Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра теоретической и прикладной информатики

Лабораторная работа №3

по предмету

«Введение в искусственный интеллект и логическое программирование**»**

Факультет: прикладной математики и информатики

Группа: ПМИ-12

Бригада: 4

Студенты: Михайловский М.А.

Швадченко А.В

Преподаватель: Целебровская М. Ю.

Новосибирск, 2024

**1. Задание**

1.

а) напишите на Турбо-Прологе программу (используя внутреннюю базу данных), позволяющую спрашивать у пользователя, каким языком он владеет, и записывать ответы в базу данных. За основу можно взять следующую схему, сделав соответствующие изменения:

ЯЗЫК (итальянский).

ЯЗЫК (немецкий).

ЯЗЫК (японский).

ЯЗЫК (французский).

ЯЗЫК (английский).

ДИАЛОГ:- WRITE (‘Введите Ваше имя :’),

READ (Имя) , ЯЗЫК(Яз),

WRITE (Знаете ли вы),

WRITE (Яз),

WRITE (‘язык’),

READ (да),

ASSERT (ВЛАДЕЕТ(Имя, Яз)),

FAIL.

б) в базу данных включите факты: ЯЗЫК(...), ВЛАДЕЕТ ( \_ , \_ );

в) измените программу, включив в нее предикаты чтения базы данных из файла и записи в файл по окончании сеанса работы.

2. Измените свое индивидуальное задание из лабораторной рабо¬ты № 2 таким образом, чтобы все основные факты вашей программы хра¬нились во внутренней базе данных (считывались из файла, обрабатывались программой и снова записывались в файл). При этом введите диалог с поль¬зователем для добавления или удаления фактов в базу данных (за основу организации диалога возьмите предыдущее задание 1).

**2.Листинг разработанной базы знаний на языке ПРОЛОГ**

**1 Задание**

domains

s = symbol

database

lang(s)

speak(s, s)

predicates

nondeterm main\_menu\_item(integer).

nondeterm main\_menu.

nondeterm repeat.

clauses

main\_menu\_item(1):-

write("Write your name: "),

readln(Name),

lang(Lang),

write("Do you know: ", Lang, "?\n"),

readln(Ans),

Ans = "yes",

assertz(speak(Name, Lang)),

fail.

main\_menu\_item(2):-

write("Write a new langugage: "), nl,

readln(Lang),

asserta(lang(Lang)), fail.

main\_menu\_item(3):-

write("Peoples and languages:"), nl,

speak(Name, Lang),

write(Name, " speak ", Lang, ";"), nl, fail.

main\_menu\_item(4):-

write("Languages:"), nl,

lang(Lang),write(Lang), nl, fail.

main\_menu\_item(5):-

save("C:\prolog\data.txt"),

write("The information is saved"), nl, fail.

main\_menu\_item(6):-

consult("C:\prolog\data.txt"),

write("Information extracted from the file"), nl, fail.

main\_menu\_item(7):-

!.

main\_menu :-

repeat,

write("====================================\n"),

write("Make a choice:\n"),

write("1 - add information about a person\n"),

write("2 - add language to database\n"),

write("3 - show all people\n"),

write("4 - show all languages\n"),

write("5 - save database in file\n"),

write("6 - load database from file\n"),

write("7 - exit the program\n"),

readint(Choice),

main\_menu\_item(Choice),

!.

repeat.

repeat:- repeat.

goal

main\_menu.

