1 Formal Reasoning

Formal Reasoning is a course which treats multiple sections of logic. Propositional logic is one of those sections. Below is a short explanation of how propositional logic works.

There are multiple connectives which are used in the normal English language. English connectives are for example ”and”, ”or”, ”not” and ”if”. Below is a table of how we define these connectives in propositional logic.

$\neg A$	not A
$A \wedge B$	A and B
$A \vee B$	A or B
$A \to B$	If A then B

For this assignment we use the following dictionary, which contains the meaning of multiple symbols:

D	The train is delayed
A	I arrive on time
B	The bus is going
T	I can buy a ticket
M	I have money

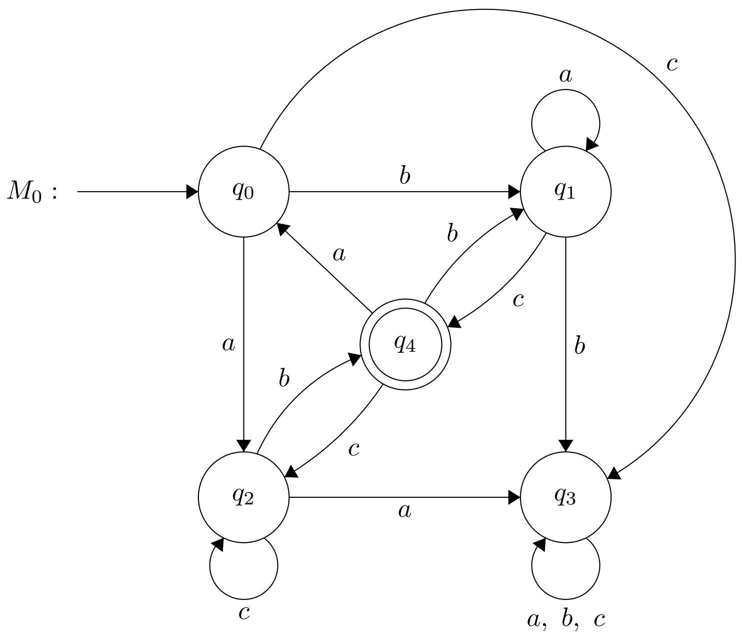
Examples:

$D \to B$	means	If the train is delayed, then the bus is going.
$\neg D \wedge A$	means	The train is not delayed and I arrive on time.
$(T \wedge B) \to A$	means	If I can buy a ticket and the bus is going, then I arrive on time.

1. Translate the following formula to a normal English. $(M \to T) \wedge (T \to A)$

2. Write the following English sentence as a formula: If the train is delayed and the bus is not going, then I do not arrive on time.

The course Formal Reasoning also includes Automata. Such automata can be used to model a language. For example, the following automaton represent such a language. It represents the words that are in a language. To find out whether a word is in the language of the automaton you have to evaluate it letter by letter. So let’s take the word *abcb* and the automaton M_0 (on the next page). Then we start at the initial state q_0, then the first letter is a, and there is an edge from $q_0 \to q_2$ that corresponds to a. Then there is an edge $q_2 \to q_4$ which corresponds to b. Using that logic, we eventually end up in q_4. Since q_4 is a final state (indicated by the double circle), this word is accepted. If the state where you end up, is not a final state, the word is not part of the language corresponding to M_0.



1. Is the word *cabbc* accepted by M_0?
2. Is the word *bccab* accepted by M_0?
3. Is the word *baaac* accepted by M_0?
4. Is the word *acbaba* accepted by M_0?
5. Is the word *bcaab* accepted by M_0?

2 Probability theory

Let's assume there is a highly infectious disease and 1% of the population is infected. You start to feel a bit ill, so you decide to take a test to see whether you have caught the disease. You do know that if you have the disease, there is a 90% chance that you will test positive. And there is a 5% change you will test positive if you do not have the disease.

Calculate the change that you are actually ill if you test positive. Use the following formula for that

$$P(\text{ill} \mid +) = \frac{P(+ \mid \text{ill})P(\text{ill})}{P(+)}$$

and

$$P(+) = P(+ \mid \text{ill})P(\text{ill}) + P(+|\text{not ill})P(\text{not ill})$$

3 Programming

```python
x: int = 6
y: int = 3

if x/y > y:
    print(x + y)
else:
    print(x - y)
```

1. What is the output of this code?
2. What would the output be if y would be 2 instead of 3?

```python
n: int = 1
m: int = 6

for i in range(m):
    print(i * i)
    n = n + i - 1
```

1. What is the output of this code?
2. What is the value of n at the end?

4 Machine Learning

A the core of machine learning and neural networks, there are perceptrons. Perceptrons take a numerical input and return either 0 or 1, based on an activation function. For this exercise we will use the following activation function:

$$f(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

Or in normal English, if the input is 0 or larger, the perceptron returns 1, otherwise it returns 0.

We will be considering the following perceptron:

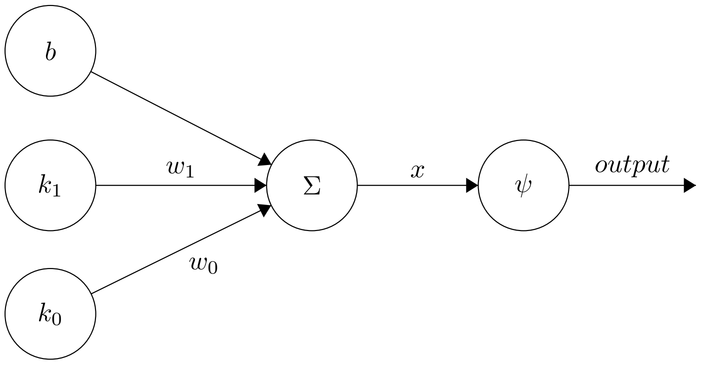


Figure 1: Single layer perceptron

Where k_0 and k_1 are input values, and b is the bias. w_0 and w_1 are the weights corresponding to the inputs. These weights have a value between 0 and 1. They indicate how strong the output depends on the input of the weight. x is calculated in the node denoted as Σ. This calculation can be formulated as follows:

$$x = b + \sum_{i=0} k_i w_i = b + k_0 w_0 + k_1 w_1$$

Or again in normal English, the bias plus the sum of all inputs multiplied with their weight. x then gets passed to ψ where the activation function is applied to get the output.

1. With all that info, compute the values for x and the output given the inputs in the following table:

b	k_0	k_1	w_0	w_1	x	output
1	0	0	1	1	...	...
0.5	1	0	0.80	0.25	...	...
-0.5	1	1	0.9	0.50	...	...
-1	1	0	0.60	0.40	...	...
-1	1	1	0.70	0.25	...	...

2. Given the perceptron from figure 1. Can you set the weights and biases in such a way that you get the following outputs? If not explain why it is not possible. Assume $b = 0$ in all cases.

k_0	k_1	Output
0	0	1
0	1	0
1	0	0
1	1	1

3. What would a perceptron that can do this look like?