23,5 + 3.5

PŘÍJMENÍ A JMÉNO: [redacted]

LOGIN STUDENTA: [redacted]

DATUM: 20. května 2025

Zkoušková písemka z předmětu „Úvod do teoretické informatiky“

Doba trvání: **100 minut**

Max. zisk: **70 bodů**

Část 1: Jazyky a automaty

Příklad [1] (9 bodů): Ke každé z následujících bezkontextových gramatik a každému z následujících regulárních výrazů přiřaďte jazyky (z níže uvedeného seznamu jazyků), které jsou generovány danou bezkontextovou gramatikou nebo popsány daným regulárním výrazem. Řešení pište do připravené tabulky, kde ke každé gramatice a každému výrazu napište písmena všech odpovídajících jazyků, popřípadě buňku tabulky proškrtněte, pokud gramatice nebo výrazu neodpovídá žádný jazyk.

(1) $S \longrightarrow A111A$
$A \longrightarrow 0A \mid 0$

(2) $S \longrightarrow 0S \mid S0 \mid 01110$

(3) $S \longrightarrow 0S0 \mid 01110$

(4) $(0+1)^*(01110+10^*10^*1)(0+1)^*$

(5) $0^*011100^*$

(6) $(0+1)^*01110(0+1)^*$

9

(A) $\{w \in \{0,1\}^* \mid w$ obsahuje podslovo 01110 a $|w|_1 = 3\}$

(B) $\{w \in \{0,1\}^* \mid w$ obsahuje podslovo 01110 a $|w|_1 \geq 3\}$

(C) $\{w \in \{0,1\}^* \mid w$ obsahuje podslovo 01110 nebo $|w|_1 \geq 3\}$

(D) $\{0^n01110^m \mid n \geq 0, m \geq 1\}$

(E) $\{0^n1110^n \mid n \geq 1\}$

Řešení:

Gramatika / Reg. výraz	Jazyky
(1)	A, D
(2)	A, D
(3)	E
(4)	C
(5)	A, D
(6)	B

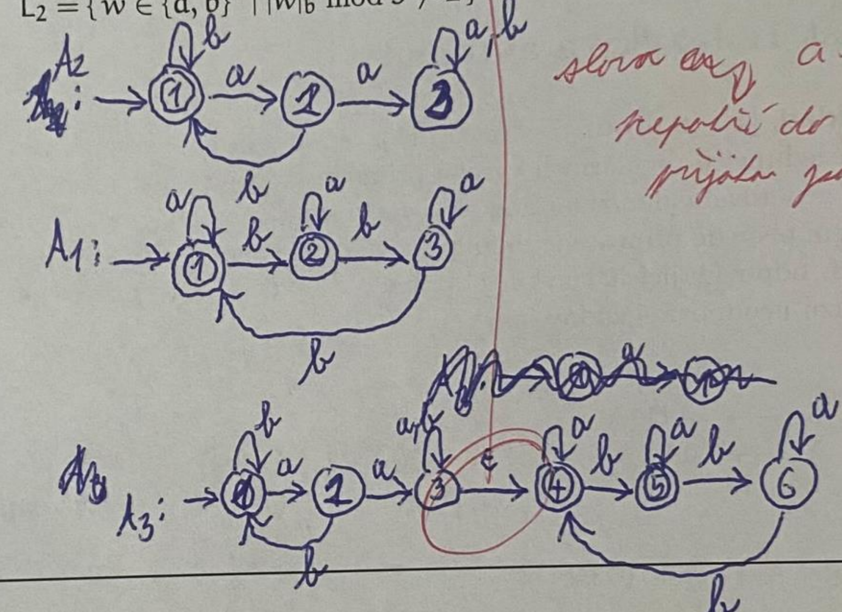
Příklad [2] (9 bodů):

Uvažujme níže uvedené jazyky L_1 a L_2. Zkonstruujte deterministické konečné automaty A_1, A_2 a A_3 tak, že $\mathcal{L}(A_1) = L_1$, $\mathcal{L}(A_2) = L_2$ a $\mathcal{L}(A_3) = L_1 \cap L_2$. Při tvorbě automatu A_3 je výhodné využít standardní konstrukci pro zadanou jazykovou operaci na konečných automatech. Je jedno, jestli budou automaty zadány tabulkou nebo grafem, ale u každého automatu musí být uveden jeho název (tedy A_1, A_2 nebo A_3).

