МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Самарский национальный исследовательский университет  
имени академика С.П. Королева»  
(Самарский университет)   
  
  
Факультет информатики  
Кафедра программных систем  
  
Дисциплина  
**Логическое программирование  
  
  
  
ОТЧЕТ**по лабораторной работе №3

Тема: Получение структурированной информации из базы фактов.  
Вариант №10

Студент: Лапин К.С.   
Группа: 6303-020302D   
  
Преподаватель: Лобанков А.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Теоретические сведения**

http\_handler(Path, Closure, Options):

Регистрация пути Closure в качестве обработчика HTTP-запросов. Path – это абсолютный или относительный путь.

dynamic:

Информирует о том, что определение предиката(ов) может измениться во время выполнения (используя assert/1 и/или retract/1). Динамические предикаты можно превратить в статические, используя compile\_predicates/1.

assert/1:

Добавление предложения (факта или правила) в базу данных. Предикат asserta/1 вставляет в первую позицию, а assertz/1 в последнюю. Устаревший assert/1 эквивалентен assertz/1.

findall/3:

Создает список экземпляров, которые удовлетворяют условию. Возвращает пустой список, если нет совпадений.

retract/1:

Удаление предложения (факта или правила) из базы данных.

http\_redirect/3:

Перенаправление на новое местоположение.

reply\_html\_page/3:

Возвращает HTML-страницу, которой предшествует HTTP-заголовок, как того требует библиотека (http\_wrapper).

**Задание**

Задание 1

Создать базу данных о заданной предметной области в виде множества фактов языка Пролог (не менее 5 фактов). Информацию о каждом компоненте БД представить в виде структуры.

Задание 2

Разработать набор предикатов, осуществляющих взаимодействие с БД, при помощи которых можно реализовать все статические запросы, приведенные в варианте задания.

Задание 3

Используя HTTP server libraries написать CRUD (Create/Read/Update/Delete) для своей базы фактов. Предоставить пользователю возможность сбросить базу данных к исходному (тестовому) варианту заполнения.

Задание 4

Реализовать вывод результатов всех поисковых запросов к базе фактов на html страницы.

Задание 5

Оформить отчет. Ответить на контрольные вопросы. Предметная область и запросы следующие:

Предметная область – спортивные соревнования. Каждое соревнование может быть описано структурой: ранг соревнований, вид спорта, год проведения, страна проведения, список команд – участников. Команды - участники могут быть описаны следующей структурой: название команды, страна, результаты соревнований.

Результаты соревнований могут быть описаны списком структур: название команды – соперника, страна, тип результата (выигрыш, проигрыш, ничья). Реализовать следующие типы запросов:

1. Найти год, в который участвовало максимальное число команд, в заданном ранге соревнований;
2. Найти вид спорта, в котором выступает заданная команда;
3. Найти все команды, которые участвовали в Олимпийских играх по определенному виду спорта;
4. Найти все команды, участвовавшие в соревнованиях в заданном году;
5. Найти все команды определенной страны, у которых были выигрыши.

**Описание работы программы**

Вывод списка соревнований, команд и результатов реализован с помощью функции findall, создаются списки всех элементов структур competition, team, result. После эти списки выводятся в таблицу в html-странице.

Добавление и удаление людей и семей из базы данных реализовано с помощью assertz и retractall.

Запросы по варианту реализованы с помощью функции findall.

**Примеры работы программы**

На рисунках 1-21 приведены скриншоты работы лабораторной работы.

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание  
Рисунок 1 – Пример главной html-страницы

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 2 – Пример страницы добавления соревнования

Изображение выглядит как текст, снимок экрана, число, Шрифт

Автоматически созданное описание  
Рисунок 3 – Пример главной страницы после добавления соревнования

Изображение выглядит как текст, снимок экрана, Шрифт, дисплей

Автоматически созданное описание  
Рисунок 4 – Пример поиска соревнования для вывода таблицы

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 5 – Результат поиска соревнования

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 6 – Пример страницы поиска для изменения данных

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 7 – Пример результата поиска, где можно изменять данные

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 8 – Пример замены

Изображение выглядит как текст, снимок экрана, число, Шрифт

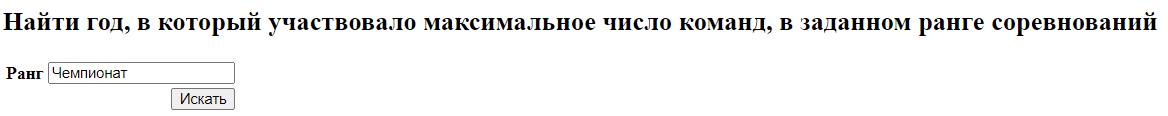
Автоматически созданное описание  
Рисунок 9 – Результат изменения

