Выполнила студентка группы 09-832
Кузьмина Виктория

Инструкция по запуску приложения:

1. Разархивировать папку “*SportForKid_ES”*

2. В ней зайти в папку “*exe*” и запустить приложение “*SportForKid_ES.exe”*

Про интерфейс:

Экспертная система оформлена в виде классического консольного приложения, написанного на языке Prolog, а именно Visual Prolog 10. Среда разработки этой новой 10 версии (в отличие от всех предыдущих) дает возможность создать и графический интерфейс. Но инструменты управления его компонентами достаточно сложные и неочевидные, да и примеров их использования практически нет, поэтому я решила свои силы направить больше на содержание программы, а не на ее оформление.

Для чего нужна эта экспертная система:

Данная ЭС помогает родителям выбрать вид спорта для их ребенка, если они сами не могут определиться с подходящим занятием. Пользователю приложения предлагается отвечать на задаваемые вопросы (да или нет), после чего система может предложить подходящий под ответы вид спорта. Если родитель не согласен с предложением, то опрос продолжается, чтобы попробовать подобрать новый вариант.

Описание базы знаний:

База знаний находится в файле “*SportForKid_ES\SportForKid_catalog.dba”.* Именно этот файл использует приложение. База состоит из двух предикатов:
- cond хранит порядковый номер свойства и его описание,
- rule хранит название вида спорта и список номеров соответствующих ему свойств.

```
cond(18,"Вас устраивают достаточно дорогостоящие занятия").
cond(19,"Возрастная категория ребенка - 10 лет и меньше").
cond(20,"Ребенок любит соревноваться").
rule("спортивные бальные танцы",[19,18,1,4,6]).
rule("художественная гимнастика",[19,18,1,6,7]).
rule("спортивная гимнастика",[19,4,7,9]).
```

ЭС предлагает определенный вид спорта в качестве результата своей работы, только если на все вопросы о его свойствах пользователь ответил положительно.

Про пополнение базы знаний:

Особенностью системы является то, что если в базе нет информации о виде спорта, который бы подходил под указанные критерии, то у пользователя есть возможность пополнить базу знаний новым видом спорта, который будет соответствовать текущему набору критериев. Для этого система предлагает либо вписать свое название спорта, либо оставить это поле пока без названия (тогда система автоматически вписывает в поле «no_name»). После чего система предлагает добавить к новому спорту дополнительных свойств: сначала с помощью таких же ответов на вопросы, а потом с помощью ручного ввода.

Пример пополнения базы после опроса:

```
К сожалению, мы пока не знаем, какой вид спорта подходит под указанные параметры.
Но можно обновить информацию в базе знаний.

Если при прохождении опроса вы подразумевали некоторый вид спорта, пожалуйста, введите
его название и нажмите ENTER.
В противном случае просто нажмите ENTER
Ввод:

Были указаны следущие критерии:
Ребенок спокойно относится к воде и бассейнам
Возрастная категория ребенка - 10 лет и меньше
Важно ли указать ещё что-то? (1 - Да, 2 - Нет)1

Подходит спорт на льду? (1 - Да, 2 - Нет)2

У ребенка есть чувство ритма или танцевальные способности? (1 - Да, 2 - Нет)2

Одна из главных целей - развивать гибкость ребенка? (1 - Да, 2 - Нет)1

Ребенок готов к силовым нагрузкам? (1 - Да, 2 - Нет)2
```

……[ряд других вопросов о возможных свойствах]

```
Важны ли еще какие-либо критерии? (1 - Да, 2- Нет)1

Тогда напишите дополнительный параметр: Ребенок любит соревноваться

Важны ли еще какие-либо критерии? (1 - Да, 2- Нет)2

База знаний успешно обновлена.
```

Таким образом, в процессе использования приложения пользователи автоматически могут пополнять базу знаний новыми видами спорта и новыми свойствами.

