# Assignment 1

COMP30026 Models of Computation
School of Computing and Information Systems

*Due:* Friday 30 August at 8:00pm

## Aims

To improve your understanding of propositional logic and first-order predicate logic, including their use in mechanised reasoning; to develop your skills in analysis and formal reasoning about complex concepts, and to practise writing down formal arguments with clarity.

## Marking

Each question is worth 2 marks, for a total of 12. We aim to ensure that anyone with a basic comprehension of the subject matter receives a passing mark. Getting full marks is intended to be considerably more difficult; the harder questions provide an opportunity for students to distinguish themselves.

Your answers will be marked on correctness and clarity. Do not leave us guessing! It is better to be clear and wrong; vague answers will attract few if any marks. This also means you must show your working in mechanical questions!

Finally, make sure your writing is legible! We cannot mark what we cannot read. (Keep in mind that the exam will be on paper, so this will be even more important later!)

## Academic Integrity

In this assignment, ***individual work*** is called for. By submitting work for assessment you declare that:

1. You understand the University's policy on ***academic integrity***.
2. The work submitted is your original work.
3. You have not been unduly assisted by any other person or third party.
4. You have not unduly assisted anyone else.
5. You have not used any unauthorized materials, including ***but not limited to*** AI and translation software.

However, ***if you get stuck***, you can use the discussion board to ask any questions you have. If your question reveals anything about your approach or working, please make sure that it is set to "private".

You may ***only*** discuss the assignment in ***basic terms*** with your peers (e.g. clarifying what the question is asking, or recommending useful exercises). You may ***not*** directly help others in solving these problems, even by suggesting strategies.

Soliciting or accepting further help from non-staff is cheating and ***will*** lead to disciplinary action.

## Q1 Propositional Logic: Island Puzzle

You come across three inhabitants of the Island of Knights and Knaves. Now, a mimic has eaten one of them and stolen their appearance, as well as their status as a knight or knave. (And is thus bound by the same rules. Remember that knights always tell the truth, and knaves always lie!)

Each makes a statement:

1. A says: "C is either the mimic or a knight, or both."
2. B says: "It is not the case that both A is the mimic and C is a knave."
3. C says: "If B is a knight, then the mimic is a knave."

### Task A

Translate the information above into propositional formulas. Give an appropriate interpretation of all propositional letters used. Use the same interpretation throughout the question; do not give multiple interpretations.

### Task B

Determine which of $A$, $B$, and $C$ is the mimic, and prove that it must be the case using an informal argument.

**Some advice:** A good answer should not be much longer than about 250 words. But do not worry about the length of your first draft! Instead focus on finding a proof in the first place. Once you have that, it is *much* easier to find a shorter proof. Also, remember that clarity is key: write in complete sentences with good grammar, but do not include irrelevant information or repeat yourself unnecessarily.

## Q2 Propositional Logic: Validity and Satisfiability

For each of the following propositional formulas, determine whether it is valid, unsatisfiable, or contingent. If it is valid or unsatisfiable, prove it by drawing an appropriate resolution refutation. If it is contingent, demonstrate this with two appropriate truth assignments.

1. $\neg P \wedge (P \to \neg P)$

2. $(P \lor (P \land (Q \rightarrow Q))) \land (\neg P \lor \neg(\neg Q \rightarrow \neg Q))$
3. $\neg((Q \lor P) \rightarrow P) \lor (P \leftrightarrow Q) \lor (P \land \neg Q)$
4. $(R \leftrightarrow P) \rightarrow ((R \rightarrow Q) \leftrightarrow (P \leftrightarrow Q))$

**Hint:** If you are unsure, you can use a truth table to help you decide!

## Q3 Predicate Logic: Translation and Semantics

### Task A

Translate the following English sentences into formulas of predicate logic. Give an appropriate interpretation of any non-logical symbols used. Use the same interpretation throughout this question; do not give multiple interpretations.

1. Iron is heavier than oxygen.
2. All actinides are radioactive.
3. Some, but not all, lanthanides are radioactive.
4. Actinides are heavier than lanthanides.
5. Both lanthanides and actinides are heavier than iron and oxygen.
6. At least three isotopes of lanthanides are radioactive, but the only lanthanide without any non-radioactive isotopes is promethium.

