COMP 3803 — Assignment 4

Due: Thursday December 9, 23:59.

Assignment Policy:

- Your assignment must be submitted as one single PDF file through Brightspace.

> Use the following format to name your file:
>
> `LastName_StudentId_a4.pdf`

- **Late assignments will not be accepted. I will not reply to emails of the type "my internet connection broke down at 23:57" or "my scanner stopped working at 23:58", or "my dog ate my laptop charger".**
- You are encouraged to collaborate on assignments, but at the level of discussion only. When writing your solutions, you must do so in your own words.
- Past experience has shown conclusively that those who do not put adequate effort into the assignments do not learn the material and have a probability near 1 of doing poorly on the exams.
- When writing your solutions, you must follow the guidelines below.
  - You must justify your answers.
  - The answers should be concise, clear and neat.
  - When presenting proofs, every step should be justified.

Question 1: Write your name and student number.

Question 2: Construct a Turing machine with one tape that gets as input an integer $x \geq 0$ and returns as output the integer $2x$. Integers are represented in binary.

Start of the computation: The tape contains the binary representation of the input x. The tape head is on the **leftmost** bit of x and the Turing machine is in the start state.

End of the computation: The tape contains the binary representation of the number $2x$. The tape head is on the **leftmost** bit of $2x$ and the Turing machine is in the final state.

The Turing machine in this question does not have an accept state or a reject state; instead, it has a final state. As soon as this final state is entered, the Turing machine terminates. At termination, the contents of the tape is the output of the Turing machine.

Start by explaining your algorithm in plain English, then mention the states that you are going to use, then explain the meaning of these states, and finally give the list of instructions.

Question 3: Construct a Turing machine with two tapes that accepts the language

$$\{a^n b^n : n \geq 0\}.$$

Start of the computation: Tape 1 contains a string w in $\{a, b\}^*$, its head is on the **leftmost** symbol of w (or at an arbitrary position if $w = \epsilon$). Tape 2 is empty (that is, it contains only $\Box$'s) and its head is at an arbitrary position. At the start, the Turing machine is in the start state.

End of the computation: The computation terminates as soon as the Turing machine enters the accept state or the reject state. If the input string w is of the form $a^n b^n$ for some $n \geq 0$, the Turing machine must terminate in the accept state. Otherwise, the Turing machine must terminate in the reject state.

Running time: On any input string w, the number of steps made by the Turing machine must be[1] $O(1 + |w|)$. (This can be achieved by using the second tape!)

Start by explaining your algorithm in plain English, then mention the states that you are going to use, then explain the meaning of these states, and finally give the list of instructions.

Question 4: In class, we have seen that the language

$$\mathit{Halt} = \{\langle P, w\rangle : \ P \text{ is a Java program that terminates on the binary input string } w\}$$

is undecidable.

A Java program P is called a *Hello-World-program*, if the following is true: When given the empty string ϵ as input, P can do whatever it wants, as long as it outputs the string `Hello World` and terminates. (We do not care what P does when the input string is non-empty.)

Consider the language

$$\mathit{HW} = \{\langle P\rangle : \ P \text{ is a Hello-World-program}\}.$$

The questions below will lead you through a proof of the claim that the language HW is undecidable.

(4.1) Consider a fixed Java program P and a fixed binary string w.

We write a new Java program J_{Pw} which takes as input an arbitrary binary string x. On such an input x, the Java program J_{Pw} does the following:

[1] The term "1+" is to deal with the case when $w = \epsilon$.

Algorithm $J_{Pw}(x)$:
run P on the input w;
print `Hello World`

- Argue that P terminates on input w if and only if $\langle J_{Pw} \rangle \in HW$.

(4.2) The goal is to prove that the language HW is undecidable. We will prove this by contradiction. Thus, we assume that H is a Java program that decides HW. Recall what this means:

- If P is a Hello-World-program, then H, when given $\langle P \rangle$ as input, will terminate in the accept state.
- If P is not a Hello-World-program, then H, when given $\langle P \rangle$ as input, will terminate in the reject state.

We write a new Java program H' which takes as input the binary encoding $\langle P, w \rangle$ of an arbitrary Java program P and an arbitrary binary string w. On such an input $\langle P, w \rangle$, the Java program H' does the following:

Algorithm $H'(\langle P, w \rangle)$:
construct the Java program J_{Pw} described above;
run H on the input $\langle J_{Pw} \rangle$;
if H terminates in the accept state
then terminate in the accept state
else terminate in the reject state
endif

Argue that the following are true:

- For any input $\langle P, w \rangle$, H' terminates.
- If P terminates on input w, then H' (when given $\langle P, w \rangle$ as input), terminates in the accept state.
- If P does not terminate on input w, then H' (when given $\langle P, w \rangle$ as input), terminates in the reject state.

(4.3) Now finish the proof by arguing that the language HW is undecidable.

Question 5: Consider the two languages

$$Empty = \{\langle M \rangle : \ M \text{ is a Turing machine for which } L(M) = \emptyset\}$$

and

$$\mathit{UselessState} = \{\langle M, q\rangle: \quad M \text{ is a Turing machine, } q \text{ is a state of } M, \text{ for every input string } w, \text{ the computation of } M \text{ on input } w \text{ never visits state } q\}.$$

(5.1) Use Rice's Theorem to show that *Empty* is undecidable.

(5.2) Use **(5.1)** to show that *UselessState* is undecidable.