$$L_1 = \{w \in \{a, b\}^* \mid \text{ ve } w \text{ je bezprostředně před každým výskytem symbolu } a \text{ symbol } b\}$$

$$L_2 = \{w \in \{a, b\}^* \mid |w|_b \bmod 3 \neq 2\}$$

4,5



slova ab, abb, ... nepatří do L_1, ale přijala jsem

Příklad [3] (5 bodů):

Vytvořte bezkontextovou gramatiku $\mathcal{G}$ generující jazyk

$$(\mathcal{L}(\mathcal{G}_1) \cdot \mathcal{L}(\mathcal{G}_2)) \cup \mathcal{L}(\mathcal{G}_1)^*$$

(Vypište všechna její pravidla, i když jsou třeba stejná, jako některá ze zadaných.)

$$\begin{array}{lll} \mathcal{G}_1: S & \longrightarrow & abSA \mid B \\ A & \longrightarrow & bA \mid Ba \\ B & \longrightarrow & a \mid Bb \end{array} \qquad \begin{array}{lll} \mathcal{G}_2: S & \longrightarrow & AaabB \mid AA \\ A & \longrightarrow & Aa \mid \varepsilon \\ B & \longrightarrow & bB \mid abb \end{array}$$

$\mathcal{G}$: $L \to L1\,L2 \mid X$

$X \to L1\,X \mid \varepsilon$

$L1 \to ab\,L1\,Y \mid Z$

$Y \to bY \mid Za$

$Z \to a \mid Zb$

$L2 \to CaabD \mid CC$

$C \to Ca \mid \varepsilon$

$D \to bD \mid abb$

5

Příklad [4] (6 bodů): Uvažujme následující jazyky ($|w|$ označuje délku slova w, $|w|_a$ označuje počet výskytů symbolu a ve w; w^R značí zrcadlový obraz slova w):

$$L_1 = \{ a^n b^{2n} c^{3n} \mid n \geq 0 \}$$
$$L_2 = \{ w \in \{a, b, c\}^* \mid |w|_a \geq |w|_b \text{ nebo } |w|_b \geq |w|_c \}$$
$$L_3 = \{ w \in \{a, b, c\}^* \mid w \text{ neobsahuje podslovo } bac \}$$
$$L_4 = \{ 0^i 1^j 0^k \mid i = k + 1,\ j \neq 3 \}$$
$$L_5 = \mathcal{L}(a^* abba(a + b)^*)$$
$$L_6 = \{ u001u^R \mid u \in \{0, 1\}^* \}$$

4

Do následující tabulky napište, jaké jsou odpovědi na otázky v řádcích pro jednotlivé jazyky určené sloupcem (místo „ano“ a „ne“ pište „A“ a „N“).

	L_1	L_2	L_3	L_4	L_5	L_6
Je regulární?	N	N	A	N	A	N
Je bezkontextový?	A	A	A	N	A	A

Příklad [5] (6 bodů): Přesně popište, jak lze obecně k libovolnému deterministickému konečnému automatu $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ sestrojit bezkontextovou (resp. regulární) gramatiku $\mathcal{G} = (\Pi, \Sigma, S, P)$ takovou, že

$$\mathcal{L}(\mathcal{G}) = \mathcal{L}(\mathcal{A}) \,.$$

Pro každý z následujících tří prvků ze čtveřice určující gramatiku $\mathcal{G}$ tedy uveďte, jak jej získáme z pětice určující konečný automat $\mathcal{A}$.
(Množina terminálů Σ bude stejná jako vstupní abeceda automatu $\mathcal{A}$.)

- Množina neterminálních symbolů Π:

Pomocí Q.

- Počáteční neterminál S:

~~Pomocí počátečního~~ Pomocí počátečního stavu v DKA (q_0).

