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 №8

Студент: Колбанов Д.О.   
Группа: 6301-020302D   
  
Преподаватель: Лобанков А.А.  
  
Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Дата: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2024

**Цель и постановка задания**

В процессе выполнения заданий ознакомиться с получением информации из базы фактов, с созданием http сервера и html страниц на прологе.

**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.
2. Разработать набор предикатов, осуществляющих взаимодействие с БД, при помощи которых можно реализовать все статические запросы, приведенные в варианте задания.
3. Используя HTTP (Create/Read/Update/Delete) пользователю возможность сбросить базу данных к исходному (тестовому) варианту заполнения.
4. Реализовать вывод результатов всех поисковых запросов к базе фактов на html страницы.

Предметная область – страны мира. Каждая страна может быть

описана структурой: название, площадь, географическое положение,

население. Географическое положение может быть описана структурой: часть

света, материк, океаны, моря, горные хребты. Население может быть описано

структурой: численность, список государственных языков, национальный

состав. Национальный состав может быть описан структурой: национальность,

численность, процент от всего населения.

Реализовать следующие типы запросов:

1. Найти страну, которую омывает больше всего морей;

2. Найти все страны, на территории которых находится указанный

горный хребет;

3. Найти все страны, у которых число национальностей превышает

заданную величину;

4. Найти все горные хребты, находящиеся на территории указанной

страны;

5. Найти все страны, у которых численность населения меньше заданной

величины.

**Описание работы программы**

Вывод списка стран реализовано с помощью функции findall. После список выводится в таблицу в html-странице.

Добавление и удаление стран из базы данных реализовано с помощью asserta и retract.

Запросы по варианту реализованы с помощью функции findall.

**Пример работы программы**

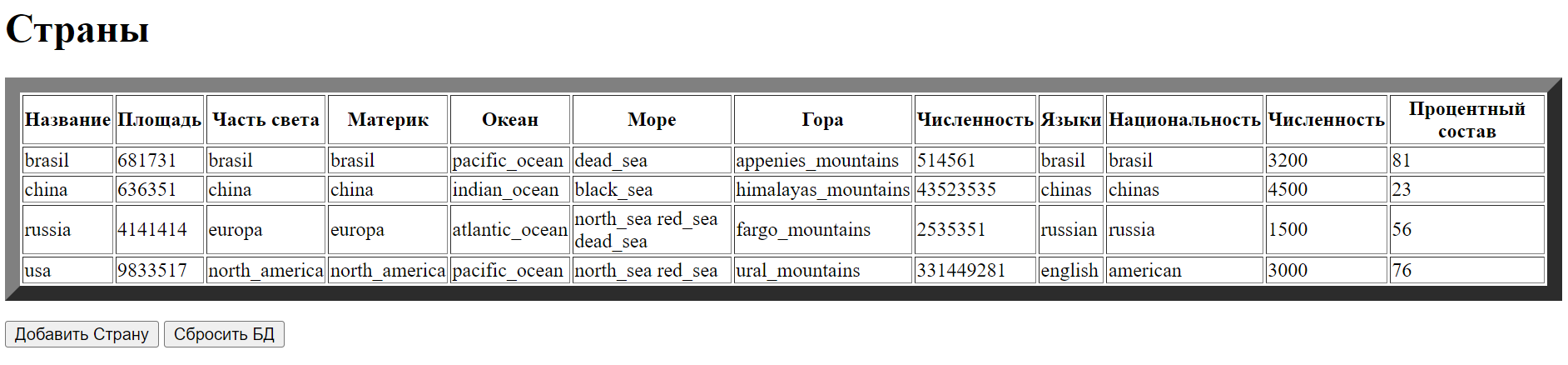
На рисунках 1-16 приведены скриншоты работы лабораторной работы. 

