

Teoria da Computação

Nome: _____

Número: _____

Segundo Semestre 2016/2017

Mini-teste 2 - G

4/4/2017

Duração: 15 Minutos

Classificar (Sim/Não) _____

Este enunciado tem 4 páginas (incluindo esta) e 6 questões.

Apenas voltar a página quando o professor assim o disser. A folha de respostas múltiplas está anexa a este enunciado. Qualquer pergunta errada desconta 1/3 do seu valor no total da pontuação obtida com as respostas certas.

Tabela de Pontuação

Question	Points	Score
1	10	
2	10	
3	20	
4	20	
5	20	
6	20	
Total:	100	

Uma empresa Multinacional, representada por um nome comercial, gere os Funcionários (univocamente identificados pelo seu nome), sede e todas as suas sucursais. De um modo simplista a sede tem uma morada e um funcionário especial, que é o CEO. Cada sucursal tem um nome, os seus funcionarios, de todos um é o diretor financeiro e outro é o administrador. Adicionalmente, cada sucursal tem os vários departamentos.

Para as perguntas seguintes admita as seguintes definições:

$FUNCIONARIO \stackrel{\text{def}}{=} STRING$

$MORADA \stackrel{\text{def}}{=} STRING$

$NOME \stackrel{\text{def}}{=} STRING$

$DEPARTAMENTO \stackrel{\text{def}}{=} \mathcal{P}(PROGRAMADOR) \times \mathcal{P}(CHEFEPROJETO) \times \mathcal{P}(PROJETO)$

$PROGRAMADOR \stackrel{\text{def}}{=} SENIORIDADE \times ID \times NOME \times \mathcal{P}(TAREFA)$

$CHEFEPROJETO \stackrel{\text{def}}{=} NOME \times TELEFONE$

$$PROJETO \stackrel{\text{def}}{=} \mathcal{P}(TAREFA) \times NOME$$

$$TAREFA \stackrel{\text{def}}{=} ID \times TITULO \times CHEFEPROJETO \times SENIORIDADE$$

$$TITULO \stackrel{\text{def}}{=} STRING$$

$$ID \stackrel{\text{def}}{=} NAT$$

$$TELEFONE \stackrel{\text{def}}{=} NAT$$

$$SENIORIDADE \stackrel{\text{def}}{=} \{1, 2, 3, 4, 5\}$$

1. (10 points) Escolha a opção que melhor modela a situação descrita anteriormente para o sistema da Multinacional :

A. $MULTI \stackrel{\text{def}}{=} \mathcal{P}(FUNCIONARIO) \times NOME \times SEDE \times \mathcal{P}(SUCURSAL)$
 $SEDE \stackrel{\text{def}}{=} MORADA \times FUNCIONARIO$

B. $MULTI \stackrel{\text{def}}{=} FUNCIONARIO \times NOME \times SEDE \times SUCURSAL$
 $SEDE \stackrel{\text{def}}{=} MORADA \times FUNCIONARIO$

C. $MULTI \stackrel{\text{def}}{=} \mathcal{P}(FUNCIONARIO) \times \mathcal{P}(NOME) \times \mathcal{P}(SEDE) \times \mathcal{P}(SUCURSAL)$
 $SEDE \stackrel{\text{def}}{=} \mathcal{P}(FUNCIONARIO)$

D. $MULTI \stackrel{\text{def}}{=} \mathcal{P}(FUNCIONARIO) \times NOME \times SEDE \times \mathcal{P}(SUCURSAL)$
 $SEDE \stackrel{\text{def}}{=} MORADA \times \mathcal{P}(FUNCIONARIO)$

E. Nenhuma das anteriores.

2. (10 points) Tendo por base as definições de conjuntos anteriores, escolha a opção que melhor modela a situação descrita anteriormente para SUCURSAL:

A. $SUCURSAL \stackrel{\text{def}}{=} NOME \times DEPARTAMENTOS \times FUNCIONARIOS$

B. $SUCURSAL \stackrel{\text{def}}{=} NOME \times \mathcal{P}(DEPARTAMENTO) \times \mathcal{P}(FUNCIONARIO) \times$
 $FUNCIONARIO \times FUNCIONARIO$

C. $SUCURSAL \stackrel{\text{def}}{=} \mathcal{P}(NOME) \times \mathcal{P}(DEPARTAMENTO) \times \mathcal{P}(FUNCIONARIO)$

D. $SUCURSAL \stackrel{\text{def}}{=} NOME \times DEPARTAMENTO \times FUNCIONARIO \times$
 $FUNCIONARIO \times FUNCIONARIO$

E. Nenhuma das anteriores.

3. (20 points) Defina a função *programadoresProjeto* que retorna os nomes dos programadores do departamento envolvidos em determinado Projeto com certo nome de código.