1

- Množina přepisovacích pravidel P:

~~Pravidla~~ Přepisovací pravidla P získáme z DKA pomocí jednotlivých přechodů (δ) mezi stavy.

jak?

Část 2: Vyčíslitelnost a složitost

Příklad [6] (8 bodů):

$R_3 := 4$ 1
$R_1 :=$ READ () 2
$R_2 := R_3$ 3
$R_0 := 1$ 4
$L_1 : [R_2] := R_0$ 5
$R_2 := R_2 + 1$ 6
if $(R_1 \leq 0)$ **goto** L_2 7
$R_1 := R_1 - 1$ 8
$R_0 := R_0 + R_0$ 9
$R_0 := R_0 + 1$ 10
goto L_1 11
$L_2 :$ **if** $(R_2 = R_3)$ **goto** L_3 12, 17
$R_2 := R_2 - 1$ 13
$R_0 := [R_2]$ 14
WRITE (R_0) 15
goto L_2 16
$L_3 :$ **halt**

~~$R_3=4, R_1=8, R_2=4, R_0=1$~~ 7 5 3

$[R_2]=1, 2, 1$ 2 3 1

$R_3=4, R_1=8, R_2=4, R_0=1$
7

a) Do tabulky uveďte, kolik kroků (tj. provedení instrukcí) vykoná uvedený RAM, bude-li na vstupu číslo 8 a jaký výstup RAM vydá pro tento vstup:

počet kroků	výstup
30	

b) Do následující tabulky vypište obsah paměti uvedeného RAMu po provedení 10 kroků a obsah paměti po skončení výpočtu (bylo-li vstupem číslo 8).

adresa paměti	R_0	R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9	R_{10}	R_{11}	R_{12}
obsah po 10 krocích	3	7	~~1~~2	4	1	–	–	–	–	–	–	–	–
obsah po skončení					–	–	–	–	–	–	–	–	–

c) Uveďte kolik kroků maximálně provede výše uvedený stroj RAM, pokud dostane jako vstup číslo n, kde $n \geq 0$ (určete tento počet kroků co nejpřesněji, ne jen asymptotický odhad):

0.5

d) Uveďte, co bude výstupem, pokud na vstupu bude číslo n, kde $n \geq 0$:

Příklad [7] (8 bodů): Pro následující algoritmus určete co nejpřesněji jeho časovou složitost v nejhorším případě a vyjádřete ji pomocí asymptotické notace (nejlépe jako jeden odhad vyjádřený pomocí Θ, případně jako dva odhady vyjádřené pomocí O a Ω). Jako velikost vstupu uvažujte hodnotu uloženou v proměnné n.

Odvozené výsledky **zdůvodněte**.

Poznámka: Předpokládejte, že A, B a C jsou pole, jejichž prvky jsou celá čísla, a že tato pole jsou indexována od nuly. Pole A a B mají n prvků, zatímco pole C má $2n$ prvků.

```
COMPUTE(A, B, C, n):
  i := 0; j := 0; k := 0
  while TRUE do
    if i < n then
      if (j ≥ n) or (j < n and A[i] ≤ B[j]) then
        C[k] := A[i]; i := i + 1
      else
        C[k] := B[j]; j := j + 1
    else if j < n then
      C[k] := B[j]; j := j + 1
    else
      return
    k := k + 1
```

V nejhorším případě tento algoritmu má ~~...~~ časovou složitost $\Theta(n^2)$, ~~..~~ kvůli tomu že vnější cyklus ~~..~~ má složitost $\Theta(n)$, větvení má jak ~~...~~ if a else, kde je složitost $\cancel{\Theta(n^2)}$, kvůli else.

0

Příklad [8] (5 bodů): V následující tabulce vyznačte, které ze vztahů typu $f \in O(g)$, $f \in \Omega(g)$, $f \in \Theta(g)$, $f \in o(g)$ a $f \in \omega(g)$ platí, a které ne.

Vztahy, které platí, **zakroužkujte**, a vztahy, které neplatí, **přeškrtněte**.

