The Australian National University (compiled: 16:12, Friday 7 October) **S2 2022**
School of Computing Assignment 3

Convenor: Pascal Bercher
Lecturer of the respective content: Pascal Bercher

Foundations of Computation

Released:	Friday, 7 October, 2022
Due:	Friday, 21 October, 2022, 23:55 AEST
Mode:	Individual submissions only
Submit:	As single PDF file via Wattle. Individual questions might require turnitin. In this case, the respective exercise mentions the turnitin requirement. Also note that turnitin submissions *must* be typed. Scans of any form will score zero. Not submitting it to turnitin (but Wattle) will also score zero. Turnitin submissions are not allowed to contain the cover page or the exercises.
Note:	**Signatures will only be accepted if they are handwritten and not typed.** Acceptable signatures include those done by hand or electronically using a tablet. Don't use "PDF annotations" for signatures since they cannot be displayed by Wattle.

Pledge of Academic Integrity and Originality statement[1]

I am committed to being a person of i
University community, to a
ANU statement on honesty and plagiarism. I understand that it is wrong to ever misrepresent another person's work as my own.

I am aware of the relevant legislation, and understand the consequen
have read the COMP1600/COMP6260 academic integrity and ple

I declare that everything I have submitted in this assignment was entirely my own work.

Name: ____________________________

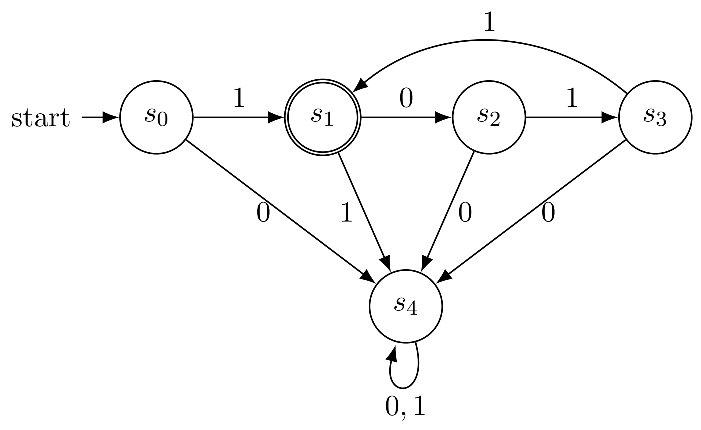
UID: ____________________________

Signature: ____________________________

[1]Submissions without your name, UID and signature will not be marked.

Question 1 **DFA Languages** [20 + 5 Credits]

Consider the below DFA D:



Also consider the language

$$\mathcal{L} = \{(101)^n 1 \mid n \geq 0\}.$$

a) Prove that $L(D) = \mathcal{L}$ as follows:

   i) Clearly state your two proof obligations with necessary quantifiers

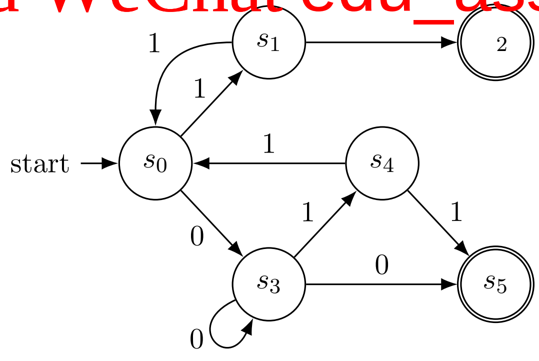
   ii) Prove both obligations

   Make sure you give your proof in full detail, and justify any use of the append theorem or other lemmas/assumptions.

b) Give a right linear grammar

Question 2 **FSA Conversions** [10 + 10 Credits]

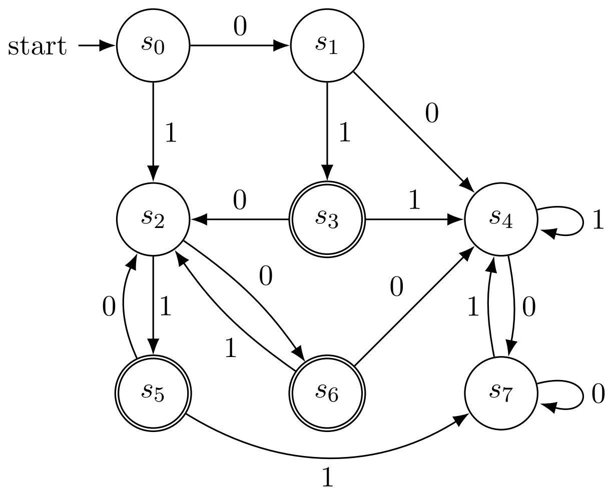
a) Consider the below NFA A:



Convert A into a DFA using the subset construction algorithm from lectures. You do not need to compute the entire table, only reachable subsets. Do *not* perform the minimization afterwards.

You do not need to draw the DFA if all the relevant information is clear in the table. To be clear: You *must* provide the table. (Also drawing it in a graph makes it easier for us, but it's not required.)

