		Departamento de Informática -	**2da PROVA**
			P 7,0
Disciplina: **Teoria da Computação**		**Curso: CIÊNCIA DA COMPUTAÇÃO**	**T**
Código 5607.5	**Carga Horária**: 60 horas	**Créditos**: 4.0.0	**NOTA**
Professor: Luciano Reis Coutinho		**Email**: l uciano.rc@ufma.br	

2da Avaliação Data: 03 de janeiro 2022 Aluno :

______________________________________ Código: _________________

INSTRUÇÕES

- A prova deve ser realizada INDIVIDUALMENTE. Respostas iguais ocorrendo em provas de alunos diferentes são passíveis de anulação. ✧ Cada questão consiste em um enunciado e um conjunto de requisitos que uma resposta aceitável deve satisfazer. Respostas dadas que não atendam aos requisitos podem em última instância ser completamente desconsideradas durante a correção da prova. Tenham sempre em mente os requisitos ao dar as suas respostas.
- A interpretação das questões faz parte da avaliação. Caso ache um enunciado ambíguo ou impreciso escreva na folha de resposta sua interpretação e a correspondente resposta. Todas as questões devem ser interpretadas tendo em vista que foi discutido nas aulas de Teoria da Computação.
- Todas as questões devem ser respondidas em arquivo .DOC ou PDF. Ao final, tanto o arquivo de questões quanto o arquivo de respostas devem ser enviados via SIGAA.
- **O tempo total de prova é de 100 min**. **Início**: 14:00, **término**: 15:40. **Limite** para submissão via SIGA: 16:00.

QUESTÕES

1. **(2,0 pontos)** Considerando a codificação de programas monolíticos como números naturais que foi discutida durante as aulas (a codificação de Gödel), MOSTRE PASSO a PASSO como o programa iterativo abaixo --- após traduzido para a forma monolítica --- pode ser representado por meio de um único número natural.

até T **faça** (F) ; G

Resposta:

Programa Monolítico
1: se T então vp 2 senão vp 3
2: faça F vp 1
3: faça F vp 0

1:(1, 0, 2, 3)
2:(0, 0, 1, 1)
3:(0, 1, 0, 0)

onde T -> 0, F -> 0, G -> 1

1:cod(1, 0, 2, 3) = 2^1 * 3^0 * 5^2 * 7^3 = 2 * 25 * 343 = 17150
2:cod(0, 0, 1, 1) = 2^0 * 3^0 * 5^1 * 7^1 = 35

3:cod(0, 1, 0, 0) = 2^0 * 3^1 * 5^0 * 7^0 = 3

cod(P) = cod(cod(1:), cod(2:), cod(3:)
cod(P) = 2^17150 * 3^35 * 5^3

2. **(2,0 pontos)** Escreva uma macro R := **mod**(N,D) para a máquina NORMA que armazena em R o resto da divisão inteira do conteúdo do registrador N pelo conteúdo de D. Lembre-se que em NORMA, apenas as operações de incremento e decremento, e o teste de zero, são definidos. Assim, quaisquer outras operações e testes necessários para escrever R := **mod**(N,D) DEVEM também ser escritos explicitamente na resposta da questão como macros auxiliares. Para facilitar, assuma como já escritas as macros que realizam atribuições.

Resposta:

```
R := mod(N, D)

X := N;
até X<D faça (
    X := X - D;
)
R := X;
```

Macros

- X := X - D usando H

  ```
  H := D;
  até H = 0 faça (
          X := X - 1;
          H := H - 1;
  )
  ```

- X < D usando E, F

  2,0

  ```
  E := X;
  F := D;
  até E=0 faça (
          E := E - 1;
  ```

F := F - 1;
)
(se F=0
  então FALSO
  senão VERD
)

3. **(2,0 pontos)** Dada a tabela de transição abaixo, determine qual a linguagem reconhecida pela **máquina de Turing** correspondente. Justifique sua resposta destacando o alfabeto de entrada Sigma, e apresentando exemplos de aceitação e rejeição de palavras sobre Sigma.

	ℙ	a	b	A	B	β
q0	[q0,ℙ,D]	[q1,A,D]	[q3,B,D]	[q0,A,D]	[q0,B,D]	[q4,β,D]
q4						
q1		[q1,a,D]	[q2,B,E]	[q1,A,D]	[q1,B,D]	
q2	[q0,ℙ,D]	[q2,a,E]	[q2,b,E]	[q2,A,E]	[q2,B,E]	
q3		[q2,A,E]	[q3,b,D]	[q3,A,D]	[q3,B,D]	