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание  
Рисунок 10 – Пример удаления соревнования

Изображение выглядит как текст, число, Параллельный, снимок экрана

Автоматически созданное описание  
Рисунок 11 – В результате запроса были удалены данные

  
Рисунок 12– Пример страницы запроса для задания 1

Изображение выглядит как текст, снимок экрана, Шрифт

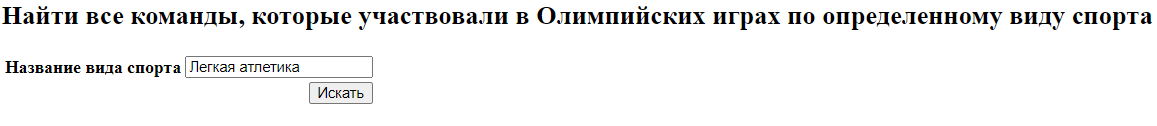
Автоматически созданное описание  
Рисунок 13 – Пример страницы результата запроса для задания 1

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 14 – Пример страницы запроса для задания 2

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание  
Рисунок 15 – Пример страницы результата запроса для задания 2

  
Рисунок 16 – Пример страницы запроса для задания 3

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание  
Рисунок 17 – Пример страницы результата запроса для задания 3

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 18 – Пример страницы запроса для задания 4

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание  
Рисунок 19 – Пример страницы результата запроса для задания 4

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 20 – Пример страницы запроса для задания 5

Изображение выглядит как текст, снимок экрана, Шрифт, линия

Автоматически созданное описание  
Рисунок 21 – Пример страницы результата запроса для задания 5

**Листинг программы**

% Предметная область – спортивные соревнования.

% Каждое соревнование может быть описано структурой: ранг соревнований, вид спорта,

% год проведения, страна проведения, список команд - участников. Команды

% - участники могут быть описаны следующей структурой: название команды,

% страна, результаты соревнований. Результаты соревнований могут быть

% описаны списком структур: название команды – соперника, страна, тип

% результата (выигрыш, проигрыш, ничья). Реализовать следующие типы

% запросов:

% 1 Найти год, в который участвовало максимальное число команд, в

% заданном ранге соревнований;

% 2 Найти вид спорта, в котором выступает заданная команда;

% 3 Найти все команды, которые участвовали в Олимпийских играх по

% определенному виду спорта;

% 4 Найти все команды, участвовавшие в соревнованиях в заданном году;

% 5 Найти все команды определенной страны, у которых были выигрыши.

% Библиотеки для работы с http

:-use\_module(library(http/http\_server)).

:-use\_module(library(http/http\_dispatch)).

:-use\_module(library(http/http\_parameters)).

:-use\_module(library(http/html\_write)).

% Библиотека для полного стектрейса ошибок

:- use\_module(library(http/http\_error)).

% Хэндлер для корневой страницы

:- http\_handler(root(.), home\_page, []).

% Хэндлер странички для ввода данных для поиска года, с максимальным

% числом команд

:- http\_handler(root(max\_teams\_year\_find), max\_teams\_year\_find\_page, []).

% Хэндлер поиска года, в который учавствовало максимальное число комад

:- http\_handler(root(max\_teams\_year), max\_teams\_year\_page, [method(post)]).

% Хэндлер странички для ввода данных для поиска вида спорта, в котором

% выступает заданная команда

:- http\_handler(root(sport\_by\_team\_find), sport\_by\_team\_find\_page, []).

% Хэндлер поиска вида спорта, в котором выступает заданная команда

:- http\_handler(root(sport\_by\_team), sport\_by\_team\_page, [method(post)]).

% Хэндлер странички для ввода данных для поиска всех команд, которые

% учавствовали в Олимпийских играх в определенном виде спорта

:- http\_handler(root(olympc\_teams\_by\_sport\_find), olympc\_teams\_by\_sport\_find\_page, []).

% Хэндлер поиска всех команд, которые учавствовали в Олимпийких играх в

% определенном виде спорта

:- http\_handler(root(olympc\_teams\_by\_sport), olympc\_teams\_by\_sport\_page, [method(post)]).

% Хэндлер странички для ввода данных для поиска всех команд, которые

% учавствовали в заданном году

:- http\_handler(root(teams\_in\_year\_find), teams\_in\_year\_find\_page, []).

% Хэндлер поиска всех команд, учавствоваших в соревнованиях в заданном

% году

:- http\_handler(root(teams\_in\_year), teams\_in\_year\_page, [method(post)]).

% Хэндлер странички для ввода данных для поиска всех команд определенной

% страны, у которых были выигрыши

:- http\_handler(root(winning\_teams\_by\_country\_find), winning\_teams\_by\_country\_find\_page, []).