$f_1 < f_3 < f_2$

$f_1(n) = \cancel{658}n^2 \log n$ $\qquad$ $f_2(n) = 2\cancel{00}n^3 - 2\cancel{00}n^2$ $\qquad$ $f_3(n) = \cancel{658}n^2 + n \log n$

2

$f_1 \in O(f_2)$	~~$f_2 \in O(f_1)$~~	$f_1 \in O(f_3)$	~~$f_3 \in O(f_1)$~~	~~$f_2 \in O(f_3)$~~	$f_3 \in O(f_2)$
~~$f_1 \in \Omega(f_2)$~~	$f_2 \in \Omega(f_1)$	~~$f_1 \in \Omega(f_3)$~~	$f_3 \in \Omega(f_1)$	$f_2 \in \Omega(f_3)$	~~$f_3 \in \Omega(f_2)$~~
~~$f_1 \in \Theta(f_2)$~~	~~$f_2 \in \Theta(f_1)$~~	~~$f_1 \in \Theta(f_3)$~~	~~$f_3 \in \Theta(f_1)$~~	~~$f_2 \in \Theta(f_3)$~~	~~$f_3 \in \Theta(f_2)$~~
$f_1 \in o(f_2)$	~~$f_2 \in o(f_1)$~~	$f_1 \in o(f_3)$	~~$f_3 \in o(f_1)$~~	~~$f_2 \in o(f_3)$~~	$f_3 \in o(f_2)$
~~$f_1 \in \omega(f_2)$~~	$f_2 \in \omega(f_1)$	~~$f_1 \in \omega(f_3)$~~	$f_3 \in \omega(f_1)$	$f_2 \in \omega(f_3)$	~~$f_3 \in \omega(f_2)$~~

$f_1 = n \log_2 n$

$f_2 = n^3$

$f_3 = n^2$

Příklad [9] (8 bodů): Navrhněte a podrobně popište (nejlépe pomocí pseudokódu) algoritmus řešící následující problém. V případech, kdy není očividné, že vámi navržený algoritmus skutečně řeší daný problém, uveďte nějaké alespoň neformální zdůvodnění korektnosti tohoto algoritmu.

Uveďte, jakou má tento algoritmus časovou a prostorovou složitost. Vaše odhady složitosti **zdůvodněte**.

VSTUP: Orientovaný graf $G = (V, E)$, kde V je množina vrcholů a E množina hran.

OTÁZKA: Existuje v grafu G hamiltonovský cyklus?

Poznámka: Hamiltonovský cyklus je cyklus procházející každým vrcholem právě jednou.

0.5 ?

Jednalo by se o kód, kde by se kontrolovalo, zda daný V (vrchol) nebyl už navštíven (visited) a snažili by jsme se zjistit, zda hamiltonovský cyklus se tam nachází. Časová složitost by byla $\Theta(n)$, kvůli tomu že by jsme procházeli n vrcholů v množině. Prostorová by byla taktéž $\Theta(n)$.

10a) ~~[illegible]~~ Pokud problém A nelze vyřešit, tak ~~[illegible]~~ problém B bude taky neřešitelný a naopak. ? může být algor. nerozh.

Příklad [10] (6 bodů): Pokud problém A je algor. nerozhodnutelný, tak se může stát že díky tomu bude i problém B

a) Definujte přesně, co to znamená, že problém A je *převeditelný* na problém B.

~~Problém, který převeditelný z problému A na problém B~~, je ~~takový, jestliže víme že problém A je nerozhodnutelný. Pokud problém A je algoritmicky nerozhodnutelný, může to~~ ~~kromě~~ ~~, že díky~~

0.5

b) Řekněme, že o problému A již máme dokázáno, že je algoritmicky nerozhodnutelný. Existenci jakého převodu mezi problémy A a B by bylo třeba dokázat, pokud bychom chtěli ukázat, že i problém B je algoritmicky nerozhodnutelný?

Vaši odpověď podrobně **zdůvodněte**.

~~[illegible]~~ Jestliže máme problém A algor. nerozhodnutelný, tak stačí aby problém B ~~[illegible]~~ byl závislý na problému A, ~~pak~~ pak by byl také algor. nerozhodnutelný.

?