A.

6

1. Person × Long → Prop ✓
2. Var × Prop → Cmd ✓
3. Prop → Cmd ✓
4. Person ✓
5. Prop × Int → Prop ✓
6. Prop → ~~Prop~~ Cmd ✓
7. Type × Prop → ~~Set~~ Cmd ✓
8. City × ~~Int~~ Nat → Prop ✓

B.

Sejam P, Q : Prop → já feito pelo enunciado

Escolho L. ✓

ExtraioR. ✗ ...de qual conjunção?

Escolho L. ✗

C.

1 - Person x Long -> Prop ✓

2 - Var x Prop -> ~~Prop~~ Cmd ✓

3 - Prop -> CMD ✓

4 - Person ✓

5 - Prop x ~~Var~~ Int -> Prop ✓

6 - Prop -> ~~Prop~~ Cmd ✓

7 - ~~Var~~ Type x Prop -> ~~Prop~~ Cmd ✗

8 - City x nat -> Prop ✓

RESPOSTAS:

7 A. ① (Person × Lang) → Prop ② (Var × Prop) → Cmd

③ Prop → Cmd

4. Person

5. (Prop × Int) → Prop

6. [Prop → Cmd

7. (Type × Prop) → ~~Prop~~ Cmd

8. (City × Nat) → Prop

8 B. Escolha-R.
Suponha ¬P ou ¬Q.
Separo em casos.
Caso ¬P:
Suponha P & Q. (1)
Extraio-L de (1).
Contradição.

Caso ¬Q:
Suponha P & Q. (2)
Extraio-R de (2).
Contradição. ∎

C.
8 Sejam u, v : ints.
Suponha a | u ou a | v.
Seja x : int
Separo em casos.
Caso a | u.
Seja j : int tal que aj = u.
Calculamos:
(uv)x = (aj)v)x [Pela escolha do j]
= a(j(vx)) [Pela assoc. de (x)] (x2)
(x2)
Uso como testemunha j(vx).

CASO a | v.
Seja k : int tal que ak = v.
Uso como testemunha k(ux)

→ Calculamos:
(uv)x = (u(ak))x [Pela escolha do k]
~~= (u(ka))x [Comut. de (x)]~~
= ((ak)u)x [Comutatividade de (x)]
= a(k(ux)) [Assoc. de (x)].

Respostas

A)

1. (Person × Lang) → prop ✓
2. (Var × Prop) → Cmd ✓
3. ~~Prop~~ → Cmd ✓
4. Person ✓
5. (prop × int) → Prop ✓
6. ~~Prop~~ → ~~Prop~~ Cmd ✓
7. (~~Int~~ Type × Prop) → Cmd ✓
8. (Lit × Nat) → prop ✓

B)

Escolha - R. ✓

~~Supor~~ Suponha (¬P ou ¬Q). ✓

Suponha (P & Q). ✓

não use aspas dessa!

Supor em casos "¬P" e "¬Q". ✓

Caso "¬P": ✓

Extraia - L de (P & Q). ✓

Aplique ¬P em P. ✓

Contradiction. ✓

Encerro o caso "¬P".

Caso "¬Q":

Extraia - R de (P & Q).

Aplique ¬Q em Q.

Contradiction.

Encerro o caso "¬Q".

Encerro a ~~demonstração~~.

C)

Seja v: int. ✓

Seja v: int. ✓

Suponha (a|v) ou (a|v). ✓

Seja x: int. ✓

Supor em casos "a|v" e "a|v". ✓

Caso "a|v": ✓

Seja i: int tal que a.i = v. ✓

Demonstrarei a (v.v).x, escolho i.v.x ✓

como testemunha. ✓

Calculamos:

(v.v).x = (i.v.x).a

= (a.i).v.x [?]

= v.v.x [?]

Encerro o caso "a|v".

Caso "a|v":

Similar ao caso "a|v".

Encerro a demonstração.

a.

(1) Person X Lang ⇒ Prop ✓ NÃO confunda os símbolos → e ⇒!
(2) Var X Prop ⇒ Cmd ✓
(3) Prop ⇒ Cmd ✓
(4) Person ✓
(5) Prop X ~~Var~~ Int ⇒ Prop ✓
(6) Prop ⇒ ~~Prop~~ Cmd ✓
(7) ~~Int~~ Type X Prop ⇒ ~~Prop~~ Cmd ✓
(8) City X ~~Real~~ Net ⇒ Prop ✓

b.

8 Escolho - R. ✓
~~Seja P Q : Prop.~~ !! Enunciado!
Suponha ¬P ou ¬Q
Separo em casos. [usando qual disjunção?]
Caso ¬P:
~~Suponha~~ Suponha Q&P. (1)
~~Contradição [Aplicando P⇒⊥]~~
~~Caso ¬Q:~~
~~Suponha P&Q.~~
Extraio - R de (1) ✓
Contradição [Aplicando P⇒⊥ em P] ✓
Caso ¬Q:
Suponha P&Q. (1)
Extraio - L de (1).
Contradição [Aplicando Q⇒⊥ em Q].

c.

~~Seja a:int.~~ Enunciado!!
8 Seja u,v:int.
Suponha a|u ou a|v.
Separo em casos.
Caso a|u:
Seja m:int tal que a·m = u.
Seja x:int.
Calculemos:
uvx = (am)vx [Escolha m]
= a(mvx) [Associatividade]
Uso mvx como testemunha ✓
Caso a|v:
Seja n:int tal que a·n = v
Seja x: int
Calculemos
uvx = u(an)x [Escolha n]
= (an)xu
= a(nxu) [Associatividade]
Uso nxu como testemunha

por que escolheu nomes diferentes?

Respostas:

(A) 1 - Person → Prop (x string) ~

2 - int x Prop → Prop

3 - string → string ?!

4 - Prop ✗

5 - ~~string~~ x int → Prop

6 - Prop → Prop

7 - int x Prop → Prop

8 - City x int → Prop

3

(B) $((\neg P \,\&\, \neg Q) \Rightarrow \neg P \text{ ou } \neg Q) \text{ ou } (\neg P \,\&\, \neg Q <= \neg P \text{ ou } \neg Q)$

$(\neg P \,\&\, \neg Q <=> \neg P \text{ ou } \neg Q)$ //

0

?

(C) a/ $<\overset{def}{=}> (\exists K : int)(PK =)$

?? 0

3 Respostas: VEJA O GABARITO!

a) ??

1) Person, lang
2) Var, prop
3) Prop

4) Prop, Var ?!
5) Prop, Var
6) Prop

7) int, Prop
8) City, int

B)

1 Demonstração:

Suponha P. ✗
Suponha Q ✗
Escolha - b
Suponha ¬(P&Q)
Escolha - b
Suponha ¬P ✗

⊥ ■

C)

0 Demonstração:

Suponha a\/b ✗
Suponha a\/b
Suponha x

Calculamos:

A.