% Хэндлер поиска всех команд определенной страны, у которых были

% выигрыши

:- http\_handler(root(winning\_teams\_by\_country), winning\_teams\_by\_country\_page, [method(post)]).

% Хэндлер страницы для добавления нового соревнования

:- http\_handler(root(add), add\_page, []).

% Хэндлер для добавления нового соревнования

:- http\_handler(root(add\_competition), add\_competition, [method(post)]).

% Хэндлер страницы для получения информации о соревновании, которое

% нужно считать

:- http\_handler(root(read), read\_page, []).

% Хэндлер для для чтения информации

:- http\_handler(root(read\_competition), read\_competition\_page, [method(post)]).

% Хэндлер страницы для удаления соревнований

:- http\_handler(root(delete), delete\_page, []).

% Хэндлер для удаления соревнования

:- http\_handler(root(delete\_competition), delete\_competition, [method(post)]).

% Хэндлер страницы для нахождения соревнований для обновления

:- http\_handler(root(update\_find), update\_find\_page, []).

% Хэндлер страницы для изменения данных

:- http\_handler(root(update), update\_page, [method(post)]).

% Хэндлер страницы для изменения данных

:- http\_handler(root(update\_competition), update\_competition, [method(post)]).

% Хэндлер для ресета БД

:- http\_handler(root(reset\_DB), resetDB, [method(post)]).

% Определеяем 5-ти местный предикат в БД

:- dynamic competition/5.

:- dynamic team/3.

:- dynamic result/4.

% Запуск сервера

server(Port):- http\_server(http\_dispatch, [port(Port)]).

server:-server(8080).

% Остановка сервера

stop(Port):- http\_stop\_server(Port, http\_dispatch).

stop:- stop(8080).

resetDB(\_):-

% Очистка БД

retractall(competition(\_, \_, \_, \_, \_)),

retractall(team(\_, \_, \_)),

retractall(result(\_, \_, \_, \_)),

% Инициализация БД

assertz(competition('Олимпийские игры', 'Легкая атлетика', '2024', 'Франция',

[team('Сборная США', 'США',

[result('Сборная Германии', 'Германия', 'Выигрыш'),

result('Сборная Ямайки', 'Ямайка', 'Выигрыш')]),

team('Сборная Германии', 'Германия',

[result('Сборная США', 'США', 'Проигрыш'),

result('Сборная Ямайки', 'Ямайка', 'Проигрыш')]),

team('Сборная Ямайки', 'Ямайка',

[result('Сборная США', 'США', 'Проигрыш'),

result('Сборная Германии', 'Германия', 'Выигрыш')])

])),

assertz(competition('Чемпионат', 'Футбол', '2020', 'Англия',

[team('Сборная Англии', 'Англия',

[result('Сборная Франции', 'Франция', 'Выигрыш'),

result('Сборная Испании', 'Испания', 'Проигрыш')]),

team('Сборная Франции', 'Франция',

[result('Сборная Англии', 'Англия', 'Проигрыш'),

result('Сборная Испании', 'Испания', 'Проигрыш')]),

team('Сборная Испании', 'Испания',

[result('Сборная Англии', 'Англия', 'Выигрыш'),

result('Сборная Фрнации', 'Франция', 'Выигрыш')])

])),

assertz(competition('Чемпионат', 'Баскетбол', '2021', 'США',

[team('Сборная США', 'США',

[result('Сборная Канады', 'Канада', 'Выигрыш'),

result('Сборная Мексики', 'Мексика', 'Выигрыш'),

result('Сборная Франции', 'Франция', 'Проигрыш')]),

team('Сборная Канады', 'Канада',

[result('Сборная США', 'США', 'Проигрыш'),

result('Сборная Мексики', 'Мексика', 'Ничья'),

result('Сборная Франции', 'Франция', 'Выигрыш')]),

team('Сборная Мексики', 'Мексика',

[result('Сборная США', 'США', 'Проигрыш'),

result('Сборная Канады', 'Канада', 'Ничья'),

result('Сборная Франции', 'Франция', 'Выигрыш')]),

team('Сборная Франции', 'Франция',

[result('Сборная США', 'США', 'Выигрыш'),

result('Сборная Канады', 'Канада', 'Проигрыш'),

result('Сборная Мексики', 'Мексика', 'Проигрыш')])

])),

asserta(competition('Дружеский матч', 'Футбол', '2022', 'Россия',

[team('Зенит', 'Россия',

[result('Спартак', 'Россия', 'Выигрыш')]),

team('Спартак', 'Россия',

[result('Зенит', 'Россия', 'Проигрыш')])