- A. $programadoresProjeto(n, d) \stackrel{\text{def}}{=} \exists p(p \in \pi_3(d) \wedge n = \pi_2(p) \wedge cn = \{nap \in NOME \mid \exists ap(ap \in \pi_1(d) \wedge nap = \pi_3(ap) \wedge \exists t(t \in \pi_1(p) \wedge t \in \pi_4(ap)))\})$
- B. $programadoresProjeto \in NOME \times DEPARTAMENTO \rightarrow \mathcal{P}(PROGRAMADOR)$
 $programadoresProjeto \stackrel{\text{def}}{=} \{(n, d) \mapsto apl \mid \exists p(p \in \pi_3(d) \wedge n = \pi_2(p) \wedge apl = \{ap \in PROGRAMADOR \mid \exists a(a \in \pi_1(d) \wedge na = \pi_3(a) \wedge \exists t(t \in \pi_1(p) \wedge t \in \pi_4(a)))\})\}$
- C. $programadoresProjeto \in PROJETO \times DEPARTAMENTO \rightarrow \mathcal{P}(NOME)$
 $programadoresProjeto \stackrel{\text{def}}{=} \{(projeto, d) \mid \exists p(p \in \pi_3(d) \wedge projeto = \pi_2(p) \wedge cn = \{nap \in NOME \mid \exists ap(ap \in \pi_1(d) \wedge nap = \pi_3(ap) \wedge \exists t(t \in \pi_1(p) \wedge t \in \pi_4(ap)))\})\}$
- D. $programadoresProjeto(n, d) \stackrel{\text{def}}{=} \exists p(p \in \pi_3(d) \wedge n = \pi_2(p) \wedge cn = \{nap \in PROGRAMADOR \mid \exists ap(ap \in \pi_1(d) \wedge nap = ap \wedge \exists t(t \in \pi_1(p) \wedge t \in \pi_4(ap)))\})$
- E.** $programadoresProjeto \in NOME \times DEPARTAMENTO \rightarrow \mathcal{P}(NOME)$
 $programadoresProjeto \stackrel{\text{def}}{=} \{(n, d) \mapsto cn \mid \exists p(p \in \pi_3(d) \wedge n = \pi_2(p) \wedge cn = \{nap \in NOME \mid \exists ap(ap \in \pi_1(d) \wedge nap = \pi_3(ap) \wedge \exists t(t \in \pi_1(p) \wedge t \in \pi_4(ap)))\})\}$

4. (20 points) Defina a função *chefesProjetoDoProgramador* que retorna os nomes dos chefes de projecto que gerem as tarefas em que determinado programador do departamento é suposto realizar.

- A.** $chefesProjetoDoProgramador \in PROGRAMADOR \times DEPARTAMENTO \rightarrow \mathcal{P}(NOME)$
 $chefesProjetoDoProgramador \stackrel{\text{def}}{=} \{(ap, d) \mapsto cn \mid ap \in \pi_1(d) \wedge cn = \{n \in NOME \mid \exists t(t \in \pi_4(ap) \wedge n = \pi_1(\pi_3(t)))\}\}$
- B. $chefesProjetoDoProgramador \in PROGRAMADOR \times DEPARTAMENTO \rightarrow \mathcal{P}(NOME)$
 $chefesProjetoDoProgramador \stackrel{\text{def}}{=} \{(ap, d) \mapsto cn \mid ap \in \pi_1(d) \wedge cn = \{n \in NOME \mid \exists t(t \in \pi_1(ap) \wedge n = \pi_3(\pi_4(t)))\}\}$
- C. $chefesProjetoDoProgramador(ap, d) \stackrel{\text{def}}{=} ap \in \pi_1(d) \wedge cn = \{n \in NOME \mid \exists t(t \in \pi_1(ap) \wedge n = \pi_3(\pi_4(t)))\}$
- D. $chefesProjetoDoProgramador(ap, d) \stackrel{\text{def}}{=} ap \in \pi_1(d) \wedge cn = \{n \in NOME \mid \exists t(t \in \pi_4(ap) \wedge n = \pi_1(\pi_3(t)))\}$
- E. $chefesProjetoDoProgramador \in PROGRAMADOR \times DEPARTAMENTO \rightarrow CHEFEPROJETOS$
 $chefesProjetoDoProgramador \stackrel{\text{def}}{=} \{(ap, d) \mapsto pf \mid ap \in \pi_1(d) \wedge pf = \{n \in CHEFEPROJETO \mid \exists t(t \in \pi_4(ap) \wedge n = \pi_3(t))\}\}$