1) (Pessoa × Strings) → Prop ✓
2) ~~(Var × Prop(Int)) → Cmd~~ (Var × Prop) → Cmd ✓
3) Prop → Cmd ✓
4) Person ✓
5) (Prop × Int) → Prop ✓
6) Prop → Cmd ✓
7) ~~Type →~~ (Type × Prop) → Prop ✓
8) (City × Nat) → Prop ✓

B. Demonstração:
Do alto, escolha R ✓
Suponha (¬P ou ¬Q) (1) ✓
~~Suponha (P & Q) (1)~~
Suponha (P & Q) (2) ✓
Separe (1) em casos: ✓
Caso L (¬P) ✓
De (2), extraia 2. (P).
Aplicamos ¬P em P, e obtemos ⊥.
Contradição.
~~Fim do caso L.~~
Caso R (¬Q) ✓
De (2), extraia R (Q)
Aplicamos ¬Q em Q e obtemos ⊥.
~~Fim do caso R~~
Fim da demonstração

C. Demonstração.
Seja u: int ✓
Seja v: int ✓
Suponha que ((a|u) ou (a|v)) (1) ✓
Seja x: int ✓
Separe (1) em casos: ✓
- Caso L (a|u) ✓
Seja k: int tal que a·k = u ✓
~~Seja k: int~~
Calculamos:
(u·v)·x = ((a·k)·v)·x [Trans./=]
= a·(k·v)·x [Ass.(·)]
Uso k·v·x como testemunha. ✓
Caso R (a|v) ✓
Seja k: int tal que a·k = v. ✓
Calculamos:
(u·v)·x = (u·(a·k))·x [Trans./=] (& com.)
= a·(u·k)·x [Ass.(·)]
Uso (u·k)·x como testemunha.
Fim da demonstração.

Respostas

A.
(1) Person × long → Prop
(2) Var x (Var → Prop) → Prop
(3) Prop → Prop
(4) Person
(5) Prop × Int → Prop
(6) Prop → Prop
(7) Type x (Type → Prop) → Prop
(8) City × Nat → Prop

B. Demonstração:
Escolho ¬P.
Suponha ¬P ou ¬Q. (2)
Suponha P & Q. (1)
Extraio ¬L do (1)
Extraio P do (1)
Supono (2) em casos.
Caso ¬P:
Contradição. [¬P, P]
Caso ¬Q:
Contradição. [¬Q, Q]

C. Demonstração:
Sejam u, v : Int.
Suponha a/u ou a/v. (1)
Seja x : Int.
Supono (1) em casos.
~~Caso a/u:~~ Caso a/u:
seja i : Int tal
que ai = u.
Calculamos:
(uv)x = ((ai)v)x [pela escolha de i]
= (a(iv))x [(·)-Asso]
= a((iv)x) [(·)-Asso]
= a(ivx) [(·)-Asso]
Testemunho (ivx).
Como (uv)x = a(ivx) então a(ivx) = (uv)x [Simetria]

Caso a/v:
Seja i : Int tal que ai = v.
Calculamos:
(uv)x = ((ai)v)x [escolha i]
= (a(iv))x [(·)-Asso]
= a(ivx) [(·)-Asso]
Testemunho (ivx).
Como (uv)x = a(ivx)
então a(ivx) = (uv)x [Simetria]

A)

(1) Person × Lang → Prop ✓
(2) Var × Prop → Cmd ✓
(3) Prop → Cmd ✓
(4) Person ✓
(5) Prop × Int → Prop ✓
(6) Prop → Cmd ✓
(7) Type × Prop → Cmd ✓
(8) City × ~~Int~~ Nat → Prop ✓

B)

DEMONSTRAÇÃO:
Escolho - R. ✓
Suponha ¬P ou ¬Q. ✓
Separo em casos.
Caso ¬P: ✓
Suponha P ∧ Q (1). ✓
Extraio L da (1) ✓
Aplicando ¬P na P temos ⊥. ✓
Contradiction. ✓
Caso ¬Q:
Suponha P ∧ Q (1).
Extraio R da (1).
Aplicando ¬Q na Q temos ⊥.
Contradiction.
Fim.
Similar

C) DEMONSTRAÇÃO:
Seja u, v : Int. ✓
Suponha a | u ou a | v (1). ✓
Seja x : Int. ✓
Separo (1) em casos: ✓
Caso a | u: ✓
Seja k : Int t.q ak = u (2). ✓
Uso k.x como testemunha. ✓
Calculamos:
a(k.x) = (ak)(x) [assoc. (·)]
= (u)(vx) [(2)]
= (uv)x [Assoc. (·)]
Termino o caso.
Caso a | v
Seja k : Int t.q ak = v (3).
Uso k.ux como testemunha.
Calculamos:
a(k.ux) = (ak)(ux) [assoc.(·)]
= v(ux) [(3)]
= v(xu) [comut.]
= (vx)u [Associat.]
= u(vx) [Comutati]
= (uv)x [Associati]
Termino o caso.
Termino a demonstração. ✓

A.
1. Person × Lang → Prop ✓
2. Var × Prop → Prop
3. Prop → Cmd ✓
4. Person ✓
5. Prop × Var → Prop
6. Prop → Prop
7. Type × Prop → Prop
8. City × Int → Prop ✓

essa frase não faz sentido. «ser elemento» não significa nada.

C. Seja w: int, tal que w é um elemento genérico.

Se $a|u$ ou $a|v$, temos que existe um inteiro tal que $a \cdot w = u$ ou $a \cdot w = v$.

Podemos então afirmar que a se $a|u$ ou $a|v$, $a|u.v$. COMO?

Por w ser um elemento genérico, fica provado que $\forall x$ inteiro $a|(u \cdot v)x$

→ Nunca escreva esse tipo de frase
se tu acha que pode afirmar algo, então afirme!

Pedi para usar a linguagem low-level exatamente para evitar esses abusos e handwavings...

RESP.

A.

2

1- .: (PERSON × LANG) → ?

2- .: (VAR → REAL) ?!

3- .: (PROP) ✗

4- .: (PERSON) ✓

5- .: (TYPE × VAR)

6- .: (VAR → TYPE)

7- .: (VAR × PROP) ✗

8- .: (CITY × NAT) ~

C.

1

undeclared!

SUPONHA $a|U$ & $a|V$

$a|U \overset{DEF}{\Leftrightarrow} (\exists J:NT)(a \cdot J = U)$ ✗ não é permitido definir isso!!

$a|V \overset{DEF}{\Leftrightarrow} (\exists T:INT)(a \cdot T = V)$ ✗

SEJA X := INT ← nunca usamos essa notação para tipagem

CALCULAMOS:

$\frac{(UV)X}{a} = \frac{(a \cdot J \cdot a \cdot T)X}{a} = (aJT)X$?!?