### Task B

By arguing from the semantics of predicate logic, prove that the universe of every model of following formula has at least 3 distinct elements. (Resolution refutations will receive 0 marks.)

$$\forall x \forall y (P(x, y) \rightarrow \neg P(y, x)) \land \forall x \exists y (P(x, y))$$

## Q4 Predicate Logic: Red-Black Trees

The use of function symbols in our notation for predicate logic allows us to create a simple representation of binary trees. Namely, let the constant symbol $a$ represent the root node of the tree, and the unary functions $l$ and $r$ represent the left and right children of a node. The idea is that $l(a)$ is the left child of the root node, $r(l(a))$ is the right child of the left child of the root node, and so on. With this representation defined, we can now prove statements about trees.

A red-black tree is a special type of binary tree that can be searched faster, in which each node is assigned a colour, either red or black. Let the predicates $R$ and $B$ denote whether a node is red or black respectively. A red-black tree is faster to search because it must satisfy some constraints, two of which are:

1. Every node is red or black, but not both:

$$\forall x((R(x) \land \neg B(x)) \lor (B(x) \land \neg R(x))) \tag{1}$$

2. A red node does not have a red child:

$$\forall x(R(x) \to (\neg R(l(x)) \land \neg R(r(x)))) \tag{2}$$

### Task

Use resolution to prove that these two conditions entail that a tree consisting of a non-black root with a red left child is *not* a red-black tree.

## Q5 Informal Proof: Palindromes

Assume the following definitions:

1. A string is a finite sequence of symbols.
2. Given a symbol $a$, we write the string consisting of just $a$ also as $a$.
3. Given strings $x$ and $y$, we write their concatenation as $xy$.
4. Given a collection of $n$ symbols $a_1, \dots, a_n$, we have the following:
   (a) The expression $a_1 \dots a_n$ stands for the string of $n$ symbols whose $i$th symbol is equal to $a_i$ for all integers $i$ from 1 to $n$.
   (b) The reverse of the empty string is the empty string.
   (c) The reverse of a nonempty string $a_1 \dots a_n$ of length $n$ is the string $b_1 \dots b_n$ where $b_i = a_{n-i+1}$ for all positive integers $i \leq n$.
   (d) The expression $a_n \dots a_1$ stands for the reverse of $a_1 \dots a_n$.
5. A string is a palindrome if and only if it is equal to its reverse.

### Task

The proof attempt below has problems. In particular, it does not carefully argue from these definitions. Identify and describe the problems with the proof. Then, give a corrected proof.

**Theorem.** *Let $s$ be a palindrome. Then $ss$ is also a palindrome.*

*Proof (attempt).* We have $s = a_1 \dots a_n$ for some symbols $a_1, \dots, a_n$ where $n$ is the length of $s$. Since $s$ is a palindrome, it is by definition equal to itself under reversal, so $s = a_n \dots a_1$ and $a_i = a_{n-i+1}$ for all positive integers $i \leq n$.

Therefore $ss = a_1 \dots a_n a_n \dots a_1$, and hence there exist symbols $b_1, \dots, b_{2n}$ such that $ss = b_1 \dots b_{2n}$. Since the reverse of $a_1 \dots a_n a_n \dots a_1$ is itself, it follows that $ss$ is a palindrome, as desired. □

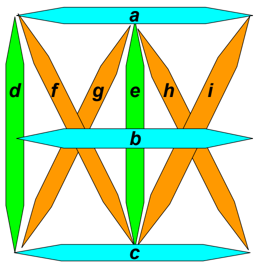

Figure 1: Diagram of our 9-segment display. Colour key: horizontal segments are blue, vertical segments are green, and diagonal segments are orange.

## Q6 Propositional Logic: Logic on Display

One common practical application of propositional logic is in representing logic circuits. Consider a 9-segment LED display with the segments labelled $a$ through $i$, like the one shown on Figure 1. To display the letter "E", for example, you would turn on LEDs $a$, $b$, $c$, and $d$, and turn the rest off.

Arrays of similar displays are commonly used to show numbers on digital clocks, dishwashers, and other devices. Each LED segment can be turned on or off, but in most applications, only a small number of on/off combinations are of interest (e.g. displaying a digit in the range 0–9 only uses 10 combinations). In that case, the display can be controlled through a small number of input wires.

