{b}) \vee (\bar{a} \wedge b) \vee (a \wedge b)\big)}$
     - $\overline{\big((a \wedge \bar{b}) \vee ((\bar{a} \vee a) \wedge b)\big)}$
     - $\overline{\big((a \wedge \bar{b}) \vee b\big)}$
     - $(a \vee \bar{b}) \wedge b$
     - $\bar{a} \wedge \bar{b}$
2. Simplify the following formula.
   - $(\bar{a} \rightarrow b) \leftrightarrow \overline{(a \rightarrow b)}$
     - $(a \vee b) \leftrightarrow \overline{(\bar{a} \vee b)}$
     - $(a \vee b) \leftrightarrow (a \wedge \bar{b})$
     - $\big((a \vee b) \wedge (a \wedge \bar{b})\big) \vee \big(\overline{(a \vee b)} \wedge \overline{(a \wedge \bar{b})}\big)$
     - $\big((a \vee b) \wedge (a \wedge \bar{b})\big) \vee \big((\bar{a} \wedge \bar{b}) \wedge (\bar{a} \vee b)\big)$
     - $\big((a \vee b) \wedge a \wedge \bar{b}\big) \vee \big(\bar{a} \wedge \bar{b} \wedge (\bar{a} \vee b)\big)$
     - $(a \wedge \bar{b}) \vee (\bar{a} \wedge \bar{b})$
     - $\bar{b} \vee (a \vee \bar{a})$
     - $\bar{b}$
3. Lookup what a multiplexer is. Give the truth table of $f(a,b,c)$, which is described in the following figure and then try to specify the related DNF.



   - The truth table of $f(a,b,c)$ is described in the following figure.

a	b	c	$f(a,b,c)$
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

   - $(\bar{a} \wedge b \wedge \bar{c}) \vee (\bar{a} \wedge b \wedge c) \vee (a \wedge b \wedge c)$
4. Given are the following functions, where $x \in \mathbb{N}$.
   - $R(x) : x\ is\ divisible\ by\ 3$

- $S(x) : x\ is\ divisible\ by\ 6$
- $P(x) : x\ is\ a\ prime\ number$

Verbally define the following statements and state whether they are true or false.

- $R(10)$
  - 10 is divisible by 3
  - $false$
- $\exists\, x\, [R(x)]$
  - natural numbers exist, that are divisible by 3
  - $true$
- $\forall\, x\, [R(x)]$
  - all natural number is divisible by 3
  - $false$
- $\forall\, x\, [S(x) \rightarrow R(x)]$
  - if a natural number is divisible by 6, it is also divisible by 3
  - $true$
- $\exists\, x\, [P(x) \wedge S(x+1)]$
  - natural numbers exist, such that they are prime numbers and their successor is divisible by 3
  - $true$
- $\exists!\, x\, [P(x) \wedge S(x+1)]$
  - exactly one natural number exists, such that it is a prime number and its successor is divisible by 3
  - $false$
- $\forall\, x\, [R(x) \vee R(x+1) \vee R(x+2)]$
  - for every natural number, either itself or one of its two successors is divisible by 3
  - $true$

5. Determine without a calculator.
   - $(79 * 102^7 - 53)\ mod\ 5$
     - $4 * 2^7 - 3$
     - $4 * 3 - 3$
     - 4
   - $(65^3 * 95 - 87 * 29^4)\ mod\ 9$
     - $2^3 * 5 - 6 * 2^4$
     - $8 * 5 - 6 * 7$
     - $4 - 6$
     - 7
   - $(16^{58} * 9 - 48 * 17^3)\ mod\ 14$
     - $2^{58} * 9 - 6 * 3^3$
     - $2^{10*5+8} * 9 - 6 * 3^3$
     - $2^{10^5} * 2^8 * 9 - 6 * 27$
     - $2^5 * 4 * 9 - 6 * 13$
     - $4 - 8$
     - 10

6. Lookup how the International Standard Book Number with ten digits is defined and calculate the checksum of the following one.
    - $3 - 8348 - 0094 - ?$
        - $(1 * 3 + 2 * 8 + 3 * 3 + 4 * 4 + 5 * 8 + 6 * 0 + 7 * 0 + 8 * 9 + 9 * 4)\ mod\ 11$
        - $(3 + 5 + 9 + 5 + 7 + 0 + 0 + 6 + 3)\ mod\ 11$
        - $(38)\ mod\ 11$
        - 5
7. Write a program in Python that uses the following function in order to generate a truth table.
    - $f(A, B, C, D) = A\bar{B} \vee A\bar{C}D \vee ABD \vee \bar{C}\bar{D}$

```
print('A' + '\t' + 'B' + '\t' + 'C' + '\t' + 'D' + '\t' + 'f')

for i in range(0, pow(2, 4)):
    A = bool((i >> 3) % 2)
    B = bool((i >> 2) % 2)
    C = bool((i >> 1) % 2)
    D = bool((i >> 0) % 2)

    f = (A and not B) or (A and not C and D) or (A and B and D) or (not C and not D)

    print(str(A) + '\t' + str(B) + '\t' + str(C) + '\t' + str(D) + '\t' + str(f))

print('A' + '\t' + 'B' + '\t' + 'C' + '\t' + 'D' + '\t' + 'f')
```

8. Write a program in Python that checks whether a given number t is a prime number.
    - I have three possibilities in mind, but there are many more. Since I am only rephrasing each one, feel free to come to my office hours if you want to learn more about them.
    - Try to divide the given number in a loop from 1 to $\sqrt{t}$ and look at the remainders. If all remainders are greater than zero, then the given number is a prime number. This is the easiest and most obvious solution, which I would have expected. Please note that the upper bound of the loop is $\sqrt{t}$, in order to optimize the number of divisions.
    - Using the sieve of Eratosthenes reduces the number of divisions, but requires more space in memory. Feel free to look it up, since it is quite easy to understand and it is a relatively popular algorithm related to prime numbers.
    - The primality teat by Agrawal, Kayal and Saxena would be the most sophisticated one. I have this one in mind, because it is the first primality test that is general, polynomial, deterministic and unconditional. I obviously didn't expect that someone would have come up with this one and you also don't have to understand it.
9. Before smartphones gained popularity, $T9$ has been a usual way to implement a fast way to insert textual content. Think about how you would implement the conversion from the number sequence into strings in Python. You don't need to write the code, I would just like you to think about this problem. This question is by the way not related to the content of the previous week.
    - The solution is actually quite simple. A dictionary is being used, where the number sequences are the keys and their values are arrays that contain the related words. The words within the arrays are furthermore ordered by their likelihood.
    - Even though my described approach would work perfectly fine, it is not necessarily the approach that is being used or has been used. As with every problem, there are most certainly several solutions.