POR $a:=INT, J:=INT, T:=INT, X:=INT$, ?!

TEMOS $(aJT)X := INT$

~~SLOGO, EXISTE INTEIRO TAL QUE~~
~~PARA TODO X, O PRODUTO DESSE INTEIRO~~
~~POR a RESULTA EM (UV)X~~

LOGO PARA TODO (X), EXISTE ~~INT~~ INTEIRO
TAL QUE ~~DESSE INTEIRO VEZES~~ O
~~INTEIRO~~ O PRODUTO DESSE INTEIRO
POR (a) RESULTE EM ((UV)X).

esse texto não faz sentido! Low-level!

usamos variáveis para referir...

RESP.

A.

7 ① Person × Lang → Prop ✓
② Var × Prop → Cmd ✓
③ Prop → Cmd ✓
④ Person ✓
⑤ Prop → Cmd ✗
⑥ Prop → Cmd ✓
⑦ ~~Type → Prop~~ Type × Prop → Cmd ✓
⑧ City × ~~Int~~ Nat → Prop

B.

1 ① Demanz.

Escolho - L

Suponha ¬(P & Q)

... Já que não conseguiu continuar, por que não voltou para escolher-R?

C. Posso ter confundido os sinais (∨/∧) (&/ou)

1 Demanz.

Suponha alu Valv (1)

Escolho - da 1

Se alu, então para todo inteiro, ak = u

Prop?

(∃k. int) [ak = u]

5

Prova - Respostas

A

1. Person x Lang → Prop ✓
2. VAR x PROP → CMD ✓
3. PROP → CMD ✓
4. STRING Minha irmã não é um string! ⁘
5. STRING x INT → PROP ✗
6. PROP → ~~PROP~~ cmd
7. TYPE ~~x~~ PROP → CMD ✓
8. CITY x ~~INT~~ MAT → PROP ✓

B

ESCOLHO O LADO ESQUERDO; ✓
SUPONHO ~~(FALSO)~~ ¬(P & Q); ✓
EM MEU ALVO, ESCOLHO L E SUPONHO P; (1) ✓
EM MEUS DADOS SPLITO P&Q EM LADOS P & Q ✗ não tens esse dado!
LADO P:
CONFORME (1), AO SUPOR P, INFERE-SE UMA ABSURDO;
LADO Q:

C

$\forall (U, V : int)\,[a|u \text{ ou } a|v] \Rightarrow \forall (x : int)\,[a|(ux)x]$

FMC I

[redacted]

Respostas avaliação

A.

6

1. : Person × Lang → Prop ✓
2. : Var × Prop → Cmd ✓
3. : Prop → Cmd ✓
4. : Person → Prop ✓
5. : Prop × ~~Var~~ Int → Prop
6. : Prop → ~~Prop~~ Cmd
7. : ~~Int~~ Type × Prop → Prop
8. : City × Nat → Prop ✓

B.

7

~~Sejam P, Q : Prop~~ enunciado
Escolho_L ✓
~~Suponha P~~
Suponha ¬(P & Q) ✓
~~Escolho_L~~
Separo ¬ P em casos ✗ separar em casos é o uso de disjunções, mas aqui tu tá (tentando) usar uma negação.
Caso P:
Assumption

C.

2

Sejam u, v : Int ✓
Suponha a|u ou a|v ✓
Escolho_L ✗ teu alvo não é uma disjunção.
Seja y : Int t.q. ay = u
Escolho_R
Seja z : Int t.q. az = v
~~Calculamos~~ Seja x : Int
Seja k : Int t.q. ak = (uv)x ← aqui tá usando um existencial? (:)
Calculamos:

RESPOSTAS

A.1 – Pessoa → Long X

1 2 – Var → Card X

3 – Card X

4 – Pessoa

5 – Var → Var X

6 – Card X

7 – Int → Int X

8 – City → Set X

B – P e Q proposições

0 $(\neg(P \& Q) \Rightarrow \neg P \vee \neg Q) \vee (\neg(P \& Q) \Leftarrow \neg P \vee \neg Q)$

DEMONSTRAÇÃO

Seja P e Q proposições

Escolha Id ~~Direito~~ Esquerdo

Considere $\neg(P \& Q) \Leftrightarrow \neg P \vee \neg Q$ (D'MORGAN)

Portanto $\neg(P \& Q) \Rightarrow \neg P \vee \neg Q$

C – Seja a : Int

0 Demonstre para quaisquer u, v inteiros

Se a|u ou a|v, então para todo inteiro, a|(u v) x

DEMONSTRAÇÃO

Seja a : Int

Considere a|u ou a|v $\forall u, v$: Int ??

Calculo u.v : Int

Calculo (u.v).x : Int

⋮

Respostas:

A)

1): (Person x Lang) → Prop ✓
2): (Var x Prop) → Prop
3): (Prop) → Cmd ✓
4): Person ←
5): (Prop x Int) → Prop
6): Prop → Prop
7): (Type x Prop) → Cmd ✓
8): (City x Nat) → Prop ✓

B) Demonstrações

0 (-1)

Considere a natureza da conjunção "&", que recebe 2 "inputs", neste caso as Props P e Q. 1 "output".

Dados	Alvo(s)
P: Prop	
Q: Prop	

Nenhuma "natureza" para "considerar".

Considere escrever linhas de demonstração! :P

C)

0

Dados	Alvo
a: Int	(∃x: Int)[a | (u . v) x]
u: Int	
v: Int	

RESPOSTAS

(A)
1: Person × Lang → Prop ✓
2: Var × Prop → Cmd ✓
3: Prop → Cmd ✓
4: → String Minha irmã não é um string! :-(
5: Prop × ~~Var~~ → Prop
6: Prop → ~~Prop~~
7: Type × Prop → ~~Prop~~
8: City × ~~int~~ → Prop

4

RESPOSTAS

(A)

① Person x Long → Prop ✓
② Var x Prop → Cmd ✓
③ Prop → Cmd ✓
④ String Grrr...
⑤ Prop x Int → ~~Cmd~~ Prop
⑥ Prop → Cmd ✓
⑦ ~~Int~~ Type x Prop → ~~Prop~~ Cmd
⑧ City x ~~Int~~ Nat → ~~Prop~~ Cmd

(B)

DEMONS

Escolho - R ✓
Suponho (¬P ou ¬Q) (1) ✓
Suponho P & Q (2) ✓
Extraio - L de (2), logo P (3) ✓
Extraio - R de (2), logo Q (4) ✓
Separando (1) em casos: ✓

Caso ¬P:

Como ¬P ⇔ P ⇒ ⊥
Contradição.