Работа программы на примере:

*(без объяснений синтаксиса и механизма поиска языка Prolog)

При запуске приложения вызывается предикат run(), который инициализирует консоль, загружает данные в базу (из указанного файла), выводит вводную информацию… Первое существенное действие – вызов предиката choiceSport ().

```
choiceSport() :-
    rule(X, L),
    check(L),
    stdio::write("\nНам кажется, что для вашего ребенка подо
йдет следующий вид спорта: ", X),
    stdio::write("\nА вы согласны с таким выбором? (1 -
 Да, 2 - Нет)"),
    read_true_char(C),
    C = '1',
    !.
```

Этот предикат отвечает непосредственно за подбор вида спорта. Приложение выдаст пользователю свой вердикт только в том случае, если для вида спорта X будут положительно одобрены все его свойства L.

```
rule(X, L),
check(L),
```

Перебор видов спорта программа начинает сверху вниз по загруженной базе данных. В текущем файле с данными первым видом спорта являются спортивные бальные танцы.

```
rule("спортивные бальные танцы",[19,18,1,4,6]).
rule("художественная гимнастика",[19,18,1,6,7]).
```

Затем для списка порядковых номеров свойств L этого вида спорта вызывается предикат check.

```
check([H | T]) :-
    test_cond(H),
    check(T).
check([]).
```

Рекурсивно от списка свойств отделяется текущий первый элемент, для которого проверяется условие test_cond. В нашем примере отделяется номер свойства 19.

```
test_cond(H) :-
    cond_is(H, '1'),
    !. /* в базе имеется инфа о наличии критерия*/
```

```
test_cond(H) :-
    cond_is(H, '2'),
    !,
    fail. /* в базе имеется инфа об отсутствии критерия */
test_cond(H) :-
/* в базе нет никакой инфы о критерии, поэтому спрашиваем */
    cond(H, S),
    stdio::writef("\n%? (1 - Да, 2 - Нет)", S),
    read_true_char(A),
    assert(cond_is(H, A)),
    test_cond(H).
```

В нашем случае в базе ответов нет еще никакой информации о свойствах, поэтому первые 2 варианта данного условия не выполняются. 3й вариант по индексу свойства получает его описание S и с помощью консоли выводит вопрос о нем. В нашем примере под номером 19 следующее свойство:

```
cond(19,"Возрастная категория ребенка - 10 лет и меньше").
```

```
Данная ЭС может помочь выбрать наиболее подходящий вид спорта для вашего ребенка.
Для этого вам необходимо ответить на несколько вопросов о вашем ребенке и о предпочитетельных
критериях спортивных занятий.
После каждого ввода ответа нажмите ENTER.
***Обращаем ваше внимание на то, что результаты носят чисто рекомендательный характер.

Возрастная категория ребенка - 10 лет и меньше? (1 - Да, 2 - Нет)1
```

Предикат read_true_char(A) отвечает за то, чтобы считанный с консоли введенный символ являлся 1 или 2, в противном случае выведется сообщение об ошибке.

```
Возрастная категория ребенка - 10 лет и меньше? (1 - Да, 2 - Нет)цузщал
Ошибка ввода! Нажмите 1 или 2.
1
```

Следующим вызывается предикат assert, он является встроенным и отвечает за пополнение базы знаний текущей программы. В нашем примере в базе появится cond_is(19, ‘1’), то есть что на вопрос о свойстве 19 пользователь ответил положительно. В конце предиката test_cond есть рекурсивный вызов, и с помощью оператора отсечения в первом варианте данного предиката программа вернется к условию check(). На этом заканчивается обработка свойства 19.

По точно такой же схеме будут обрабатываться следующие свойства спортивных бальных танцев.

```
Возрастная категория ребенка – 10 лет и меньше? (1 – Да, 2 – Нет)1

Вас устраивают достаточно дорогостоящие занятия? (1 – Да, 2 – Нет)1

Подходит художественный вид спорта? (1 – Да, 2 – Нет)1

Ваш ребенок любит работать в одиночку? (1 – Да, 2 – Нет)2
```