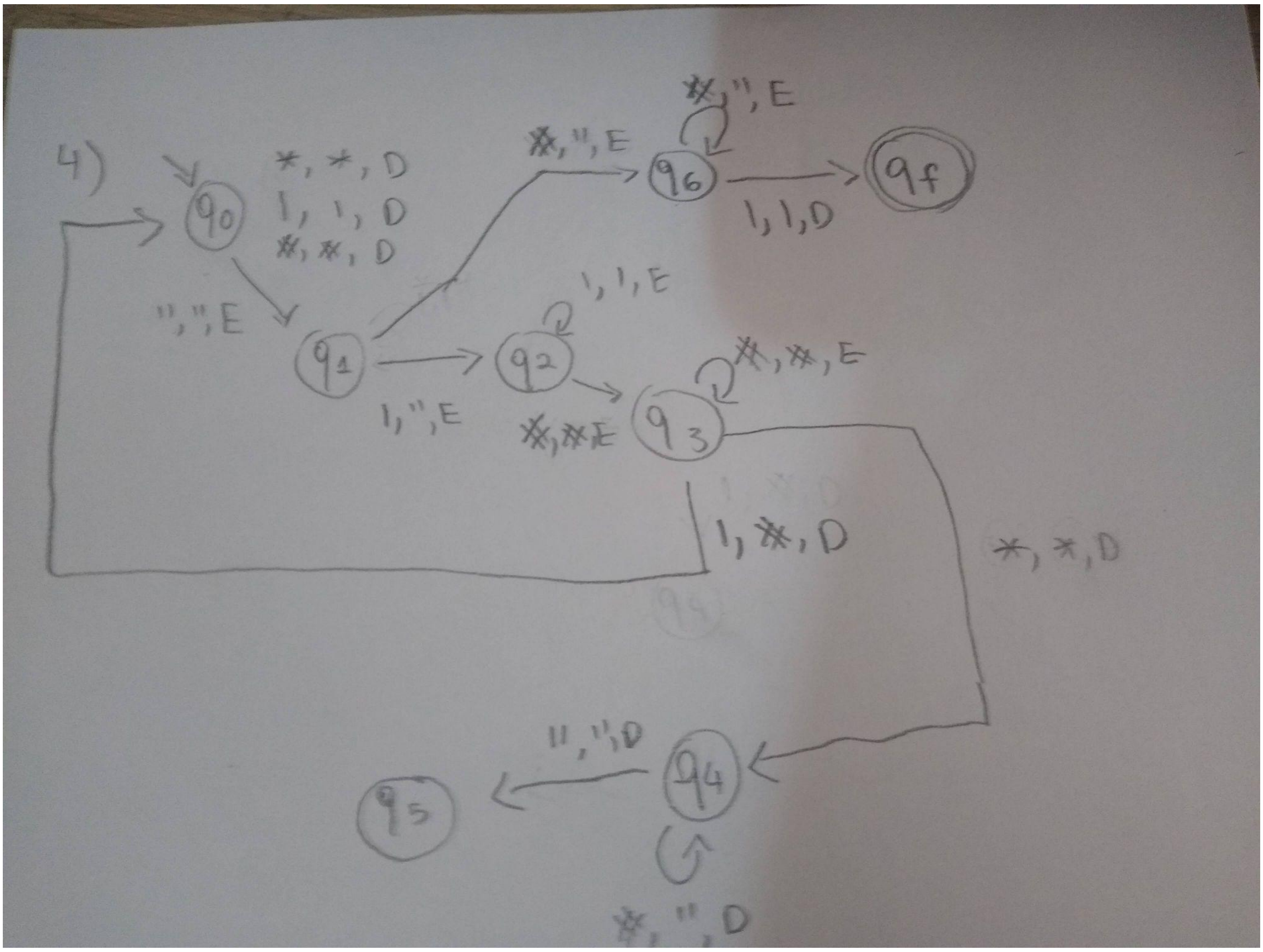
4. **(2,0 pontos)** Escreva uma MÁQUINA DE TURING = <Σ, Q, Π, qo, F, V, β, **❚** > que realize a operação de subtração entre dois números naturais, caracterizada por:

x - y , se x > y
sub (x, y) = {
0 , caso contrário

Resposta:
M = <Σ, Q, Π, qo, F, V, β, b >
M = <{1, #}, {q0, q1, q2, q3, q4, q5, q6, qf}, Π, q0, {qf}, {}, ‘ ’, *>

Observações:
- A máquina opera sobre um sistema unário, utilizando o símbolo # para definir o fim do número X e o começo do número Y.
- O estado q5 identifica 0, caso Y > X, também acarretando na fita vazia (salvo o símbolo de começo de fita).

2,0



5. (a) (**1,0 ponto**) Explique em que consiste a HIPÓTESE de CHURCH-TURING ?
(b) (**0,5 ponto**) A HIPÓTESE de CHURCH-TURING não pode ser provada. Explique por que! (c) (**0,5 ponto**) No entanto, ela pode ser justificada. Explique como!

Resposta:

a) A Hipótese de Church-Turing é uma regra não-verificável, idealizada por Alonzo Church, que diz que qualquer dispositivo computacional terá uma capacidade de computação máxima igual à Máquina de Turing. Em outras palavras, de acordo com a hipótese, é impossível criar qualquer tipo de artifício ou dispositivo que utilize de meios computacionais em que a capacidade computacional máxima ultrapasse a máquina elaborada por Turing. Dessa definição, também se deriva a ideia de funções computáveis e não-computáveis, já que qualquer problema que não possa ser resolvido com a Máquina de Turing é um problema que foge a computação em termos de tempo e espaço.

b) Entretanto, esta “regra” continua sendo apenas uma hipótese por não ser possível verificá-la ou prová-la, já que a noção de algoritmos e função computável é intuitiva. Isso se dá pela noção de algoritmos não ser matematicamente formal, sendo criar uma conjuntura que prove que a Máquina de Turing é a mais genérica possível.
c) A hipótese é justificada pelo fato de que, na época em que foi formulada, diversos trabalhos que elaboraram máquinas e dispositivos teóricos com a função de computar algoritmos terem esforço igual ou menor à Máquina de Turing.

2,0

Boa Sorte!