Caso ¬Q:

Como ¬Q ⇔ Q ⇒ ⊥
Contradição

~~Concluo.~~ ✓

(C) DEMONS

Seja u, v: Int
Suponha a|u ou a|v (1)
Seja f: Int t.q. af = u (2) ✗ não tens a|u!
Seja g: Int t.q. ag = v (3) ✗ não tens a|v!

Calculamos:

PMova

A

1) : [illegible] (Person x Long) → PMOP ✓

2) : [illegible] (VeM x PMOP) → Cmd ✓

3) : [illegible] PMOP → Cmd ✓

4) : Person ✓

5) : [illegible] (PMOP x Int) → PMOP ✓

6) : [illegible] PMOP → ~~PMOP~~ Cmd ~

7) : [illegible] (Type x PMOP) → ~~PMOP~~ Cmd ~

8) : [illegible] (City x Nat) → PMOP ✓

B

Demons

Escolha R. ✓

Suponha ¬P ∨ ¬Q. ✓ → Separo em casos.

~~Parte~~ CASO ¬P // ¬P <=> P => ⊥

Suponha P ∧ Q.

Extraia L de P ∧ Q. ✓

// Pois P => ⊥

~~Demonstração parte ¬P concluída.~~

~~Parte~~ CASO ¬Q // ¬Q <=> Q => ⊥

Suponha P ∧ Q.

Extraia R de P ∧ Q

// Pois Q => ⊥

~~Demonstração Parte ¬Q concluída.~~

□

C

Demons

Undeclared.

? Suponha a|u ∨ a|v. ??

~~Parte~~ a|u // a|u <=> (∃K: Int)[a.K=u]

Seja x: Int.

Seja K: Int t.q K =

. .

~~Parte~~ a|v

Sorry :(

RESPOSTAS

(A)

7

① Person × Long → Prop ✓
② Var × Prop → Cmd ✓
③ Prop → Cmd ✓
④ Person ✓
⑤ Prop × Int → Prop ✓
⑥ Prop → ~~Prop~~ ~
⑦ Type × Prop → Cmd ✓
⑧ City × Nat → Prop ✓

(B) DEMONS:

Escolho - R ✓
Suponha (¬P ou ¬Q) (1) ✓
Suponha (P & Q) (2) ✓
~~Split~~ Separo em casos [a partir de (1)]
~~Pelo~~ ¬P: (3)
CASO Extraio - L [a partir de (2)] (4)
Aplico (3) em (4) [Aplicar ¬P em P]
Contradiction
~~Pelo~~ ¬Q: (5)
CASO Extraio - R [a partir de (2)] (6)
Aplico (5) em (6) [Aplicar ¬Q em Q]
Contradiction

(C) DEMONS:

7 Sejam u, v: Int ✓
Suponha (a|u ou a|v) (1) ✓
Seja x: Int ✓
~~Split~~ [a partir de (1)]
~~Pelo~~ a|u:
Seja j: Int t. ? a.j = u
Seja k: Int t. ? k = v.x ?! isso nunca faz sentido.
~~Uso~~ Uso j.k como testemunha

Pelo a|v:
Seja l: Int t. ? a.l = v
Seja m: Int t. ? m = ux
Uso l.m como testemunha

TÁ NOS DADOS.

Ⓐ 1. Person × Lang → Prop. ✓
2. Var × Prop → Cmd. ✓
3. Prop → Cmd. ✓
4. Person. ✓
5. Prop × Int → Prop. ✓
6. Prop → Prop.
7. ~~Int~~ × Prop → ~~Prop~~.
8. Cit × Var → Prop. ✓

Ⓑ Escolha R. ✓

3 Suponha ¬P em ¬Q. ✓
Separar em casos. ✓
Caso ¬P: ✓
Suponha P. ✗ nope.
Aplicando ¬P em P, temos ⊥.
Caso ¬Q:
Suponha Q. ✗ de onde?
Aplicando ¬Q em Q, temos ⊥.
Junte os casos. o que é isso?!?!
Tá nos dados.

Ⓒ Seja u: int. ✓
5 Seja v: int. ✓
Seja x: int. ✓
Suponha A/u em A/v. ✓
Dividir em casos. ✓
Caso A/u: ✓
Seja r: int, tal que ar = u [A/u]. ✓
Caso A/v:
Seja s: int, tal que as = v [A/v].
Junte os casos. ✗
Calculamos
(u·v)x = ar · as · x (multiplicação dos Int)
= a(r·a·s)x. (comutatividade).
~~Transitividade a(r·a·s)x = (u·v)x.~~
Uso r·a·s como testemunha.
Tá nos dados.

Respostas:

A) 4

1) person × lang → prop. ✓
2) var × prop → cmd ✓
3) prop → cmd. ✓
4) prop × int → prop. ?!?
5) prop × int → prop. ✓
6) type × prop → prop
7) ~~String~~ × ~~int~~ → prop.

B) 2

Demonstração:

Suponha ¬(P & Q); ✗
Escolho - L;
Suponha P;
Escolho - R; ✗
Suponha Q;
Como (P & Q) => ⊥, P, Q, logo ⊥. ???

?!

Demonstração:

Suponha (¬P ou ¬Q); ✓
Suponha (P & Q); ✓

Como (P => ⊥ ou Q => ⊥), separo em dois casos. ✓

Caso P => ⊥. ✓

Como P => ⊥, (P & Q), logo ⊥. por quê?

Caso Q => ⊥

Como Q => ⊥, (P & Q), logo ⊥.

C) 2

Demonstração:

Sejam u, v : int
Suponha a|u ou a|v
Seja x : int

C.

6- Seja U : int ✓
Seja V : int ✓
Suponha A|U ou A|V ✓
Seja x : int ✓
~~Split~~ : Separar em casos [pelo...]
Caso A|U : ✓
seja i : int tal que A · i = U ✓

Calculemos:

$(A \cdot i) \cdot (V \cdot x) = U \cdot (V \cdot x)$ [pela ideia de i]
$= (U \cdot V) \cdot x$ [por associatividade]

sem gerúndio — usando $i(V \cdot x)$ como argumento — ?! testemunha?

~~Logo como $A \cdot i \cdot (V \cdot x) = (U \cdot V) \cdot x$, então $(U \cdot V) \cdot x = A \cdot i \cdot (V \cdot x)$~~

Caso A|V :
seja k : int tal que $A \cdot k = V$

Calculemos:
$A \cdot k \cdot (U \cdot x) = V \cdot (U \cdot x)$ [pela ideia de k]
$= (U \cdot V) \cdot x$ [por associatividade]

usando $k \cdot (U \cdot x)$ como argumento

Logo como $A \cdot k \cdot (U \cdot x) = (U \cdot V) \cdot x$, então $(U \cdot V) \cdot x = A \cdot k \cdot (U \cdot x)$

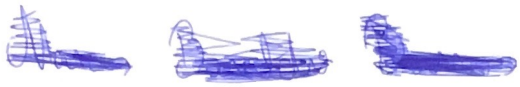
Similar

B-)

2 Escolha -L ✓
suponha $\neg(P \wedge Q)$
Escolha -L
Supondo P — Gerúndio?!
Extrair -L de $\neg(P \wedge Q)$ — extrair de uma implicação?
~~Aplicar~~ Contradiction [aplicando $P \Rightarrow \bot$ em P].