Однако получив отрицательный ответ на вопрос, в предикате test_cond срабатывает оператор fail, поэтому вся работа с предикатом check для текущего списка свойств обрубается. Тогда choiceSport () берет в качестве спорта следующий по списку – художественная гимнастика.

```
rule("спортивные бальные танцы",[19,18,1,4,6]).
rule("художественная гимнастика",[19,18,1,6,7]).
```

```
Возрастная категория ребенка – 10 лет и меньше? (1 – Да, 2 – Нет)1

Вас устраивают достаточно дорогостоящие занятия? (1 – Да, 2 – Нет)1

Подходит художественный вид спорта? (1 – Да, 2 – Нет)1

Ваш ребенок любит работать в одиночку? (1 – Да, 2 – Нет)2

У ребенка есть чувство ритма или танцевальные способности? (1 – Да, 2 – Нет)
```

Первый вопрос задан по свойству с номером 6, потому что про предыдущие критерии (19, 18, 1) в списке уже имеются положительные ответы в текущей базе знаний. При положительных ответах на новые 2 вопроса система выносит вердикт – художественная гимнастика.

```
У ребенка есть чувство ритма или танцевальные способности? (1 – Да, 2 – Нет)1

Одна из главных целей – развивать гибкость ребенка? (1 – Да, 2 – Нет)1

Нам кажется, что для вашего ребенка подойдет следующий вид спорта: художественная гимнастика
А вы согласны с таким выбором? (1 – Да, 2 – Нет)
```