**2 Задание**

database

nondeterm flight(symbol, symbol, symbol, symbol, real, real, integer)

nondeterm day(symbol, integer)

predicates

nondeterm main\_menu\_item(integer).

nondeterm main\_menu.

nondeterm repeat.

nondeterm connection(symbol, symbol, integer, integer)

nondeterm connection2(symbol, symbol, integer, symbol, symbol, integer)

nondeterm cheapest\_flight(symbol, symbol, integer, integer)

nondeterm cheapest\_flight2(symbol, symbol, integer, integer)

nondeterm cheaper\_connection2(symbol, symbol, integer, integer, integer)

nondeterm cheaper\_connection(symbol, symbol, integer, integer, integer)

clauses

connection(X, Y, Cost, D) :-

flight(Fl, X, Y, \_, \_, \_, Cost), day(Fl, D).

connection2(X, Y, Cost, Flight1, Flight2,D1) :-

flight(Flight1, X, Z, \_, \_, Departure1, Cost1),

flight(Flight2, Z, Y, \_, Arrival1, \_, Cost2),

Departure1 < Arrival1,

day(Flight1, D1),

day(Flight2, D2), D1-D2=0,

connection(Z, Y, Cost2, \_),

Cost = Cost1 + Cost2 + 999.

connection2(X, Y, Cost, Flight1, Flight2,D1) :-

flight(Flight1, X, Z, \_, \_, Departure1, Cost1),

flight(Flight2, Z, Y, \_, Arrival1, \_, Cost2),

Departure1 < Arrival1,

day(Flight1, D1),

day(Flight2, D2), D1-D2=0,

connection2(Z, Y, Cost2,\_,\_,\_),

Cost = Cost1 + Cost2 + 999.

connection2(X, Y, Cost, Flight1, Flight2,D1) :-

flight(Flight1, X, Z, \_, \_, \_, Cost1),

flight(Flight2, Z, Y, \_, \_, \_, Cost2),

day(Flight1, D1),

day(Flight2, D2), abs(D1-D2)=1,

connection(Z, Y, Cost2, \_),

Cost = Cost1 + Cost2 + 999.

connection2(X, Y, Cost, Flight1, Flight2,D1) :-

flight(Flight1, X, Z, \_, \_, \_, Cost1),

flight(Flight2, Z, Y, \_, \_, \_, Cost2),

day(Flight1, D1),

day(Flight2, D2), abs(D1-D2)=1,

connection2(Z, Y, Cost2,\_,\_,\_),

Cost = Cost1 + Cost2 + 999.

cheaper\_connection2(X, Y, Cost, OtherCost, D) :-

connection2(X, Y, OtherCost, \_,\_, D),

OtherCost < Cost.

cheaper\_connection(X, Y, Cost, OtherCost, D) :-

connection(X, Y, OtherCost, D),

OtherCost < Cost.

cheapest\_flight(X, Y, Cost, D) :-

connection(X, Y, Cost,D),

not(cheaper\_connection(X, Y, Cost, \_, \_)).

cheapest\_flight2(X, Y, Cost, D) :-

connection2(X, Y, Cost,\_,\_,D),

not(cheaper\_connection2(X, Y, Cost, \_, \_)).

main\_menu\_item(1):-

write("Write flight number: "),

readln(Num),nl,

write("Write the city of departure: "),

readln(CityA),nl,

write("Write the city of arrival: "),

readln(CityB),nl,

write("Write the type of aircraft: "),

readln(Type),nl,

write("Write the departure time: "),

readreal(Time1),nl,

write("Write the arrival time: "),

readreal(Time2),nl,

write("Write the price: "),

readint(Price),nl,

assertz(flight(Num, CityA, CityB, Type, Time1, Time2, Price)),

fail.

main\_menu\_item(2):-

write("Write the flight number: "), nl,

readln(Num),nl,

write("Write the day of the week: "), nl,

readint(Day),nl,

asserta(day(Num, Day)), fail.

main\_menu\_item(3):-

write("Flights:"), nl,

flight(Num, CityA, CityB, Type, Time1, Time2, Price),

write(Num, " ",CityA, " ", CityB, " ", Type, " ", Time1, " ", Time2, " ", Price, ";"), nl, fail.

main\_menu\_item(4):-

write("Days:"), nl,

day(F, D),

write(F, " ", D, ";" ), nl, fail.