A.

I-) PROP
II-) CMD
III-) CMD
IV-) STRING
V-) PROP
VI-) PROP
V II-) PROP
V III-) PROP



(B)

Comandos

Escolho L.

Escolho L X

minha resposta está nos meus dados ?!

Dados

P

Q

$a = P \,\&\, a$

$b = \neg(a)$

$\neg(a) = \bot$

alvo

~~$\neg(P \,\&\, Q) \Rightarrow \neg P$ ou $\neg Q$~~

$\bot$ ou $\bot$

$\bot$

(A)

2

① person → lang

② int × prop

③ prop

④ person ✓

⑤ prop → Prop

⑥ prop → ~~prop~~

⑦ (prop → Prop) × (prop → prop)

⑧ city → prop

(C) Comandos

Dados

a : int

u : int

v : int

x : int

alvo

~~$(\forall u, v : int)[a/u$ ou $a/v \Rightarrow$~~

~~$(\forall x : int)[a/(u \vee v)_x]$~~

a/u ou $a/v \Rightarrow a/(u \vee v)_x$

B) ~~Sejam P, Q : Prop.~~

Escolha-R. ✓

Suponha (¬P ou ¬Q) (1). ✓

Suponha (P & Q). ✓

Separe (1) em caso ¬P e caso ¬Q. ✓

Caso ¬P: ✓

Extraia-L de (P & Q). ✓

Aplique ¬P em (P. ✓

Contradiction. ✓

Caso ¬Q: ✓

Extraia-R de (P & Q). ✓

Aplique ¬Q em (Q. ✓

Contradiction ✓

Demonstrado

A) 1 PERSON x LANG → PROP ✓

2 VAR x PROP → CMD ✓

3 PROP → CMD ✓

4 PERSON ✓

5 PROP x INT → PROP ✓

6 PROP → ~~PROP~~ CMD

7 TYPE x PROP → ~~PROP~~ cmd

8 CITY x NAT → PROP ✓

C)

Demonstração

Seja u:int. ✓

Seja v:int. ✓

Suponha (alu) ou alv. ✓

Seja y:int tq. ay = u. (1) não tem

Seja z:int tq. az = v. (2) não tem

Seja x:int ✓

Calculemos:

uvx = (ay)(az)x [pon (1) e pon (2)]

= a(ayzx) [pon (*) associatividade]

Uso (ayzx) como testemunha.

Demonstrado.

Dado

a:int.

u:int.

v:int

alu ou alv

ay = u

az = v.

Alvo

~~alv ou alv → (∀x:int)~~

[a(uv)x]

∃.

(∃n:int)[an = (uv)x

RESPOSTAS

A)

1) Person × Lang → Prop ✓
2) Var × Prop → Cmd ✓
3) Prop → Cmd ✓
4) Person ~~→ String~~
5) Prop × ~~Var~~ → Prop
6) Prop → ~~Prop~~
7) type × prop → ~~Set~~
8) City × ~~Int~~ → Prop ✓
   Nat

B) Demonstração:

Escolho $(\neg(P \& Q) \Rightarrow (\neg P \text{ ou } \neg Q)$ X

Suponho $\neg(P \& Q)$ ✓

Extraio $\neg P$ de $\neg(P \& Q)$ Não tem como extrair algo duma implicação!

Aplico $P \Rightarrow \bot$ em P

boom!!!

C) Demonstração:

Seja u, v: Int. ✓

Suponha $a \mid u$ ou $a \mid v$. ✓

Separe (1) em casos. ✓

caso $a \mid u$ ✓

Seja x: Int ✓

Seja m Int tq $a \cdot m = u$ ✓

Escolho $(m \cdot v \cdot x)$ como testemunha ✓

Calculamos:

$(u \cdot v)x = (am \cdot v)x$ [pela escolha de m] ✓

$= a \cdot (mvx)$ ✓

Como $(uv)x = a \cdot (mvx)$, logo $a \mid (u \cdot v)x$ ✓

Demonstração concluída!

caso $a \mid v$

Seja x: Int ✓

Seja n Int tq $a \cdot n = v$

Escolho $(u \cdot n \cdot x)$ como testemunha

Calculamos:

$(u \cdot v)x = (u \cdot an)x$ [pela escolha de n]

$= a(unx)$

Como $(u \cdot v)x = a \cdot (unx)$, logo $a \mid (u \cdot v)x$

Demonstração concluída!

DADOS

a: Int

u: Int

v: Int

(1) $a \mid u$ ou $a \mid v$

$(\exists m$: Int$)[a \cdot m = u]$ $(\exists n$: Int$)[a \cdot n = v]$

m: Int

ALVO

~~$(\forall u, v$: Int$)$~~ $[a \mid u$ ou $a \mid v$

$\Rightarrow$

$(\forall x$: ~~Int~~$)$

$[a \mid (u \cdot v)x]$

$(\exists b$: Int$)[a \cdot b = (u \cdot v)x]$

✓

A -

1 - (Person × Lang) → Prop ✓

2 - (Var × Prop) → Cmd ✓

3 - Prop → Cmd ✓

4 - Person ✓

5 - (Prop × Int) → Prop ✓

6 - Prop → Prop

7 - (Type × Prop) → Prop

8 - (country × Nat) → Prop ✓

B -

Escolho R ✓

Suponho ¬P ou ¬Q ✓

Suponho P & Q ✓

Separo em casos, pela ...

Caso (P ⇒ ⊥) ✓

Extraio-o dz

Contradição (Aplicando P a P ⇒ ⊥)

Caso (Q ⇒ ⊥)

Extraio-o ...

Contradição (Aplicando Q a Q ⇒ ⊥)

C -

Seja u, v : int ✓

Suponho a|u ou a|v ✓

Seja x : int ✓

Seja k : int t.q. ak = (uv)x ← como??

Calculemos

a 1- Person × lang → Prop. ✓

2- Var → Prop.

3 3- comando.

4- Person. ✓

5- Prop → Prop.

6- Prop → Prop.

7- Prop → type : Prop.

8- city × num → Prop.

2

3. Sejam u, v int ✓

escolha left. ✗ teu alvo não é uma disjunção.

1 seja K: int tal que aK=u

seja q: int tal que aq=(u.v)x ??

cancelamos.

aK = u

aq = (u.v)x

aq = aK(v.x)

a(q) = a(K.v.x)

→ isso não é um cálculo

Assim, tendo o q e o K como testemunhos, demonstramos que se a/u → a/(uv)x.