Код программы (язык Visual Prolog 10):

```
implement main

domains
    i = integer.
    s = string.
    ch = char.
    li = i*.

class facts - knowledge
    cond : (i, s). /* критерии */
    rule : (s, li).

class facts - dialog
    cond_is : (i, ch).
/* номер критерия; '1' - есть, '2' - нет*/

class predicates
    choiceSport : () nondeterm.
clauses
    choiceSport() :-
        rule(X, L),
        check(L),
        stdio::write("\nНам кажется, что для вашего ребенка подо
йдет следующий вид спорта: ", X),
        stdio::write("\nА вы согласны с таким выбором? (1 -
 Да, 2 - Нет)"),
        read_true_char(C),
        C = '1',
        !.
    choiceSport() :-
        stdio::write("\nК сожалению, мы пока не знаем, какой вид
 спорта подходит под указанные параметры. \n"),
        stdio::write("Но можно обновить информацию в базе знаний
.\n"),
        update.

class predicates
    update : () nondeterm.
/* добавляет в базу информацию о новом спорте */
clauses
    update() :-
        stdio::write("\nЕсли при прохождении опроса вы подразуме
вали некоторый вид спорта, пожалуйста, введите его название и на
жмите ENTER. "),
        stdio::write("\nВ противном случае просто нажмите ENTER"
),
        stdio::write("\nВвод: "),
        S = stdio::readLine(),
        add_cond(L),
        S_len = string::length(S),
```

```
        if S_len = 0 then
            assert(rule("no_name", L))
/* добавляем вид спорта без названия*/
        else
            assert(rule(S, L))
/* добавляем вид спорта с введенным названием*/
        end if,
        file::saveUtf8(@"..\SportForKid_catalog.dba", knowledge)
,
        stdio::write("\nБаза знаний успешно обновлена.").

class predicates
    test_cond : (i) nondeterm.
clauses
    test_cond(H) :-
        cond_is(H, '1'),
        !. /* в базе имеется инфа о наличии критерия*/
    test_cond(H) :-
        cond_is(H, '2'),
        !,
        fail. /* в базе имеется инфа об отсутствии критерия */
    test_cond(H) :-
    /* в базе нет никакой инфы о критерии, поэтому спрашиваем */
        cond(H, S),
        stdio::writef("\n%? (1 - Да, 2 - Нет)", S),
        read_true_char(A),
        assert(cond_is(H, A)),
        test_cond(H).

class predicates
    check : (li) nondeterm.
clauses
    check([H | T]) :-
        test_cond(H),
        check(T).
    check([]).

class predicates
    add_cond : (li) nondeterm anyflow.
/* возвращает список критериев, имеющихся у нового спорта */
clauses
    add_cond(L) :-
        cond_is(_, '1'),
        !,
        stdio::write("\nБыли указаны следущие критерии: "),
        print_cond(1, [], L1),
        stdio::write("\nВажно ли указать ещё что-то? (1 -
 Да, 2 - Нет)"),
        read_true_char(C),
        read_cond(C, L1, L).
    add_cond(L) :-
        read_cond('1', [], L).
```

```
class predicates
    print_cond : (i, li, li) nondeterm anyflow.
/* добавляет в список номера критериев с ответами «да» */
clauses
    print_cond(H, L, L) :-
        not(cond(H, _)),
        !.
    print_cond(H, L, L1) :-
        cond_is(H, '1'),
        !,
        cond(H, T),
        H1 = H + 1,
        stdio::writef("\n%", T),
        print_cond(H1, [H | L], L1).
    print_cond(H, L, L1) :-
        H1 = H + 1,
        print_cond(H1, L, L1).

class predicates
    read_cond : (ch, li, li) anyflow.
/* добавляет в список номера критериев, о которых еще не спрашив
алось */
clauses
    read_cond('1', L, L2) :-
        ex_cond(1, L, L1, N),
        new_cond(N, L1, L2),
        !.
    read_cond(_, L, L) :-
        !.

class predicates
    ex_cond : (i, li, li, i) nondeterm anyflow.
/* добавляет в список номера имеющихся в базе критериев*/
clauses
    ex_cond(N, L, L, N) :-
        not(cond(N, _)),
        !.
    ex_cond(N, L, L1, N2) :-
        cond_is(N, _),
        !,
        N1 = N + 1,
        ex_cond(N1, L, L1, N2).
    ex_cond(N, L, L1, N2) :-
        cond(N, S),
        stdio::writef("\n%? (1 - Да, 2 - Нет)", S),
        read_true_char(A),
        wr_cond(A, N, L, L2),
        N1 = N + 1,
        ex_cond(N1, L2, L1, N2).

class predicates
    wr_cond : (ch, i, li, li) nondeterm anyflow.
clauses
```

```
    wr_cond('1', N, L, [N | L]) :-
        !.
    wr_cond('2', _, L, L) :-
        !.

class predicates
    new_cond : (i, li, li) anyflow.
/* добавляет номера и описания новых критериев в базу знаний */
clauses
    new_cond(N, L, L1) :-
        stdio::write("\nВажны ли еще какие-либо критерии? (1 -
 Да, 2- Нет)"),
        read_true_char(A),
        A = '1',
        !,
        stdio::write("\nТогда напишите дополнительный параметр:
"),
        S = stdio::readLine(),
        assert(cond(N, S)),
        N1 = N + 1,
        new_cond(N1, [N | L], L1).
    new_cond(_, L, L).

class predicates
    read_true_char : (ch [out]) determ.
/* читает символ с консоли, пока он не равен '1' или '2'*/
clauses
    read_true_char(C) :-
        Cstr = stdio::readLine(),
        C1 = string::subChar(Cstr, 0),
        test(C1, C).

class predicates
    test : (ch, ch [out]) determ.
clauses
    test(C, C) :-
        '1' <= C,
        C <= '2',
        '1' <= C,
        !.
    test(_, C) :-
        stdio::write("Ошибка ввода! Нажмите 1 или 2. \n"),
        Cstr = stdio::readLine(),
        C1 = string::subChar(Cstr, 0),
        test(C1, C).

clauses
    run() :-
        console::init(stream::unicode),
        file::consult(@"..\SportForKid_catalog.dba", knowledge),
        stdio::write("\nДанная ЭС может помочь выбрать наиболее
подходящий вид спорта для вашего ребенка.\n"),
        stdio::write("Для этого вам необходимо ответить на неско
```

```
лько вопросов о вашем ребенке и о предпочитетельных критериях сп
ортивных занятий.\n"),
        stdio::write("После каждого ввода ответа нажмите ENTER.\
n"),
        stdio::write("***Обращаем ваше внимание на то, что резул
ьтаты носят чисто рекомендательный характер.\n"),
        choiceSport, /* попытка определить */
        retractFactDB(dialog), /* очищаем текущую информацию */
        retractFactDB(knowledge), /* удаляем инфу из базы */
        stdio::write("\n\nПопробуем ещё раз? (1 - Да, 2 -
 Нет)"),
        read_true_char(C),
        C = '1',
        !,
        run.
    run() :-
        stdio::write("\nНадеемся, что смогли вам помочь!"),
        _ = stdio::readChar().

end implement main

goal
    mainExe::run(main::run).
```