

Università degli Studi di Bologna

Corso di Laurea in Informatica
Esame scritto di LOGICA PER L'INFORMATICA (9/6 CFU)
29/06/2022

Scrivere **nome, cognome, numero di CFU e numero di matricola in alto a destra** in tutti i fogli protocollo. Gli esercizi con un doppio punteggio riportano prima il punteggio nel caso dell'esame da 9 CFU e poi quello nel caso di esame da 6 CFU. Fare attenzione all'esercizio 10 che è diverso a seconda del numero di CFU.

1 (1 punto). Dare la sintassi per le formule della logica proposizionale.

2 (4 punti/6 punti). Considerare la seguente grammatica che descrive alberi di interi, dove un nodo $\langle n, l \rangle$ ha una lista di figli l ed è etichettato dall'intero n :

$$\begin{aligned} T &::= \langle \mathbb{N}, L \rangle \\ L &::= T :: L \mid [] \end{aligned}$$

Scrivere, per ricorsione strutturale, una funzione $f(T)$ che, dato in input un albero T , restituisca il cammino $n_1 :: \dots :: n_k :: []$ dove n_1 è l'etichetta della radice di T , n_k l'etichetta di una foglia dell'albero e il cammino scelto a ogni passo il figlio la cui etichetta sia massima fra tutti i fratelli.

Esempio:

$$\begin{aligned} f(\langle 3, \\ \langle 4, \langle 2, [] \rangle :: \langle 6, [] \rangle :: \langle 5, [] \rangle :: [] \rangle :: \\ \langle 5, \langle 0, [] \rangle :: \langle 2, [] \rangle :: \langle 1, [] \rangle :: [] \rangle :: \\ \langle 3, \langle 2, [] \rangle :: \langle 6, [] \rangle :: \langle 5, [] \rangle :: [] \rangle :: \\ [] \rangle) = 3 :: 5 :: 2 :: [] \end{aligned}$$

in quanto 3 è la radice, $5 = \max\{4, 5, 3\}$, $2 = \max\{0, 2, 1\}$ e 2 è una foglia.

È possibile utilizzare funzioni ausiliare e/o parametri ausiliari. Nel caso di uso di parametri ausiliari per la funzione principale f , specificare il valore iniziale da passare per risolvere il problema.

3 (5 punti). Dimostrare in teoria assiomatica degli insiemi che

$$\forall A \forall B (A \cup B \subseteq A \cap B \Rightarrow A = B)$$

Prima della dimostrazione riportare l'enunciato di tutti gli assiomi di teoria degli insiemi che usate nella dimostrazione.

- 4 (1 punto). Esprimere la formula $A \wedge B \Rightarrow C$ usando solo variabili proposizionali, negazioni e disgiunzioni.
- 5 (1 punto). Dare la definizione di relazione in teoria degli insiemi.
- 6 (1 punto). Elencare due formule tautologiche e due formule non tautologiche.
- 7 (5 punti). Considerare la seguente funzione che elimina duplicati consecutivi da una lista:

```
uniq [] = []  
uniq (x :: l) = if isHead x (uniq l) then uniq l else x :: uniq l
```

dove

```
isHead x [] = false  
isHead x (y :: l) = x == y
```

Dimostrare, per induzione strutturale su l , che

- (a) per ogni x, l si ha che se `isHead x l = true` allora per ogni y si ha `isHead y l = true` sse $x = y$
- (b) per ogni x, l si ha `isHead x l = isHead x (uniq l)`

- 8 (6 punti). Si consideri il seguente ragionamento:

Se si vendono più auto elettriche, ma non si costruiscono più punti di ricarica allora le auto elettriche resteranno invendute. Le auto elettriche verranno comperate o la gente andrà a piedi. Quindi, se la gente non andrà a piedi, ma non verranno costruiti nemmeno più punti di ricarica, allora la vendita di auto elettriche non decollerà.

Verificare la correttezza del ragionamento utilizzando la deduzione naturale per la logica proposizionale. Preferire una prova intuizionista se possibile.

- 9 (1 punto/2 punti). Effettuare la seguente sostituzione minimizzando il numero di cambi di nome alle variabili.

$$(\prod_{i=0}^{\iota+1} \sum_{j=0}^{\iota} j + \alpha)[\iota + y/\alpha]$$

10 (2 punti). **PER CHI SOSTIENE L'ESAME DA 6 CFU**

Dimostrare il seguente teorema usando la deduzione naturale al prim'ordine, preferendo una prova intuizionista a una classica ove possibile. Considerare il prodotto associativo a destra.

$$(\forall y. \exists x. y \geq k^{-1} * x * k) \Rightarrow \forall x. \exists y. k * x \geq k^{-1} * y$$

10 (5 punti). **PER CHI SOSTIENE L'ESAME DA 9 CFU**

Si consideri la seguente dichiarazione di type class `Acquisto` che predica di un tipo `T` l'esistenza di una funzione `valuta` che valuti il prezzo del bene di tipo `T` e di una funzione `dimezza` che ne dimezza la quantità:

```
class Acquisto T where
  valuta :: T -> int
  dimezza :: T -> T
```

con la proprietà che, per ogni x di tipo T , `valuta (dimezza x) = (valuta x) / 2`

Considerare il seguente codice ricorsivo non strutturale che, data una lista di beni di un tipo `T` che soddisfa il predicato `Acquisto` e un prezzo massimo `max`, ritorna una lista di beni con quantità ridotte il cui prezzo non eccede `max`:

```
compra :: forall T, Acquisto T => int -> list T -> list T
compra max [] = []
compra max (x::tl) =
  if valuta x >= max then
    compra max (dimezza x :: tl)
  else
    x :: compra (max - valuta x) tl
```

- (a) Mostrare un'istanza della type class per $T = \text{int}$ dove il prodotto n ha valore n
- (b) Mostrare un'istanza della type class per $T = U \times V$ (T è il prodotto cartesiano dei tipi U e V), sotto l'ipotesi che U e V soddisfino a loro volta la type class e che il valore di una coppia sia la somma del valore delle componenti

- (c) Mostrare un'istanza della type class per $T = list\ U$ (T è una lista di elementi di tipo U), sotto l'ipotesi che U soddisfi a sua volta la type class `Acquisto` e che il valore di una lista sia la somma del valore di tutti i suoi elementi
- (d) Mostrare l'output della chiamata

```
compra 10 ( (<1,1> :: []) :: (<4,2> :: <2,2> :: []) :: [] )
```

ATTENZIONE: mostrare un'istanza non significa solamente implementare le funzioni richieste, ma anche dimostrare le proprietà richieste su tali implementazioni!