A. 9

1: person × lang → prop ✓
2: var × prop → cmd ✓
3: prop → cmd ✓
4: person ✓
5: prop × int → prop ✓
6: prop → cmd ✓
7: type × prop → cmd ✓
8: city × nat → prop ✓

C.

7+ Escolho-R ✓

Suponho ¬P ou ¬Q (1) ✓
Suponha P & Q ✓
Separamos ~~em~~ (1) em casos ✓
Caso ¬P: ✓
Extrais-L de
Aplico ¬P no P ✓
Contradição
Caso ¬Q:
Extrais-R...
Aplico ¬Q ~~no~~ Q
Contradição

B

2

Seja u, v: int ✓
Suponha u|v ou u|v (1)
Extraia-L do (1)
Seja m: int tq am = v
Extraia-R do (1)
Seja n: int tq an = v
Calculamos:
(vv)

Respostas

(A) ① (Person × Lang) → Prop ✓
② (Var × Prop) → Cmd ✓
③ Prop → Cmd ✓
④ Person ✓
⑤ (Prop × Int) → Prop ✓
⑥ Prop → Cmd ✓
⑦ (Type × Prop) → ~~Prop~~
⑧ (City × Nat) → Prop ✓

8-

(B) DEMONSTRAÇÃO:

Suponha ~~Seja~~ ¬P ou ¬Q. ✓
Suponha ~~Seja~~ P & Q. ✓
Separo em casos (~~P ou Q~~) ¬P ou ¬Q. ✓
Caso ¬P: ✓
~~Uso~~ extraio o P de P&Q. ✓
Contradição. ✓
Caso ¬Q:
~~Uso~~ extraio o Q de P&Q. ✓
Contradição. ✓

8

(C) DEMONSTRAÇÃO:

Sejam u, v inteiros. ✓
Suponha a | u ou a | v. ✓
Seja x inteiro. ✓
Separo em casos a | u ou a | v. ✓

Caso a | u: ✓
Seja i inteiro t.q. a.i = u. ✓
Calculamos:
a(ivx) = u.v.x (Pela ... msgj ... declaração de i). (escolha)

Uso ivx como testemunha para ~~demonstrar a(ivx) = (uv)x~~.

Caso a | v:
Seja j inteiro t.q. a.j = v.
Calculamos:
a(jux) = v.u.x (Por def. j)
= (u.v)x (comutatividade)

Uso (jux) ~~para dizer~~ como testemunha para demonstrar a(jux) = (uv)x.

7-

Respostas:

A.

7

① Person × Lang → Prop ✓

② Var × Prop → Cmd ✓

③ Prop → Cmd ✓

④ Person ✓

⑤ Prop × Nat → Prop ✓

⑥ Prop → Prop ✓

⑦ Set × Prop → Prop ✓

⑧ City × Nat → Prop ✓

B.

2ª

Escolho -L.

Suponha ¬(P & Q).

(P & Q) => ⊥

Logo, P => ⊥ & Q => ⊥.

Qual comando seria esse?

Caso P:

Como P => ⊥ temos contradição.

Caso Q:

Como Q => ⊥ temos contradição.

Escolho -R.

Suponha (¬P ou ¬Q).

Caso ¬P:

Temos P => ⊥.

(P & Q) => ⊥ ← afirmação seca

Logo, Q => ⊥ também.

Caso ¬Q:

Temos Q => ⊥ | Logo, P => ⊥ também.

(P & Q) => ⊥

HW

Não é "apenas" que não temos um comando para isso.

Seria errado ter.

Qual seria um exemplo para demonstrar isso?

Teu alvo era uma disjunção e tu já escolheu qual dos dois lados tu vai demonstrar.

C. Sejam u, v inteiros. ✓

I

Suponha que (a|u ou a|v). ✓

~~Para~~ Seja k : int tal que a · k = u ✗ Não tens a|u

~~Para~~ Seja i : int tal que a · i = v ✗ Não tens a|v

Seja x ∈ ℤ. ✓

Calculamos:

(uv) · x = (ak · v) · x (Pela escolha de k) ✓

= (ak · ai) · x (Pela escolha de i) ✗

= (k · i · a) · x (Associatividade & comutatividade)

o que acontece no outro a??

Por simetria temos então que:

(uv) · x <=> (k · i) a · x ✗ type error

A) (1) Person × Lang → Prop ✓

(2) Var × Prop → cmd ✓

(3) Prop → cmd ✓

(4) Person ✓

(5) Prop × Int → Prop ✓

(6) Prop → ~~Prop~~ Cmd

(7) Type × Prop → cmd ✓

(8) City × Nat → Prop ✓

7+

B)

8

Demonstração:

Escolha ¬R. ✓

Suponha ¬P ou ¬Q. ✓

Suponha P & Q (1). ✓

Caso a ¬P: ✓

Extraio -L de (1). ✓

Contradiction. ✓

Caso a ¬Q: ✓

Extraio -R de (1). ✓

Contradiction. ✓

~~Encerro a demonstração.~~

Separo em casos pela ___.

C)

4

Demonstração:

Suponha a/u ou a/v. ✓

~~Seja k: int, tal que, a k = u. (1)~~ X

~~Seja i: int, tal que a i = v. (2)~~ X

Seja x: int. ✓

Caso a/u: (3)

Calculamos

~~(u v) x = (a i . a k) x [pela escolha de (1) e (2)]~~

~~= a (i k) x~~

(u v) x = (a k . v) x [Pela Escolha de (1)]

= a (k . v) x

Encerro,

Caso a/v: (4)

Calculamos:

(u v) x = (u . a i) x [Pela Escolha de (2)]

= a (i . u) x

Encerro.

~~Pelo caso 2~~

Pelo (3) e (4), temos que, ∀ x: int, a/(u v) x.

referindo àlgum DADO rotulado escreva << pelo (1) >>.

O << pela escolha de_ >> deve ser seguido por um nome escolhido Isso não faz sentido por ti; aqui seria << pela escolha de k >>.

X

A) 6

1- Pessoa × Lang → Prop ✓
2- Var × Prop → Cmd
3- Prop → Cmd ✓
4- Pessoa ✓
5- Prop × ~~Var~~ → Prop
6- Prop → ~~Prop~~
7- type × Prop → ~~Prop~~ Cmd
8- City × ~~Int~~ → Prop
Nat

C) 8

~~Seja a : Int~~ ✓
Seja u : Int ✓
Seja v : Int ✓
Suponha a | u ou a | v ✓
Seja x : Int ✓
Separo em casos. ✓ pela _ .
Caso a | u: ✓
Seja k Int tq a.k = u ✓
Escolho (k(vx)) como testemunha
Calculemos: (uv)x = u(vx) ✓
= (ak)(vx)
= a(k(vx))
Como (uv)x = a(k(vx)), logo a|(uv)x ✓
Caso a | v:
Seja l Int tq al = v ✓
Escolho (l(ux)) como testemunho.
Calculemos: ~~(uv)x = (vu)x~~ ✓
(uv)x = (vu)x
= v(ux)
= (al)(ux)
= a(l(ux))
Como (uv)x = a(l(ux)), logo a|(uv)x.

por que escolheu nomes diferentes?

