|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования  Российской Федерации | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования | | |
| «Новосибирский государственный технический университет» | | |
|  | | |
| Теоретической и прикладной математики | | |
|  | | |
| Лабораторная работа № 5 | | |
| по дисциплине «Логическое программирование» | | |
| ***Рекурсивные структуры данных – деревья и графы*** | | |
|  | | |
|  | | |
|  | Факультет: | ПМИ |
| Группа: | ПМИ-02 |
| Вариант: | 7 |
| Студент: | Сидоров Даниил, |
|  | Дюков Богдан |
| Преподаватель: | Авдеенко Т. В., |
|  | Целебровская М. Ю.  . |
|
|  |  |
|  | | |
| Новосибирск | | |
| 2022 | | |

1. **Цель работы**

Изучение логических программ, задающих отношения над объектами рекурсивных типов и приобретение навыков самостоятельной разработ­ки таких программ.

1. **Исследование**

**Программа для исследования**

domains

formula = formula(symbol, formula); suboperation(formula,formula); letter(symbol); true(); false()

predicates

nondeterm satisfiable(formula)

nondeterm invalid(formula)

clauses

satisfiable(formula("&", suboperation(X, Y))):- satisfiable(X), satisfiable(Y).

satisfiable(formula("v", suboperation(X, \_Y))):- satisfiable(X).

satisfiable(formula("v", suboperation(\_X, Y))):- satisfiable(Y).

satisfiable(formula("-", X)):- invalid(X).

satisfiable(true()).

invalid(formula("&", suboperation(X, \_Y))):- invalid(X).

invalid(formula("&", suboperation(\_X, Y))):- invalid(Y).

invalid(formula("v", suboperation(X, Y))):- invalid(X), invalid(Y).

invalid(formula("-", X)):- satisfiable(X).

invalid(false()).

**Тест исходной программы**

satisfiable(formula("&", suboperation(true, false))).

**Ответ: no**

satisfiable(formula("v", suboperation(true, false))).

**Ответ: yes**

satisfiable(formula("-", false)).

**Ответ: yes**

satisfiable(formula("&", suboperation(formula("-", false), false))).

**Ответ: no**

**Поиск решения**

satisfiable(formula("-", false))

invalid(false())

yes

**Модифицированная программа**

domains

formula = formula(symbol, formula); suboperation(formula,formula); letter(symbol); true(); false()

predicates

nondeterm satisfiable(formula)

nondeterm invalid(formula)

clauses

satisfiable(formula("&", suboperation(X, Y))):- satisfiable(X), satisfiable(Y),!.

satisfiable(formula("v", suboperation(X, \_Y))):- satisfiable(X),!.

satisfiable(formula("v", suboperation(\_X, Y))):- satisfiable(Y),!.

satisfiable(formula("-", X)):- invalid(X),!.

satisfiable(true()).

invalid(formula("&", suboperation(X, \_Y))):- invalid(X),!.

invalid(formula("&", suboperation(\_X, Y))):- invalid(Y),!.

invalid(formula("v", suboperation(X, Y))):- invalid(X), invalid(Y),!.

invalid(formula("-", X)):- satisfiable(X),!.

invalid(false()).

**Тест модифицированной программы**

satisfiable(formula("&", suboperation(true, false))).

**Ответ: no**

satisfiable(formula("v", suboperation(true, false))).

**Ответ: yes**

satisfiable(formula("-", false)).

**Ответ: yes**

satisfiable(formula("&", suboperation(formula("-", false), false))).

**Ответ: no**

**Поиск решения**

satisfiable(formula("-", false))

invalid(false()),!

yes

**Вывод: Предикат не повлиял на решение.**

1. **Задание**

7. Напишите программу, которая по данной логической формуле определяет, будет или нет данная формула тавтологией, т.е. формулой, верной для всех комбинаций истинностных значений ее аргументов.

Например, *((X& Y)* V *¬X)*V *¬ (У &Z)* - тавтология.

1. **Листинг программы**

domains

formula = formula(symbol, formula); suboperation(formula,formula); letter(symbol); true(); false()

formulalist = formula\*

alphabet = symbol\*

predicates

nondeterm satisfiable(formula)

nondeterm invalid(formula)

determ isTautology(formula, alphabet)

unification(formula,symbol,formula,formula)

unificationFormula(formulalist,alphabet,formula,formula)

nondeterm value(formula)

length(alphabet,integer,integer)

nondeterm allVariationsOfValues(integer,formulalist)

clauses

value(true()).

value(false()).

satisfiable(formula("&", suboperation(X, Y))):- satisfiable(X), satisfiable(Y),!.

satisfiable(formula("v", suboperation(X, \_Y))):- satisfiable(X),!.

satisfiable(formula("v", suboperation(\_X, Y))):- satisfiable(Y),!.

satisfiable(formula("-", X)):- invalid(X),!.

satisfiable(true()).

invalid(formula("&", suboperation(X, \_Y))):- invalid(X),!.

invalid(formula("&", suboperation(\_X, Y))):- invalid(Y),!.

invalid(formula("v", suboperation(X, Y))):- invalid(X), invalid(Y),!.

invalid(formula("-", X)):- satisfiable(X),!.