5. (20 points) Defina a função *inserirTarefa*, que, dado um departamento, insere a tarefa no conjunto de tarefas que o programador (com determinado ID) deverá realizar no departamento. Esta é apenas inserida se a senioridade requerida pela tarefa for igual ou inferior à senioridade que o programador tem.

- A. $inserirTarefa \in DEPARTAMENTO \times TAREFA \times ID \rightarrow DEPARTAMENTO$
 $inserirTarefa = \{(d, t, id) \mapsto d' \mid \exists ap (ap \in \pi_1(d) \wedge \pi_2(ap) = id \wedge \pi_4(t) \leq \pi_1(ap) \wedge d' = (\pi_1(d) \cup \{(\pi_1(ap), \pi_2(ap), \pi_3(ap), \pi_4(ap) \cup \{t\})\}, \pi_2(d), \pi_3(d)))\}$
- B. $inserirTarefa \in DEPARTAMENTO \times TAREFA \times ID \rightarrow DEPARTAMENTO$
 $inserirTarefa = \{(d, t, id) \mapsto d' \mid \exists ap (ap \in \pi_1(d) \wedge \pi_2(ap) = id \wedge d' = (\pi_1(d) \setminus \{ap\} \cup \{(\pi_1(ap), \pi_2(ap), \pi_3(ap), \pi_4(ap) \cup \{t\})\}, \pi_2(d), \pi_3(d)))\}$
- C.** $inserirTarefa \in DEPARTAMENTO \times TAREFA \times ID \rightarrow DEPARTAMENTO$
 $inserirTarefa = \{(d, t, id) \mapsto d' \mid \exists ap (ap \in \pi_1(d) \wedge \pi_2(ap) = id \wedge \pi_4(t) \leq \pi_1(ap) \wedge d' = (\pi_1(d) \setminus \{ap\} \cup \{(\pi_1(ap), \pi_2(ap), \pi_3(ap), \pi_4(ap) \cup \{t\})\}, \pi_2(d), \pi_3(d)))\}$
- D. $inserirTarefa \in DEPARTAMENTO \times TAREFA \times ID \rightarrow DEPARTAMENTO$
 $inserirTarefa = \{(d, t, id) \mapsto d' \mid \exists ap (ap \in \pi_1(d) \wedge \pi_2(ap) = id \wedge \pi_1(ap) \leq \pi_4(t) \wedge d' = (\pi_1(d) \setminus ap \cup (\pi_1(ap), \pi_2(ap), \pi_3(ap), \pi_4(ap) \cup \{t\}), \pi_2(d), \pi_3(d)))\}$
- E. Nenhuma das anteriores.

6. (20 points) Defina o predicado *subaproveitado* que verifica se um programador está a realizar mais do que três tarefas de níveis de senioridade abaixo do seu.

- A. $subaproveitado(ap) \stackrel{\text{def}}{=} \{t \in TAREFA \mid t \in \pi_4(ap) \wedge \pi_4(t) < \pi_1(ap)\} > 3$
- B.** $subaproveitado(ap) \stackrel{\text{def}}{=} \#\{t \in TAREFA \mid t \in \pi_4(ap) \wedge \pi_4(t) < \pi_1(ap)\} > 3$
- C. $subaproveitado \in PROGRAMADOR \rightarrow BOOL$
 $subaproveitado(ap) \stackrel{\text{def}}{=} \{(ap) \mapsto b \mid b = \#\{t \in TAREFA \mid t \in \pi_4(ap) \wedge \pi_4(t) < \pi_1(ap)\} > 3\}$
- D. $subaproveitado \in PROGRAMADOR \rightarrow BOOL$
 $subaproveitado(ap) \stackrel{\text{def}}{=} \{(ap) \mapsto b \mid b = \{t \in TAREFA \mid t \in \pi_4(ap) \wedge \pi_4(t) < \pi_1(ap)\} > 3\}$
- E. Nenhuma das anteriores.