])),

/\*

asserta(competition('Турнир', 'Волейбол', 2023, 'Бразилия',

[team('Нова Серрана', 'Бразилия',

[result('Эквип Джекинс', 'Бразилия', 'Проигрыш')]),

team('Эквип Джекинс', 'Бразилия',

[result('Нова Серрана', 'Бразилия', 'Выигрыш')])

])),\*/

assertz(competition('Турнир', 'Волейбол', '2023', 'Бразилия',

[team('Нова Серрана', 'Бразилия',

[result('Эквип Джекинс', 'Бразилия', 'Проигрыш'),

result('Трез Волей', 'Бразилия', 'Выигрыш')]),

team('Эквип Джекинс', 'Бразилия',

[result('Нова Серрана', 'Бразилия', 'Выигрыш'),

result('Трез Волей', 'Бразилия', 'Ничья')]),

team('Трез Волей', 'Бразилия',

[result('Эквип Джекинс', 'Бразилия', 'Ничья'),

result('Нова Серрана', 'Бразилия', 'Проигрыш')])

])),

assertz(competition('Олимпийские игры', 'Борьба', '2016', 'США',

[team('Команда США', 'США',

[result('Команда Япония', 'Япония', 'Проигрыш')]),

team('Команда Россия', 'Россия',

[result('Команда Дания', 'Дания', 'Выигрыш')]),

team('Команда Дания', 'Дания',

[result('Команда Россия', 'Россия', 'Проигрыш')]),

team('Команда Япония', 'Япония',

[result('Команда США', 'США', 'Выигрыш')]),

team('Команда Куба', 'Куба',

[result('Команда Турция', 'Турция', 'Выигрыш')]),

team('Команда Турция', 'Турция',

[result('Команда Куба', 'Куба', 'Проигрыш')])

])),

http\_redirect(moved, '/', \_Request).

% Обработка главной страницы

home\_page(\_Request):-

% Собираем Ранг соревнований в список

findall(Rank, competition(Rank, \_, \_, \_, \_), Ranks),

% Собираем Ранг соревнований в список

findall(Sport, competition(\_, Sport, \_, \_, \_), Sports),

% Собираем Ранг соревнований в список

findall(Year, competition(\_, \_, Year, \_, \_), Years),

% Собираем Ранг соревнований в список

findall(Country, competition(\_, \_, \_, Country, \_), Countries),

% Собираем Ранг соревнований в список

%findall(Teams, competition(\_, \_, \_, \_, Teams), TeamsList),

% Добавляем заголовки таблицы

generate\_rows(Ranks, Sports, Years, Countries, Row),

% Выравниваем список

flatten(Row, FlattenRow),

ins(FlattenRow, tr(

[

th('Ранг Соревнования'),

th('Вид спорта'),

th('Год'),

th('Страна проведения'),

th('Команда'),

th('Страна'),

th('Оппонент'),

th('Страна оппонента'),

th('Результат')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица спортивных соревнований'),

[

h1('Спортивные соревнования'),

h3('Задания варианта'),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/max\_teams\_year\_find')], 'Task\_1'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/sport\_by\_team\_find')], 'Task\_2'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/olympc\_teams\_by\_sport\_find')], 'Task\_3'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/teams\_in\_year\_find')], 'Task\_4'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/winning\_teams\_by\_country\_find')], 'Task\_5'))

),

br(''),

h3('CRUD - CREAT/READ/UPDATE/DELETE'),

form(

[style('display: inline-block'), method(post)],

p(button([type(submit), formaction(location\_by\_id('resetDB'))], 'Сбросить БД'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/add')], 'Добавить соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/read')], 'Считать соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/update\_find')], 'Изменить соревнование'))

),

form(

[style('display: inline-block')],

p(button([type(submit), formaction('/delete')], 'Удалить соревнования'))

),

h3('Таблица соревнований'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Добавление элемента в начало списка

ins(L, El, [El|L]).

%generate\_rows([], [], [], [], []).

%generate\_rows([Rank|Ranks],

% [Sport|Sports],

% [Year|Years],

% [Country|Countries],

% [tr([td(Rank), td(Sport), td(Year), td(Country)])|Rows]):-

% generate\_rows(Ranks, Sports, Years, Countries, Rows).

generate\_rows([], []).

generate\_rows([Value|Values], [tr([td(Value)])|Rows]):-

generate\_rows(Values, Rows).

generate\_rows([], [], [], [], []).

generate\_rows([Rank|RestRanks],

[Sport|RestSports],

[Year|RestYears],

[Country|RestCountries],

[Rows|RestRows]):-

% Получаем список команд для данного Rank/Sport/Year/Country

competition(Rank, Sport, Year, Country, TeamsList),

% Генерируем строки для каждой команды и их результатов

generate\_team\_rows(Rank, Sport, Year, Country, TeamsList, Rows),

generate\_rows(RestRanks, RestSports, RestYears, RestCountries, RestRows).