main\_menu\_item(5):-

save("C:\prolog\datapart2.txt"),

write("The information is saved"), nl, fail.

main\_menu\_item(6):-

consult("C:\prolog\datapart2.txt"),

write("Information extracted from the file"), nl, fail.

main\_menu\_item(7):-

write("Write the flight number: "), nl,

readln(Num),

retract(flight(Num, \_,\_,\_,\_,\_,\_)), retract(day(Num, \_)), fail.

main\_menu\_item(8):-

write("Write the city of departure: "), nl,

readln(C1),nl,

write("Write the city of arrival: "), nl,

readln(C2),nl,

connection2(C1, C2, C, F1, F2, D),

write("Price=", C, " ", F1, " ", F2, " day=", D),nl, fail.

main\_menu\_item(9):-

write("Write the city of departure: "), nl,

readln(C1),nl,

write("Write the city of arrival: "), nl,

readln(C2),nl,

flight(F, C1, C2, \_,\_,\_, C),

day(F, D),

write("Price=", C, " flight=", F, " day=", D),nl,

fail.

main\_menu\_item(10):-

write("Write the city of departure: "), nl,

readln(C1),nl,

write("Write the city of arrival: "), nl,

readln(C2),nl,

cheapest\_flight2(C1, C2, Cost, D),

connection2(C1, C2, Cost, F1, F2, D),

write("Price=", Cost, " first flight=", F1, "second flight2=", F2, " day=", D),nl,

fail.

main\_menu\_item(11):-

write("Write the city of departure: "), nl,

readln(C1),nl,

write("Write the city of arrival: "), nl,

readln(C2),nl,

cheapest\_flight(C1, C2, Cost, D),

flight(F, C1, C2, \_, \_, \_, Cost),

write("Price=", Cost, " flight=", F, " day=", D),nl,

fail.

main\_menu\_item(12):-

!.

main\_menu :-

repeat,

write("====================================\n"),

write("Make a choice:\n"),

write("1 - add information about flight\n"),

write("2 - add day\n"),

write("3 - show all flights\n"),

write("4 - show all days\n"),

write("5 - save database in file\n"),

write("6 - load database from file\n"),

write("7 - delete a flight\n"),

write("8 - finding a way with transfers\n"),

write("9 - finding a way without transfers\n"),

write("10 - finding the cheapest way with transfers\n"),

write("11 - finding the cheapest way without transfers\n"),

write("12 - exit the program\n"),

readint(Choice),

main\_menu\_item(Choice),

!.

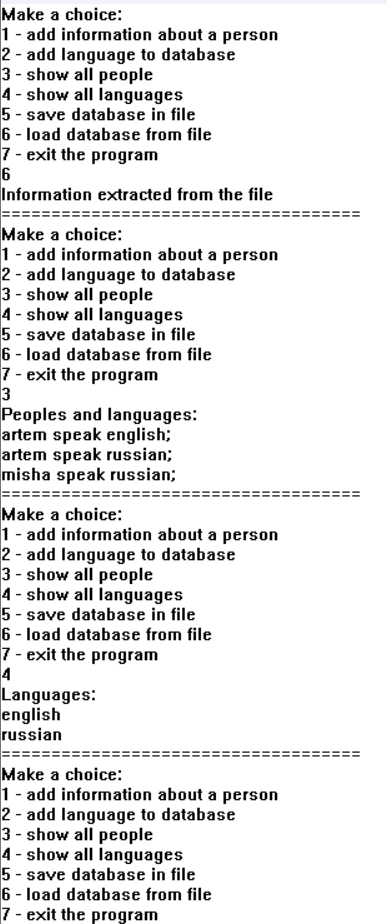
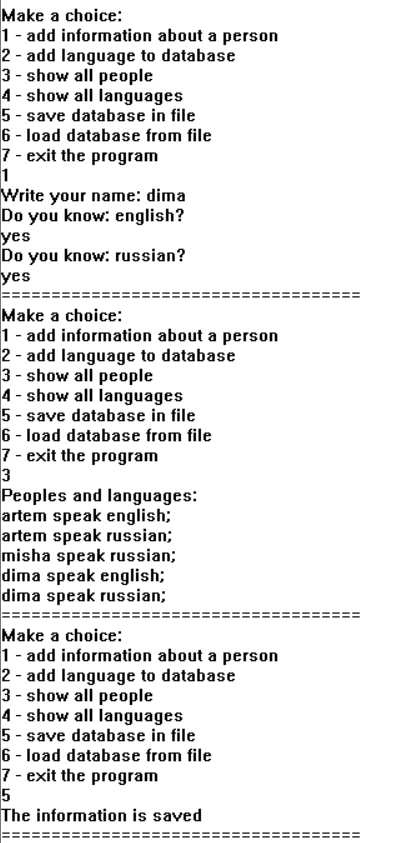
repeat.