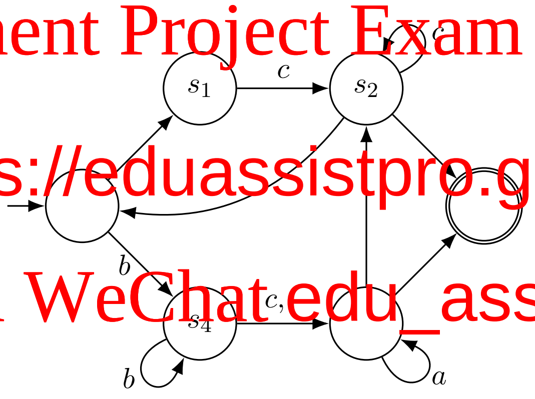
b) Consider the below DFA B:



Identify any equivalent states in B using the minimisation algorithm from lectures. Show your working and clearly state both why the algorithm terminates, and which states, if any, are equivalent. You do not need to draw the minimal DFA.

Question 3 **ϵ-NFA to NFA** [10 Credits]

Convert the following ε-NFA to an NFA using the algorithm presented in the lectures. Ensure you carefully follow the algorithm and explicitly state the ϵ-closure of each state.



Question 4 **Regular Expressions** [5 + 5 Credits]

Consider our alphabet to be $\Sigma = \{a, b, c\}$.

a) Give a regular expression that generates all strings in Σ^* that contain exactly one c, and after that c the substring ab does not appear.

b) Are the regular expressions $R_1 = (a|b)c^*(ac)^* \mid \epsilon$ and $R_2 = (b|\epsilon)(a|c)^*$ equivalent? (Being equivalent means that they genrate the same language.) Clearly state your answer and:

- If they are equivalent, write down the language in formal set notation and provide five words of different length that are part of this language.generated.
- If they are not equivalent, give a word w that is generated by only one regular expression and briefly explain how it is generated, and why it can't be generated by the other.

Question 5 **CFGs and PDAs** [5 + 10 + 5 Credits]

Consider the below grammar H with $\Sigma = \{$`f`, `(`, `)`, `+`, `.`$\}$ which models function application similar to Haskell. *(Note: type checking is not considered.)*

$$\begin{aligned} \mathrm{E} &\rightarrow \mathrm{E\,E} \mid \mathrm{E\,O\,E} \mid (\mathrm{E}) \mid \mathtt{f} \\ \mathrm{O} &\rightarrow \mathtt{+} \mid \mathtt{.} \end{aligned}$$

a) Demonstrate that H is ambiguous. Your example must use both non-terminals E and O.

b) Modify H so that function application is left-associative and the operators bind most tightly; parentheses in the language should be respected. Your resulting grammar H' must be unambiguous and must still be context-free.

For example, the string `f f f + f . (f f)` should be parsed as [`f f`] [[`f + f`] `. (f f)`]. Note that `()` are parentheses in the language and [] are parentheses not in the language used to illustrate how we parse. Spaces are also added for clarity, but this does not affect the language.

c) Using your grammar H', give a parse tree for the string `f f . f f` and briefly argue why this is the only parse tree for that string. We expect one or two paragraphs (i.e., maybe around 100 words). Note that this is just an estimate; our goal is simply that you convince us that that you understand why there can't be another way to parse the given string.

You should ensure your parse tree follows the conditions in b).

Question 6 **Determinism in PDAs** [15 Credits]

We have seen in lectures that non-deterministic PDAs are strictly more powerful than deterministic PDAs. We have also seen that any CFG can be converted into an NPDA (but not all CFG's can be converted to DPDAs).

In this question you must examine the differences between DPDAs and NPDAs, and the languages each can recognise. As an example you must consider the language $L(H)$ that is generated by the grammar H from Question 5, and explain whether or not $L(H)$ can be recognised by a DPDA.

In your response you should:

- outline the general di

- argue why NPDAs are st
  powerful), and

- discuss features or properties of the language $L($ istic or non-deterministic

You may try to design a DPDA that recognises $L(H)$, but you should not provide your attempt. Instead you should explain either how one could work, or why we cannot build one.

Respond to the prompts above in no more than 400 words (fewer is fine so long as you clearly answer the points above). Your response should be a clear and logical written piece; avoid dot points or formal definitions/proofs unless appropriate. **This question must be typed and submitted to Turnitin.** You will score zero if that's not done.

Also note that not only is collaboration strictly disallowed, but also copying from any other sources, including wikipedia or or any other encyclopedia. Well, you simply can't copy anything from anywhere! Also note that even using some existing text and just rephrasing it in your own words is still plagiarism and thus a breach of academic integrity. Your entire text must scrictly be written by yourself. **Any breach of the academic integrity rules will be investigated and might have serious consequences.** Furthermore, these investigations cost the convenor a *lot* of time that he'd rather invest in other tasks... It's also not fun to cause somebody an entry for Academic Misconduct – so please don't break any rules!