B) 0

~~Sejam P, Q : Prop~~ enunciado!
Suponha ¬(P ∧ Q) ✗ teu alvo não é uma implicação.
aplico ¬ em (P ∧ Q) ✗
Escolho ¬P
~~Suponha~~ Suponha P ⇒ ⊥ (I)
aplico P em (I)

A 6†

① (Person × Long) → Prop ✓
② (Var × Prop) → Cmd ✓
③ Prop → Cmd ✓
④ Person ✓
⑤ (Prop × Int) → Prop ✓
⑥ Prop → ~~Prop~~ –
⑦ (~~Int~~ × Prop) → ~~Prop~~
⑧ (City × Nat) → Prop ✓

B Σ

DADOS:
$\neg(P \wedge Q)$
P

ALVO:
$(\neg(P \wedge Q) \Rightarrow \neg P \vee \neg Q)$ ou $(\neg(P \wedge Q) \Leftarrow \neg P \vee \neg Q)$

$\neg(P \wedge Q) \Rightarrow (\neg P \vee \neg Q)$

$\neg P \vee \neg Q$

$\neg P$
∧
|||
∨
$P \Rightarrow \bot$
$\bot$

Demonstração
»» Escolha L; ✓
Suponha $\neg(P \wedge Q)$; ✓
Escolha L; ✓
Suponha P; ✓
Contradição ✗ Tu não tens um ⊥ nos teus dados!
Fim da demonstração.

C 7

Demonstração
»» Seja u:Int; ✓
Seja v:Int; ✓
Suponha a u; ✗ teu alvo não tem a forma a u ⇒ ..... .
Seja x:Int
Suponha a (v u) = x; ✗ teu alvo não tem a forma a (v u) = x ⇒ ... ...
Logo a 1 (u v) x;

(A)

1 PERSON x LANG → PROP ✓

2+ 2 VAR x ~~TYPE~~ → –

3 PROP → ~~PROP~~

4 ~~STRING~~ ::

5 PROP x ~~VAR~~ → PROP

6 PROP → ~~PROP~~

7 TYPE x PROP →

8 CITY x ~~INT~~ → PROP

(C)

1 SEJAM ~~a~~ U, V inteiros. ✓ → enunciado!

SUPONHA a|U ou a|V ✓

CALCULEMOS:

a|V → $a \cdot k^2 = U$, a|V → $a \cdot 2 = V$

Nenhuma dessas proposições cabe nos teus dados!

Não use setinhas mágicas!

isso não é um cálculo!

$UV = a \cdot 2 \cdot ta$

$UV = a(2 \cdot t)$ } afirmações secas

$X(UV) = (a(2 \cdot t)) \cdot X$

COMO $(2 \cdot t)$ = INTEIRO LOGO $(UV)_X \not\mid a$

$a \mid (\forall x : \mathbb{Z} \Rightarrow (\mu u) x = ax$
$\exists k : \mathbb{Int}$

Respostas

A.

4+ ① : (Person × Lang) → Prop ✓

② : (Var × Prop) → ~~Prop~~

③ : Prop → ~~Prop~~

④ : Person ✓

⑤ : ~~(Prop × Prop)~~ (Prop × ~~Int~~) → Prop

⑥ : Prop → ~~Prop~~

⑦ : (Type × Prop) → ~~Prop~~

⑧ : ((Lity × Nat) → Prop ✓

B)

7+ Escolha-R ✓

Suponha $\neg P$ ou $\neg Q$ ✓

Separe a partir do $\neg P$ ou $\neg Q$ em ~~2 jogos~~ casos ✓

Caso $\neg P$: ✓

* Suponha $P \wedge Q$ ✓

Extraia P de $P \wedge Q$ ✓

Absurdo (P e $\neg P$). ✓

~~Logo $\neg (P \wedge Q)$~~ ×

Caso $\neg Q$: ✓

* Suponha $P \wedge Q$ ✓

Extraia Q de $P \wedge Q$ ✓

Absurdo (Q e $\neg Q$). ✓

~~Logo $\neg (P \wedge Q)$~~ ×

Logo $\neg (P \wedge Q)$, demonstrado ×

C)

5+ Sejam $a, \mu, \nu : \mathbb{Int}$ ✓ (enunciado)

Suponha $a \mid \mu$ ou $a \mid \nu$ ✓

Separe $a \mid \mu$ ou $a \mid \nu$ em casos ✓

Caso $a \mid \mu$: ✓

- Seja $k : \mathbb{Int}$ tal que $\mu = a \cdot k$ (1) ✓
- [illegible]
- Multiplique (1) por $(\forall x)$ em ambos os lados ✓ (como? comando inventado?)
- Efetue propriedade comutativa (onde)
- Tome $(k \cdot \nu \cdot x)$ como testemunho que $\exists x : \mathbb{Z}. x = \mu$

↳ NUNCA use a "," num existencial como abreviação inventada de "tal que"!

Caso $a \mid \nu$

- Seja $k : \mathbb{Int}$ tal que $\nu = a \cdot k$ (2)
- Multiplique (μx) em (2) em ambos os lados
- Efetue propriedade comutativa
- Tome $(k \cdot \mu x)$ como testemunho que $\exists x : \mathbb{Z}. \nu = ax$

Veja bem como escrevemos cálculos dentro de demonstrações

Respostas

A. ① Person × Lang → Prop ✓

4⁺ ② ~~Int~~ × Prop → ~~Prop~~

③ Prop → ~~Prop~~

④ Person ✓

⑤ Prop × ~~Var~~ → Prop

⑥ Prop → ~~Prop~~

⑦ Type × Prop → ~~Prop~~

⑧ City × Nat → Prop ✓

B. Seja P : ~~Int~~ : Prop.

1 Seja Q : ~~Int~~ : Prop.

Escolho ~~b~~ -2. ✓

~~Caso L:~~

Suponha ¬(P&Q). ✓

Ou seja, P&Q ⇒ ⊥. ✓

~~Suponha P.~~

~~Ou seja, P ⇒ ⊥.~~

Extraio P. ✗

Extraio Q. ✗

Trago P⇒⊥ como minha testemunha. ✗

~~Caso R:~~

~~Suponha Q.~~

~~Ou seja, Q ⇒ ⊥.~~

enunciado ✓

não tens a conjunção P&Q nos teus dados.

teu alvo não é um existencial

ALSO: Testemunhas são objetos, não proposições.