invalid(false()).

unificationFormula([TrueOrFalse|TrueFalseListTail],[Letter|AlphabetTail],Formula,UnificationFormula):-

!,unification(TrueOrFalse,Letter,Formula,SelfResult), unificationFormula(TrueFalseListTail,AlphabetTail,SelfResult,UnificationFormula).

unificationFormula(\_,\_,Formula,Formula).

unification(TrueOrFalse,Letter,letter(Letter),TrueOrFalse):- !.

unification(TrueOrFalse,Letter,suboperation(X,Y),suboperation(X1,Y1)):- !,unification(TrueOrFalse,Letter,X,X1),unification(TrueOrFalse,Letter,Y,Y1).

unification(TrueOrFalse,Letter,formula(Z,X),formula(Z,X1)):- !,unification(TrueOrFalse,Letter,X,X1).

unification(\_,\_,Formula,Formula).

length([\_|AlphabetTail],N,Length):- N1=N+1,length(AlphabetTail,N1,Length).

length([],Length,Length).

allVariationsOfValues(0,[]):- !.

allVariationsOfValues(Length,[TrueOrFalse|TrueFalseListTail]):-value(TrueOrFalse),Length1=Length-1,allVariationsOfValues(Length1,TrueFalseListTail).

isTautology(Formula,Alphabet):-

length(Alphabet,0,Length),allVariationsOfValues(Length,TrueFalseList),unificationFormula(TrueFalseList,Alphabet,Formula,UnificationFormula),not(satisfiable(UnificationFormula)),!,fail;true.

1. **Результаты выполнения**

*((X& Y)* V *¬X)* V *¬ (У &Z)*

isTautology(formula("v", suboperation(formula("v", suboperation(formula("&", suboperation(letter("X"), letter("Y"))),formula("-", letter("X")))), formula("-", formula("&", suboperation(letter("Y"),letter("Z")))))), ["X", "Y", "Z"]).

Ответ: yes.

*(¬A* V *A)*

isTautology(formula("v", suboperation(formula("-", letter("A")), letter("A"))), ["A"]).

Ответ: yes.

*(A* V *A)*

isTautology(formula("v", suboperation(letter("A"), letter("A"))), ["A"]).

Ответ: no.

*(¬A* V *A)* *&* *(¬B* V *B)*

isTautology(formula("&", suboperation(formula("v", suboperation(formula("-", letter("A")), letter("A"))), formula("v", suboperation(formula("-", letter("B")), letter("B"))))), ["A", "B"]).

Ответ: yes.

1. **Описание задач, решаемых данной программой**

satisfiable(formula("&", suboperation(X, Y))):- satisfiable(X), satisfiable(Y),!.

satisfiable(formula("v", suboperation(X, \_Y))):- satisfiable(X),!.

satisfiable(formula("v", suboperation(\_X, Y))):- satisfiable(Y),!.

satisfiable(formula("-", X)):- invalid(X),!.

satisfiable(true()).

**Клауза истинна, если операция истинна.**

invalid(formula("&", suboperation(X, \_Y))):- invalid(X),!.

invalid(formula("&", suboperation(\_X, Y))):- invalid(Y),!.

invalid(formula("v", suboperation(X, Y))):- invalid(X), invalid(Y),!.

invalid(formula("-", X)):- satisfiable(X),!.

invalid(false()).

**Клауза истинна, если операция выдает ложь.**

length([\_|AlphabetTail],N,Length):- N1=N+1,length(AlphabetTail,N1,Length).

length([],Length,Length).

**Количество букв в выражении.**

allVariationsOfValues(0,[]):- !.

allVariationsOfValues(Length,[TrueOrFalse|TrueFalseListTail]):-value(TrueOrFalse),Length1=Length-1,allVariationsOfValues(Length1,TrueFalseListTail).

**Создает список значений букв. Для каждой буквы true и false.**

unificationFormula([TrueOrFalse|TrueFalseListTail],[Letter|AlphabetTail],Formula,UnificationFormula):-

!,unification(TrueOrFalse,Letter,Formula,SelfResult), unificationFormula(TrueFalseListTail,AlphabetTail,SelfResult,UnificationFormula).

unificationFormula(\_,\_,Formula,Formula).

unification(TrueOrFalse,Letter,letter(Letter),TrueOrFalse):- !.

unification(TrueOrFalse,Letter,suboperation(X,Y),suboperation(X1,Y1)):- !,unification(TrueOrFalse,Letter,X,X1),unification(TrueOrFalse,Letter,Y,Y1).

unification(TrueOrFalse,Letter,formula(Z,X),formula(Z,X1)):- !,unification(TrueOrFalse,Letter,X,X1).

unification(\_,\_,Formula,Formula).

**Вместо букв подставляем true и false.**

isTautology(Formula,Alphabet):-

length(Alphabet,0,Length),allVariationsOfValues(Length,TrueFalseList),unificationFormula(TrueFalseList,Alphabet,Formula,UnificationFormula),not(checkTrue(UnificationFormula)),!,fail;true.

**Перебираем все возможные варианты подстановки, если во всех вариантах выходит один ответ, то сработает true. В иных случаях false.**