generate\_team\_rows(\_, \_, \_, \_,[], []).

generate\_team\_rows(Rank,

Sport,

Year,

Country,

[team(TeamName, TeamCountry, Results)|RestTeams], [TeamRow|RestRow]):-

generate\_result\_rows(Rank, Sport, Year, Country, TeamName, TeamCountry, Results, TeamRow),

%format(atom(Row), 'Team: ~w, Country: ~w', [TeamName, TeamCountry]), % Выводим данные каждого матча

%write(Row), % Выводим данные каждого матча в консоль для отладки

generate\_team\_rows(Rank, Sport, Year, Country, RestTeams, RestRow).

generate\_result\_rows(\_, \_, \_, \_, \_, \_, [], []). % Базовый случай - завершение рекурсии

generate\_result\_rows(Rank,

Sport,

Year,

Country,

TeamName,

TeamCountry,

[result(Opponent, OpponentCountry, Outcome)|RestResults],

[tr([td(Rank), td(Sport), td(Year), td(Country), td(TeamName), td(TeamCountry),

td(Opponent), td(OpponentCountry), td(Outcome)])|RestRows]):-

%format(atom(Row), 'Match: ~w vs. ~w - ~w', [TeamName, Opponent, Outcome]), % Выводим данные каждого матча

%write(Row), % Выводим данные каждого матча в консоль для отладки

generate\_result\_rows(Rank, Sport, Year, Country, TeamName, TeamCountry, RestResults, RestRows).

% 1 Найти год, в который участвовало максимальное число команд, в

% заданном ранге соревнований;

maxTeamsYearList(Rank, YearRow) :-

findall(NumTeams, (competition(Rank, \_, Year, \_, Teams),

length(Teams, NumTeams)), NumTeamsList),

max\_list(NumTeamsList, MaxNumTeams),

findall(Year, (competition(Rank, \_, Year, \_, Teams),

length(Teams, MaxNumTeams)), YearList),

generate\_rows(YearList, YearRow).

max\_teams\_year\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод ранга соревнований'),

[form(

[action=location\_by\_id(max\_teams\_year\_page), method(post)],

[

h2('Найти год, в который участвовало максимальное число команд, в заданном ранге соревнований'),

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

max\_teams\_year\_page(Request):-

http\_parameters(

Request,

[

rank(Rank, [])

]

),

maxTeamsYearList(Rank, YearRow),

ins(YearRow, tr(

[

th('Год')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Год, с максимальным числом команд'),

[

h1('Год, с максимальным числом команд'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% 2 Найти вид спорта, в котором выступает заданная команда;

sportByTeam(TeamName, SportsRow) :-

%competition(\_, Sport, \_, \_, Teams),

%member(team(TeamName, \_, \_), Teams),

findall(Sport, (competition(\_, Sport, \_, \_, Teams),

member(team(TeamName, \_, \_), Teams)), Sports),

generate\_rows(Sports, SportsRow).

sport\_by\_team\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод названия команды'),

[form(

[action=location\_by\_id(sport\_by\_team\_page), method(post)],

[

h2('Найти вид спорта, в котором выступает заданная команда'),

table([

tr([th('Название команды'), td(input([name(teamName)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

sport\_by\_team\_page(Request):-

http\_parameters(

Request,

[

teamName(TeamName, [])

]

),

sportByTeam(TeamName, SportRow),

ins(SportRow, tr(

[

th('Вид спорта')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Вид спорта, в котором выступает заданная команда'),

[

h1('Вид спорта, в котором выступает заданная команда'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% 3 Найти все команды, которые участвовали в Олимпийских играх по

% определенному виду спорта.

olympicTeamsBySport(Sport, TeamsRow) :-

findall(Team,(competition('Олимпийские игры', Sport, \_, \_, TeamsList),

member(team(Team, \_, \_), TeamsList)), Teams),

generate\_rows(Teams, TeamsRow).

olympc\_teams\_by\_sport\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод названия спорта'),

[form(

[action=location\_by\_id(olympc\_teams\_by\_sport\_page), method(post)],

[

h2('Найти все команды, которые участвовали в Олимпийских играх по определенному виду спорта'),

table([

tr([th('Название вида спорта'), td(input([name(sport)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

olympc\_teams\_by\_sport\_page(Request):-

http\_parameters(

Request,

[

sport(Sport, [])

]

),

olympicTeamsBySport(Sport, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды, участвующие в Олимпийских играх в определенном виде спорта'),

