

Conceptual si aplicativ in programarea logica si functionala**Cuprins**

PARTEA I. Fundamentele programării logice. Inițiere în prolog

Cuvânt înainte

1. Ce anume a determinat inventarea paradigmei logice de programare?
 - 1.1. Pe scurt despre premisele teoretice ale programării logice
 - 1.2. Scurtă incursiune în universul conceptual și problematic al logicii matematice. Varianta boolean-aristotelică.
2. Prolog și programarea logică. Considerații preliminare
 - 2.1. Despre specificul programării logice
 - 2.2. Infrastructura obligatorie a unui ide prolog
 - 2.3. Sintaxa iso prolog
3. Standardul borland (turbo) prolog
 - 3.1. Scurtă introducere
 - 3.2. Structura de principiu a unui program turbo prolog
4. Un program relativ simplu în turbo prolog
5. Revenire asupra lucrului cu predicate în prolog
 - 5.1. Câteva predicate predefinite de interes general în turbo prolog
6. Recursivitatea în programarea prolog
 - 6.1. Elemente introductive
 - 6.2. Câteva exemple comentate
7. Lucrul cu liste în prolog
 - 7.1. De ce liste în genere și în prolog?
 - 7.2. Listele în turbo prolog
8. Pe larg despre alte predicate turbo prolog predefinite utile

- 8.1. Lucrul cu șiruri de caractere
- 8.2. Elemente introductive
- 8.3. Operații cu caractere și șiruri de caractere
- 8.4. Predicate aferente operațiilor i/o
- 8.5. Ferestre simple în aplicațiile turbo prolog
- 9. Trucuri utile în elaborarea programelor prolog complexe și robuste
 - 9.1. De ce este nevoie să vorbim despre trucuri în programarea prolog
 - 9.2. Trucuri în lucrul cu liste
 - 9.3. Iterarea în baza de fapte cu suportul motorului inferențial
 - 9.4. Iterarea în genere cu suportul motorului inferențial
 - 9.5. Iterarea cu număr definit de pași folosind recursivitatea
 - 9.6. Simularea alternativelor fiabile
 - 9.7. Simularea variabilelor globale
 - 9.8. Modularizarea codului prolog
 - 9.9. Tratarea defensivă a excepțiilor în prolog
- 10. Baze de fapte dinamice în programarea turbo prolog
 - 10.1. Revenire asupra faptelor și regulilor prolog
 - 10.2. Cadrul turbo prolog pentru adăugarea dinamică a faptelor
- 11. Lucrul cu fișiere în turbo prolog
 - 11.1. Un exemplu introductiv
 - 11.2. Predicate conexe lucrului cu fișiere turbo prolog
- 12. Structuri de date recursive în turbo prolog
- 13. Alte aplicații turbo prolog comentate
 - 13.1. Problema turnurilor din hanoi
 - 13.2. Tokenizarea unui fisier text
 - 13.3. Metode prolog de căutare a rutelor. Căutarea înainte și căutarea înapoi
 - 13.4. Simularea lucrului cu vectori în prolog
 - 13.5. Despărțirea cuvintelor în silabe
 - 13.6. Mini sistem expert
 - 13.7. Simularea unui automat cu număr finit de stări

- 13.8. Aplicație turbo prolog pentru simularea unui organizer. Schiță
 - 13.9. Aplicație prolog pentru căutarea drumurilor minime într-un graf
 - 13.10. Indexarea fisierelor in turbo prolog
 - 14. Pe scurt despre alte ide-uri care oferă suport pentru programarea prolog
 - 14.1. Swi-prolog
 - 14.2. Gnu prolog
 - 14.3. Amzi prolog
 - 14.4. Visual prolog
 - 14.5. Sicstus prolog
 - 14.6. Gnu prolog for java
 - 14.7. Ciao prolog
 - 15. Exemple recapitulative prolog
- Bibliografie selectivă – partea I
- Index exemple prolog
- Partea II. Inițiere în programarea funcțională. Perspectiva haskell
- Cuvânt înainte
- 1. Introducere
 - 1.1. De ce paradigma funcțională?
 - 1.2. De ce haskell?
 - 2. Fundamente relativ la mediile haskell de programare
 - 2.1. Pe scurt despre o parte dintre soluțiile alternative
 - 2.2. Utilizați ide winghci, livrat odată cu platforma haskell
 - 3. Ingrediente importante pentru programarea în haskell
 - 3.1. Tipuri de date fundamentale pentru noul venit în haskell
 - 3.2. Despre operatori în haskell
 - 3.3. Reprezentarea structurilor de prelucrare în haskell
 - 3.4. Listele haskell
 - 3.5. Recursivitatea în haskell
 - 3.6. Tipuri de date utilizator în haskell
 - 4. Operații i/o în haskell. Fluxurile standard

5. Compunerea funcțiilor haskell
 - 5.1. Despre funcții și compunerea lor în matematică
6. Funcții high order în haskell
7. Funcții lambda în haskell
8. Modularizarea programelor haskell
 - 8.1. Despre modularizare în genere
 - 8.2. Sintaxa de bază pentru definirea unui modul haskell
 - 8.3. Un exemplu de aplicație haskell cu accent mai amplu pe modularizare
 - 8.4. Alte detalii sintactice utile la modularizarea proiectelor hskell
 - 8.5. Sintetizarea efectelor produse de sintaxa aferentă importului de module în aplicațiile haskell
9. Operații i/o în haskell. Fișiere
 - 9.1. Prolegomene
 - 9.2. Conceptul de handle asociat unui fișier haskell
 - 9.3. Alte resurse predefinite importante pentru lucrul cu fișiere
 - 9.4. Alte funcții de interes când se lucrează cu fișiere în haskell
 - 9.5. Fișiere haskell cu înregistrare structurată de utilizator. O abordare alternativă
10. Clase de tipuri utilizator în haskell
 - 10.1. Tipuri de date încorporate (inbuilt)
 - 10.2. Tipuri de clase utilizator
11. Aplicații de tip consolă cu suport ansi-terminal
 - 11.1. Instalare ansi-terminal
 - 11.2. Capabilitățile ansi-terminal uzuale
12. Monadele haskell
 - 12.1. Elemente introductive
 - 12.2. Monade. Exemple și utilitate
13. Functori în haskell
 - 13.1. Elemente introductive
 - 13.2. Cum se ajunge la utilitatea functorilor. Studiu de caz
 - 13.3. Exemplu de creare a unei instanțe a clasei functor
14. Tratarea excepțiilor în aplicațiile haskell

14.1. Programarea defensivă

14.2. Tratarea anomaliilor cu suport sintactic dedicat

15. Exemple recapitulative haskell

Bibliografie selectivă / partea ii

Index exemple - haskell

Index de figuri