repeat:- repeat.

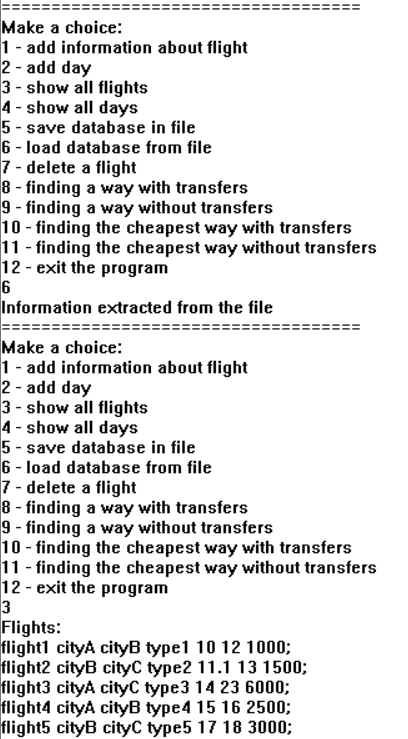
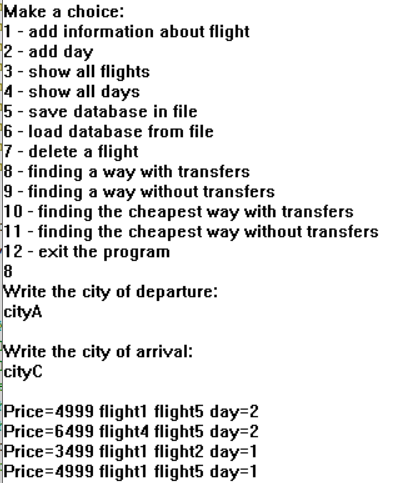
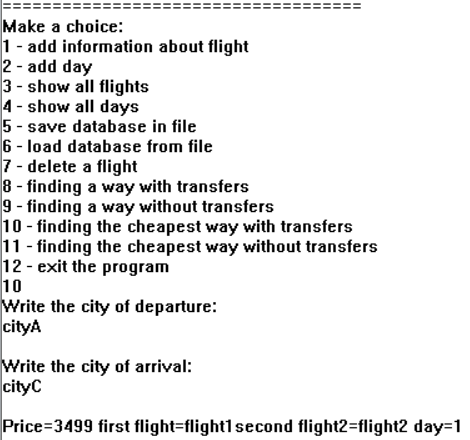
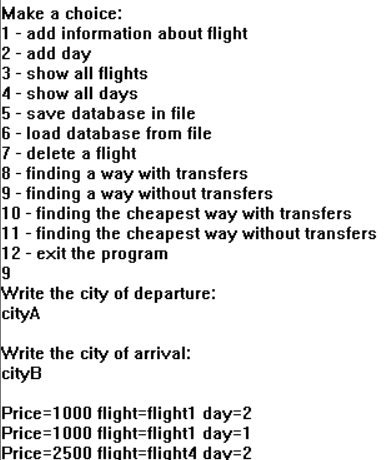
goal

main\_menu.