[

h1('Команды, которые участвовали в Олимпийских играх в определенном виде спорта'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% 4 Найти все команды, участвовавшие в соревнованиях в заданном году;

teamsInYear(Year, TeamsRow) :-

%atom\_number(Year, YearNumber),

findall(Team, (competition(\_, \_, Year, \_, TeamsList),

member(team(Team, \_, \_), TeamsList)), Teams),

generate\_rows(Teams, TeamsRow).

teams\_in\_year\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод года'),

[form(

[action=location\_by\_id(teams\_in\_year\_page), method(post)],

[

h2('Найти все команды, участвовавшие в соревнованиях в заданном году'),

table([

tr([th('Год'), td(input([name(year), type(string)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

teams\_in\_year\_page(Request):-

http\_parameters(

Request,

[

year(Year, [])

]

),

teamsInYear(Year, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды, участвовавшие в соревнованиях в заданном году'),

[

h1('Команды, участвовавшие в соревнованиях в заданном году'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

% 5 Найти все команды определенной страны, у которых были выигрыши.

winningTeamsByCountry(Country, TeamsRow) :-

findall(Team,

(competition(\_, \_, \_, \_, TeamsList),

member(team(Team, Country, Results), TeamsList),

member(result(\_, \_, 'Выигрыш'), Results)),

Teams),

list\_to\_set(Teams, UniqueTeams),

generate\_rows(UniqueTeams, TeamsRow).

winning\_teams\_by\_country\_find\_page(\_Request):-

reply\_html\_page(

title('Ввод страны'),

[form(

[action=location\_by\_id(winning\_teams\_by\_country\_page), method(post)],

[

h2('Найти все команды определенной страны, у которых были выигрыши'),

table([

tr([th('Страна'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

winning\_teams\_by\_country\_page(Request):-

http\_parameters(

Request,

[

country(Country, [])

]

),

winningTeamsByCountry(Country, TeamsRow),

ins(TeamsRow, tr(

[

th('Название команды')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Команды определенной страны, у которых были выигрыши'),

[

h1('Команды определенной страны, у которых были выигрыши'),

table(

[border(2)],

Rows\_with\_Headers

)

]

).

add\_or\_update\_competition(Name, Sport, Year, Country, NewTeams) :-

competition(Name, Sport, Year, Country, ExistingTeams),

% Если соревнование уже существует

retract(competition(Name, Sport, Year, Country, ExistingTeams)),

append(ExistingTeams, NewTeams, UpdatedTeams),

asserta(competition(Name, Sport, Year, Country, UpdatedTeams)).

add\_or\_update\_competition(Name, Sport, Year, Country, Teams) :-

% Если соревнование не существует, добавляем новое соревнование в базу знаний

asserta(competition(Name, Sport, Year, Country, Teams)).

% CRUD (Create/Read/Update/Delete)для своей базы фактов.

% Cтраница с добавлением нового соревнования

add\_page(\_Request):-

reply\_html\_page(

title('Добавление соревнования'),

[form(

[action=location\_by\_id(add\_competition), method(post)],

[

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год'), td(input([name(year), type(number)]))]),

tr([th('Страна'), td(input([name(country)]))]),

tr([th('Команда'), td(input([name(teamName)]))]),

tr([th('Страна команды'), td(input([name(teamCountry)]))]),

tr([th('Команда соперника'), td(input([name(opponentName)]))]),

tr([th('Страна команды соперника'), td(input([name(opponentCountry)]))]),

tr([th('Результат соревнования'), td(input([name(result)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

% Предикат, который будет выполнять проверку и переприсвоение значения

% переменной

check\_and\_replace(Result, NewResult) :-

(Result = 'Выигрыш' -> NewResult = 'Проигрыш';

Result = 'Проигрыш' -> NewResult = 'Выигрыш';

NewResult = Result).

%Добавление нового соревнования

add\_competition(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, []),

teamName(TeamName, []),

teamCountry(TeamCountry, []),

opponentName(OpponentName, []),

opponentCountry(OpponentCountry, []),

result(Result, [])

]

),

check\_and\_replace(Result, NewResult),

add\_or\_update\_competition(Rank, Sport, Year, Country, [team(TeamName, TeamCountry, [result(OpponentName, OpponentCountry, Result)])]),

add\_or\_update\_competition(Rank, Sport, Year, Country, [team(OpponentName, OpponentCountry, [result(TeamName, TeamCountry, NewResult)])]),

%assertz(competition(Rank, Sport, Year, Country, [

% team(TeamName, TeamCountry, [result(OpponentName, OpponentCountry, Result)]),

% team(OpponentName, OpponentCountry, [result(TeamName, TeamCountry, NewResult)])])),

http\_redirect(moved, '/', \_Request).

read\_page(\_Request):-

reply\_html\_page(

title('Чтение соревнования'),

[form(

[action=location\_by\_id(read\_competition\_page), method(post)],

[

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год'), td(input([name(year), type(number)]))]),

tr([th('Страна'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Искать'])))

] )

]

)]

).

