

## Quiz #10

Illya Starikov

June 30, 2025

### Problem #1

$$\begin{array}{ll} & S \\ \Rightarrow & AB \quad \text{via } S \rightarrow AB \\ \Rightarrow & aXB \quad \text{via } A \rightarrow aX \\ \Rightarrow & aXbYd \quad \text{via } B \rightarrow bYd \\ \Rightarrow & aXbYd \quad \text{via } B \rightarrow bYd \\ \Rightarrow & abXYd \quad \text{via } Xb \rightarrow bX \\ \Rightarrow & abYcd \quad \text{via } XY \rightarrow Yc \\ \Rightarrow & abcd \quad \text{via } Y \rightarrow \lambda \end{array}$$

## Problem #2

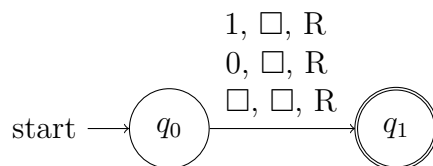
	$S$	
$\Rightarrow$	$TbC$	via $S \rightarrow TbC$
$\Rightarrow$	$gC$	via $Tb \rightarrow g$
$\Rightarrow$	$Sg$	via $gC \rightarrow Sg$
$\Rightarrow$	$TbCg$	via $S \rightarrow TbC$
$\Rightarrow$	$gCg$	via $Tb \rightarrow g$
$\Rightarrow$	$Sgg$	via $gC \rightarrow Sg$
$\Rightarrow$	$TbCgg$	via $S \rightarrow TbC$
$\Rightarrow$	$gCgg$	via $Tb \rightarrow g$
$\Rightarrow$	$egg$	via $gC \rightarrow e$

## Problem #3

The language is derived by the grammar is  $L = \{aad^nb^k \mid 0 \leq n \leq k, n+k = 2c\}$ , for some arbitrary constant  $c$  (i.e.  $n+k$  is even).

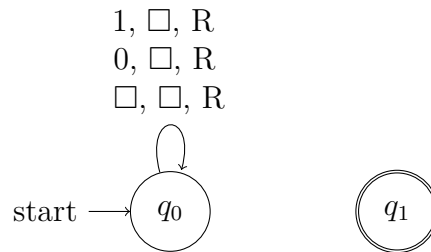
## Problem #4

The following always halts, regardless of the input from  $\Sigma = \{0, 1\}$ .



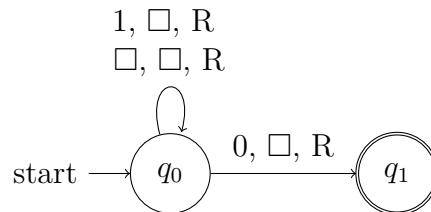
## Problem #5

The following never halts, regardless of the input from  $\Sigma = \{0, 1\}$ . Note it also never halts even if the tape is blank ( $\lambda$ ).



## Problem #6

The following halts for some, but not all, input from  $\Sigma = \{0, 1\}$ . In this case, the only input that would cause the machine to halt would be 0.



## Problem #7

No, we will prove so as follows.

**Theorem 1.** *Suppose  $A$  to be an arbitrary program — that is,  $A \in$  all programs. Then,  $\forall A \in$  all program, there **does not** exists a program  $P$  such that  $P$  can determine if  $A$  will halt **regardless of input**.*

*Proof.* Suppose not. That is, suppose  $\exists P \in$  all programs such that  $P$  can determine if  $\forall A \in$  all programs,  $A$  will halt. Suppose the following to be such program  $P$ , with an additional source code at the end:

```

bool willHalt(program P) { ... }

int recursiveNightmare() {
    if (willHalt(P)) {
        return recursiveNightmare();
    }

    return 42;
}
  
```

If we were to run this program through  $P$ , that is  $P(P)$ , we have the following cases:

**Case #1:** The program  $P$  loops forever. Assuming the program to be correct, this will return false; hence program  $P$  halts on input  $P$ , which is a contradiction (by our **recursive nightmare**).

**Case #2:** The program halts on input  $P$ . Assuming the program to be correct, this will return true; hence program  $P$  runs indefinitely on input  $P$ ; this is a contradiction.

We see this to be a paradox;  $P$  only halts when `willHalt` will not halt, and runs indecently when it does halt. Either way, we see this to be a contradiction. Therefore, there **does not** exists a program  $P$  such that  $P$  can determine if  $A$  will halt **regardless of input**.

□