**3.Результаты выполнения**

1 программа

2 программа



cheapest\_flight2(cityA, cityC, C, D).

connection2 (X, Y, Cost,\_,\_,D)

{ cityA/ cityB,}

{ cityB/ cityC,}

{ cityA/ cityB,}

{ cityB/ cityC,}

{ cityA/ cityC,}

connection2(cityA, cityC, Cost, Flight1, Flight2,D1)

False

flight(Flight1, X, Z, \_, \_, Departure1, Cost1),

X={ cityA}

No

Z={cityA}

flight(Flight1, cityA, Z, \_, \_, Departure1, Cost1),

Z={cityB}

{cityC}

flight(Flight1, cityA, cityB, \_, \_, Departure1, Cost1),

flight(Flight1, cityA, cityC, \_, \_, Departure1, Cost1),

flight(Flight2, Z, Y \_, \_, Departure1, Cost1),

Y={ cityC}

flight(Flight1, Z, cityC, \_, \_, Departure1, Cost1),

Z={cityB}

{cityA}

Z={cityC}

No

flight(Flight1, Z, cityC, \_, \_, Departure1, Cost1),

Z={cityB}

Z={cityC}

{cityA}

day(Flight1, D1)

No

Flight1 = flight1

Cost = 1500

Cost = 3000

flight(flight2, cityB, cityC, \_, \_, \_, Cost)

flight(flight5, cityB, cityC, \_, \_, \_, Cost)

Fl = flight2

flight5

No

Fl = flight1

flight3

flight4

Z={cityB}

flight(Fl, cityB, cityC, \_, \_, \_, Cost)

Y={ cityC}

flight(Fl, Z, Y, \_, \_, \_, Cost)

connection(Z, Y, Cost2, \_)

2-2=0

D1 = 2

D2 = 2

D1-D2=0

day(flight2, 2)

D2 = 2

day(flight2, D2)

Flight2 = flight2

day(Flight2, D2)

day(flight1, 1)

day(flight1, 1)

day(flight1, D1)

not(cheaper\_connection2(cityA, cityC, 3499, \_, \_))

not(cheaper\_connection2(cityA, cityC, 5499, \_, \_))

X=cityA, Y = cityC, Cost = 3499

X=cityA, Y = cityC, Cost = 5499

not(cheaper\_connection2(X, Y, Cost, \_, \_))

cheapest\_flight2(cityA, cityC, 3499, 2)

cheapest\_flight2(cityA, cityC, 5499, 2)

X=cityA, Y = cityC, Cost = 3499 D = 2

X=cityA, Y = cityC, Cost = 5499, D = 2

cheapest\_flight2(X, Y, Cost, D)

Cost = 3499

Cost = 5499

Cost1 = 1000

Cost2 = 1500

Cost1 = 3000

Cost2 = 1500

Cost = Cost1 + Cost2 + 999

Cost = 1500

Cost = 3000

flight(flight2, cityB, cityC, \_, \_, \_, 1500)

flight(flight5, cityB, cityC, \_, \_, \_, 3000)

Yes

No

X=5499

**X=3499**

not(cheaper\_connection2(cityA, cityC, X, \_, \_))

not(cheaper\_connection2(cityA, cityC, X, \_, \_))

OtherCost = 3499

Cost = 5499

Yes

No

OtherCost = 5499

Cost = 3499

OtherCost<Cost

connection2(cityA, cityC, 3499, \_,\_, D),

connection2(cityA, cityC, 5499, \_,\_, D),

X=cityA, Y = cityC, OtherCost = 3499

X=cityA, Y = cityC, OtherCost = 5499

connection2(X, Y, OtherCost, \_,\_, D),