For this question, we are interested in creating a display for eight symbols from the proto-science of alchemy. Since we only want eight different symbols (see Figure 2), we only need three input wires: $P$, $Q$, and $R$.

| symbol | P | Q | R | a | b | c | d | e | f | g | h | i |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| △: Fire | **0** | **0** | **0** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| ▽: Water | **0** | **0** | **1** | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 🜁: Air | **0** | **1** | **0** | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 🜃: Earth | **0** | **1** | **1** | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 🝗: Ashes | **1** | **0** | **0** | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| 🜊: Vinegar | **1** | **0** | **1** | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 🜿: Tartar | **1** | **1** | **0** | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |
| 🝣: Crucible | **1** | **1** | **1** | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

Figure 2: Table of symbols, their encodings in terms of $P$, $Q$ and $R$, and the corresponding on/off state of the segments $a$–$i$.

So, for example, △ is represented by $P = Q = R = \mathbf{0}$, and so when all three wires are unpowered, we should turn on segments $c$, $g$ and $h$ and turn off the other segments. Similarly, ▽ is represented by $P = Q = \mathbf{0}$ and $R = \mathbf{1}$, so when wires $P$ and $Q$ are off and the wire $R$ is on, we should turn on $a$, $f$ and $i$, and turn off the other segments.

Note that each of the display segments *a*–*i* can be considered a propositional function of the variables $P$, $Q$, and $R$. For example, segment $e$ is on when the input is one of 101, 110, or 111, and is off otherwise. That is, we can capture its behavior as the following propositional formula:

$$(P \wedge \neg Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (P \wedge Q \wedge R).$$

The logic display must be implemented with logic circuitry. Here we assume that only three types of logic gates are available:

1. An *and-gate* takes two inputs and produces, as output, the conjunction ($\wedge$) of the inputs.
2. An *or-gate* implements disjunction ($\vee$).
3. An *inverter* takes a single input and negates ($\neg$) it.

## Task

Design a logic circuit for each of *a*–*i* using as few gates as possible. Your answer does *not* need to be optimal[1] to receive full marks, but it must improve upon the trivial answer. (Incorrect answers will receive 0 marks.)

We can specify the circuit by writing down the Boolean equations for each of the outputs *a*–*i*. For example, from what we just saw, we can define

$$e = (P \wedge \neg Q \wedge R) \vee (P \wedge Q \wedge \neg R) \vee (P \wedge Q \wedge R)$$

and thus implement $e$ using 10 gates. But the formula $(P \wedge \neg Q \wedge R) \vee (P \wedge Q)$ is equivalent, so we can in fact implement $e$ using 5 gates.

Moreover, the nine functions might be able to share some circuitry. For example, if we have a sub-circuit $u$ defined by $u = \neg P \wedge Q$, then we can define $d = u \vee (P \wedge \neg Q \wedge \neg R)$, and also possibly reuse $u$ in other definitions. That is, we can share sub-circuits among multiple functions. This can allow us to reduce the total number of gates. You can define as many "helper" sub-circuits as you please, to create the smallest possible solution.

### Submission

Go to "Assignment 1 (Q6)" on Gradescope, and submit a text file named `q6.txt` consisting of one line per definition. This file will be tested automatically, so it is important that you follow the syntax exactly.

We write $\neg$ as `-` and $\vee$ as `+`. We write $\wedge$ as `.`, or, simpler, we just leave it out, so that concatenation of expressions denotes their conjunction. Here is an example set of equations (for a different problem):

```
# An example of a set of equations in the correct format:
a = -Q R + Q -R + P -Q -R
b = u + P (Q + R)
c = P + -(Q R)
d = u + P a
u = -P -Q
# u is an auxiliary function introduced to simplify b and d
```

[1]Indeed, computing an optimal solution to this problem is extremely difficult!

Empty lines, and lines that start with '#', are ignored. Input variables are in upper case. Negation binds tighter than conjunction, which in turn binds tighter than disjunction. So the equation for $a$ says that $a = (\neg Q \land R) \lor (Q \land \neg R) \lor (P \land \neg Q \land \neg R)$. Note the use of a helper function $u$, allowing $b$ and $d$ to share some circuitry. Also note that we do not allow any **feedback loops** in the circuit. In the example above, $d$ depends on $a$, so $a$ is not allowed to depend, directly or indirectly, on $d$ (and indeed it does not).