C. Sejam a, u, v : Int. ✓

4⁻ Seja x : int. ✓

Suponha a|u ou a|v. ✓

Logo separo em casos.

Caso -L:

Seja k : int ✗

Calculemos:

a|u ⇒ ak = u ✗

isso é uma proposição seq

veja bem como escrevemos cálculos.

teu alvo não é um ∀...

Caso -R:

Seja l : int

Calculemos:

a|v ⇒ al = v

como que um arbitrário inteiro acabou sendo tão legal?

A. (1) Person × Lang → Prop. ✓

7 (2) Var × Prop → Cmd. ✓

(3) Prop → Cmd ✓

(4) Person. ✓

(5) Prop × ~~Var~~ → Prop. ~

(6) Prop → Cmd. ✓

(7) Type × Prop → ~~Prop~~. ~

(8) City × Nat → Prop. ✓

B) Demonstração:

8+ Demonstrarei ¬P ou ¬Q ⇒ ¬(P & Q). (Escolho-R). ✓

Suponha ¬P ou ¬Q. ✓

Suponha P & Q. (1) ✓

Sep. em casos pelo ou.

Caso ¬P: ✓

Extraia -L de (1). ✓

Contradição. ✓

Caso ¬Q: ✓

Extraia -R de (1). ✓

Contradição. ■

C) Demonstração:

6+ Sejam x, y : Int. ✓

Suponha a | x ou a | y. ✓

Seja n : Int. ✓

Seja i : Int tal que ai = x. ✗

Seja f Int tal que af = y. ✗

Caso a | x:

Calculamos:

ai = x (pela escolha de i)

a(i.n) = x(y.n)

a(i.y.n) = (xy)n (Associatividade)

não é um cálculo afirmação ser

Temos (iyn) como testemunha. ✓

Caso a | y:

Calculamos:

af = y

af(xn) = y(xn)

a(fxn) = (yx)n

Temos (fxn) como testemunha ✓

■

Resposta

A.

2*

1: prisoner → long ✓

2: van → prop ✓

3: prop ✓

4: prisoner ✓

5: prop → int ✓

6: prop ✓

7: type → prop ✓

8: city × nat ✓

C.

1

~~Seja a inteiro~~

Sejam u, v inteiros ✓

Suponho alu valv ✓

Extraio alu de alu v XX → (i) não tens alu nos dados
(ii) alu não é uma conjunção.

alv X afirmação seca.

Seja b inteiro tal que

ab = u

Extraio alu de alu v XX

alv

Seja c inteiro tal que

ac = v

(A) ← o que é isso? RESPOSTAS

1: PERSON ∈ LANG. —

2: ~~INT~~ → PROP. —

3: PROP. —

4: PERSON. ✓

5: PROP ∈ INT.

6: PROP. —

7: ~~INT~~ → PROP. —

8: CITY ∈ NAT.

2

(B)

0

SEJA P: PROP.

SEJA Q: PROP. enunciado

SUPONHA P. ✗ teu alvo não tem a forma P⇒...

SUPONHA Q. ✗ —— Q⇒ ..

SUPONHA ¬(P & Q) ✗ —— ¬(P&Q)⇒...

SPLIT. ✗ teu alvo não tem a forma ...&...

CASO P: ✗ ...

TEMOS QUE ¬P.

CASO Q:

TEMOS QUE ¬Q.

OU SEJA TEMOS QUE ¬P OU ¬Q.

PORTANTO ¬(P & Q) ⇒ ¬P OU ¬Q.

LOGO (¬(P & Q) ⇒ ¬P OU ¬Q) OU (¬(P & Q) ⇐ ¬P OU ¬Q) ✗

(C)

SEJA A: INT.

0

SEJA K: INT. ← arbitrário

SUPONHA QUE V∈INT TAL QUE A.K = V ✗ Não vou supor isso.

SEJA S: INT

SUPONHA QUE V∈INT TAL QUE A.S = V ✗

SUPONHA UM X: INT → não é uma proposição.

CALCULEMOS

Respostas

(A)

1. person × Lang → Prop ✓
2. Var × Prop → Cmd ✓
3. Prop → Cmd ✓
4. Person ✓
5. Prop × int → Prop ✓
6. Prop → ~~Prop~~ —
7. ~~int~~ × Prop → ~~Prop~~ —
8. City × Nat → Prop ✓

(B)

Demonstração:

Escolha demonstrar $\neg(P \& Q) \Rightarrow \neg P$ ou $\neg Q$.

Suponha $\neg(P \& Q)$. ✓

Escolha demonstrar $\neg P$. ✓

(Aqui teriz como fazer mais um passo.)

(C)

Demonstração:

Sejam u, v inteiros. ✓

Suponha a/u ou a/v. ✓

Seja x inteira. ✓

Separe em casos. [pela ...] ✓

Caso a/u: ✓

Seja l inteira tal que $u = a.l$ ✓

Calculamos:

$(uv)x = ((la)v)x$ (Pela escolha de l)

$= a(lvx)$?

Nesse caso, temos que $a/(uv)x$ com (lvx) de testemunha ✓

Caso a/v:

Seja i inteira tal que $v = i.a$

Calculamos:

$(uv)x = (u(ia))x$

$= a(uia)$

Com (uia) test temos $a/(uv)x$ ✓

por que nomes diferentes?

e, escolhendo diferentes, por que não vizinhos?

FORA DO PRAZO -2

RESPOSTAS



A) 1) Person → Long —
2) x → Cmd —
3) String → Var —
4) Prop —
5) Real → Int, x —
6) Counter → Int —
7) Prop —
8) City → Prop —

B) Sejo P ⇒ Prop
Sejo Q ⇒ Prop
Sejo P & Q ⇒ Int

C) u : int
V : Prop

a | (u v) x : Type

G

A

1. Pbinom × Long → Prop ✓
2. VAR × Prop → Cmd ✓
3. Int → Prop ?! ✗
4. Pbinom ✓
5. Prop × Int → Cmd ✓
6. Prop → Cmd ✓
7. Types × Prop → Cmd ✓
8. ~~Contra~~ × ~~Prop~~ ⇒ ~~Cmd~~

FORA DO PRAZO

-2

⇒ e → são diferentes

B

0 Demonstração:

Escolho L: — ?

Suponho (A P ou A Q) ?

Split (A P & A Q) ?

Case A P:

Suponho A P: ✗

Case A B:

Suponho A Q ✗

A e ¬ são diferentes

C

0 Demonstração

Escolho R: ✗ teu alvo não é uma disjunção!

Pick diff [∃ K: Int] [Q K = V];

Defax Int to Q V = K; ✗ type error: (Prop) = (obj)

Instances XV;

Calculemos

Q V = K
Q V = K) 2 firmações seguir

Q (X V) K

End.