% Обработка главной страницы

read\_competition\_page(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, [])

]

),

% Добавляем заголовки таблицы

generate\_rows([Rank], [Sport], [Year], [Country], Row),

% Выравниваем список

flatten(Row, FlattenRow),

ins(FlattenRow, tr(

[

th('Ранг Соревнования'),

th('Вид спорта'),

th('Год'),

th('Страна проведения'),

th('Команда'),

th('Страна'),

th('Оппонент'),

th('Страна оппонента'),

th('Результат')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Таблица спортивных соревнований'),

[

h1('Спортивные соревнования'),

table(

[border(2)],

Rows\_with\_Headers

)

]).

% Cтраница с удалением соревнований

delete\_page(\_Request):-

reply\_html\_page(

title('Удаление соревнований'),

[form(

[action=location\_by\_id(delete\_competition), method(post)],

[

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год'), td(input([name(year), type(number)]))]),

tr([th('Страна'), td(input([name(country)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Удалить'])))

] )

]

)]

).

check\_competition(Rank, Sport, Year, Country):-

findall(Competition, competition(Rank, Sport, Year, Country, Competition), Competitions),

length(Competitions, Length),

Length \= 0 -> retractall(competition(Rank, Sport, Year, Country, \_)).

% Удаление соревнований

delete\_competition(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, [])

]

),

%check\_competition(Rank, Sport, Year, Country), % И так работает

retractall(competition(Rank, Sport, Year, Country, \_)),

http\_redirect(moved, '/', Request).

% Cтраница поиска соревнования для обновления

update\_find\_page(\_Request):-

reply\_html\_page(

title('Поиск соревнования'),

[form(

[action=location\_by\_id(update\_page), method(post)],

[

table([

tr([th('Ранг'), td(input([name(rank)]))]),

tr([th('Вид спорта'), td(input([name(sport)]))]),

tr([th('Год'), td(input([name(year), type(number)]))]),

tr([th('Страна'), td(input([name(country)]))]),

tr([th('Команда'), td(input([name(teamName)]))]),

tr([th('Страна команды'), td(input([name(teamCountry)]))]),

tr([th('Команда соперника'), td(input([name(opponentName)]))]),

tr([th('Страна команды соперника'), td(input([name(opponentCountry)]))]),

tr([th('Результат соревнования'), td(input([name(result)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Найти'])))

] )

]

)]

).

% Cтраница обновления данных

update\_page(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, []),

teamName(TeamName, []),

teamCountry(TeamCountry, []),

opponentName(OpponentName, []),

opponentCountry(OpponentCountry, []),

result(Result, [])

]

),

reply\_html\_page(

title('Изменение соревнования'),

[form(

[action=location\_by\_id(update\_competition), method(post)],

[

table([

tr([th('Команда'), td(input([name(teamName), value(TeamName)]))]),

tr([th('Страна команды'), td(input([name(teamCountry), value(TeamCountry)]))]),

tr([th('Команда соперника'), td(input([name(opponentName), value(OpponentName)]))]),

tr([th('Страна команды соперника'), td(input([name(opponentCountry), value(OpponentCountry)]))]),

tr([th('Результат соревнования'), td(input([name(result), value(Result)]))]),

tr(td([colspan(2), align(right)], input([type=submit, value='Изменить']))),

tr([th(''), td(input([name(activTeamName), value(TeamName), type(hidden)]))]),

tr([th(''), td(input([name(activTeamCountry), value(TeamCountry), type(hidden)]))]),

tr([th(''), td(input([name(activOpponentName), value(OpponentName), type(hidden)]))]),

tr([th(''), td(input([name(activOpponentCountry), value(OpponentCountry), type(hidden)]))]),

tr([th(''), td(input([name(rank), value(Rank), type(hidden)]))]),

tr([th(''), td(input([name(sport), value(Sport), type(hidden)]))]),

tr([th(''), td(input([name(year), value(Year), type(hidden)]))]),

tr([th(''), td(input([name(country), value(Country), type(hidden)]))]),

tr([th(''), td(input([name(activResult), value(Result), type(hidden)]))])

] )

]

)]

).

% Предикат для удаления старых и добавления новых данных

update\_data(OldData, NewData):-

retract(OldData),

asserta(NewData).

% Предикат для обновления соревнований

update\_competition(Rank, Sport, Year, Country, OldTeams, NewTeams):-

update\_data(competition(Rank, Sport,Year, Country, OldTeams),

competition(Rank, Sport,Year, Country, NewTeams)).

% Предикат для замены команд в списке с учетом изменения одного результата

replace\_teams([], \_, \_, []).

replace\_teams([OldTeam|RestOldTeams], ChangedTeam, UpdateTeam, [ModifiedTeam|NewTeams]) :-

% Если текущая команда совпадает с командой, которую нужно изменить

OldTeam = team(OldTeamName, OldCountry, OldResults),

ChangedTeam = team(ActivTeamName, ActivCountry, [result(ActivOpponentName, ActivOpponentCountry, ActivResult)]),

UpdateTeam = team(UpdateTeamName, UpdateCountry, [result(UpdateOpponentName, UpdateOpponentCountry, UpdateResult)]),

OldTeamName = ActivTeamName, OldCountry = ActivCountry,

modify\_team\_results(OldResults, result(ActivOpponentName, ActivOpponentCountry, ActivResult), result(UpdateOpponentName, UpdateOpponentCountry, UpdateResult), ModifiedResults),

%format(atom(Row), 'Match: ~w vs. ~w ~n', [OldTeam, ChangedTeam]), % Выводим данные каждого матча

%write(Row), % Выводим данные каждого матча в консоль для отладки

ModifiedTeam = team(UpdateTeamName, UpdateCountry, ModifiedResults),

% Заменяем только одну команду с измененным результатом, остальные остаются без изменений

replace\_teams(RestOldTeams, ChangedTeam, UpdateTeam, NewTeams).

replace\_teams([OldTeam|RestOldTeams], ChangedTeam, UpdateTeam, [OldTeam|NewTeams]) :-

%OldTeam = team(OldTeamName, OldCountry, \_),

%ChangedTeam = team(ActivTeamName, ActivCountry, \_),

%OldTeamName \== ActivTeamName, OldCountry \== ActivCountry,

%format(atom(Row), 'Match: ~w vs. ~w ~n', [OldTeamName=ActivTeamName, OldCountry \= ActivCountry]), % Выводим данные каждого матча

%write(Row), % Выводим данные каждого матча в консоль для отладки

% Если текущая команда не совпадает с командой для замены, оставляем ее без изменений

replace\_teams(RestOldTeams, ChangedTeam, UpdateTeam, NewTeams).

% Предикат для модификации результатов команды

modify\_team\_results([], \_, \_, []).

modify\_team\_results([OldResult|RestOldResults], ChangedResult, NewResult, [NewResult|ModifiedResults]) :-

OldResult = ChangedResult,

modify\_team\_results(RestOldResults, ChangedResult, NewResult, ModifiedResults).

modify\_team\_results([OldResult|RestOldResults], ChangedResult, NewResult, [OldResult|ModifiedResults]) :-

OldResult \= ChangedResult,

modify\_team\_results(RestOldResults, ChangedResult, NewResult, ModifiedResults).

% Обновление соревнования

update\_competition(Request):-

http\_parameters(

Request,

[

rank(Rank, []),

sport(Sport, []),

year(Year, []),

country(Country, []),

teamName(TeamName, []),

teamCountry(TeamCountry, []),

opponentName(OpponentName, []),

opponentCountry(OpponentCountry, []),

result(Result, []),

activTeamName(ActivTeamName, []),

activTeamCountry(ActivTeamCountry, []),

activOpponentName(ActivOpponentName, []),

activOpponentCountry(ActivOpponentCountry, []),

activResult(ActivResult, [])

]

),

% Получаем список Активных команд в БД по Rank/Sport/Year/Country, производим замену команд на новый список и обновляем соревнование

competition(Rank, Sport, Year, Country, ActivTeams),

replace\_teams(ActivTeams,

team(ActivTeamName, ActivTeamCountry,[result(ActivOpponentName, ActivOpponentCountry, ActivResult)]),

team(TeamName, TeamCountry, [result(OpponentName, OpponentCountry, Result)]),

NewTeams),

update\_competition(Rank, Sport, Year, Country, ActivTeams, NewTeams),

% Получаем список Новых команд в БД по Rank/Sport/Year/Country, изменяем результаты соревнований и производим аналогичные действия как выше

competition(Rank, Sport, Year, Country, NewTeams),

check\_and\_replace(Result, NewResult),

check\_and\_replace(ActivResult, NewActivResult),

replace\_teams(NewTeams,

team(ActivOpponentName, ActivOpponentCountry,[result(ActivTeamName, ActivTeamCountry, NewActivResult)]),

team(OpponentName, OpponentCountry, [result(TeamName, TeamCountry, NewResult)]),

NewTeamsTwo),

update\_competition(Rank, Sport, Year, Country, NewTeams, NewTeamsTwo),

http\_redirect(moved, '/', \_Request).