Рисунок 1 – Пример заглавной html-страницы

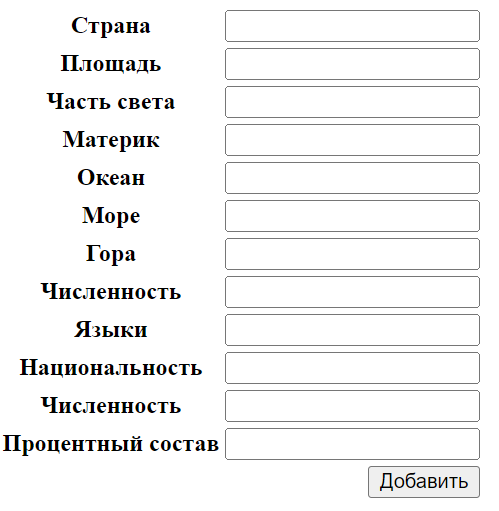


Рисунок 2 – Пример страницы добавления страны

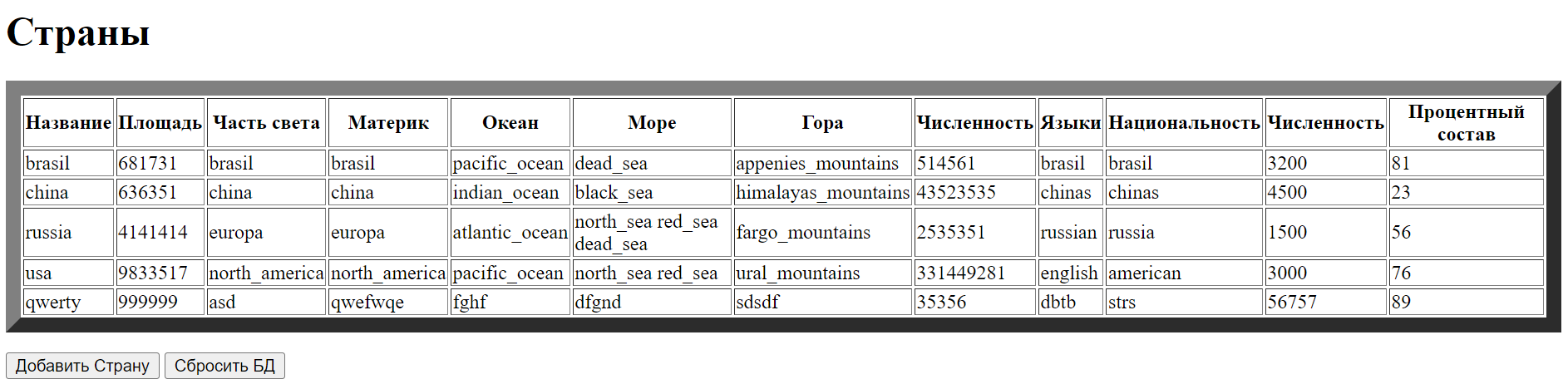


Рисунок 3 – Пример заглавной страницы после добавления страны

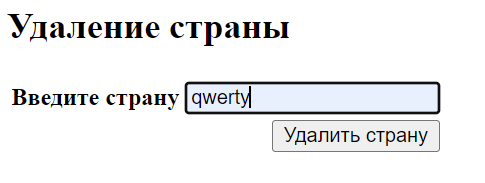


Рисунок 4 – Пример удаления страны

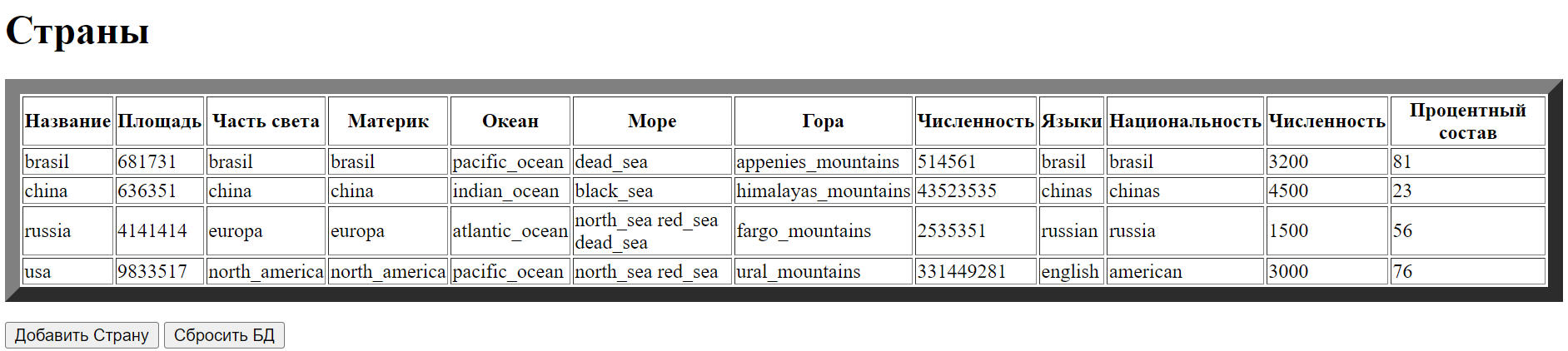


Рисунок 5 – Пример страницы после удаления страны

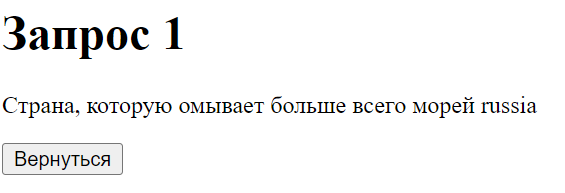


Рисунок 6 – Результат первого запроса

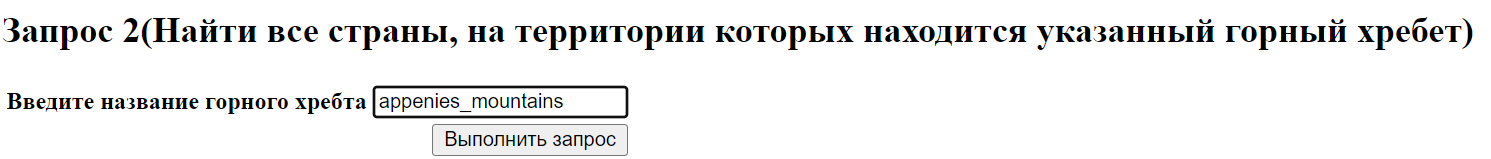


Рисунок 7 – Пример поиска стран, на территории которых находится указанный горный хребет

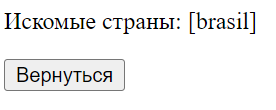


Рисунок 8 – Результат второго запроса

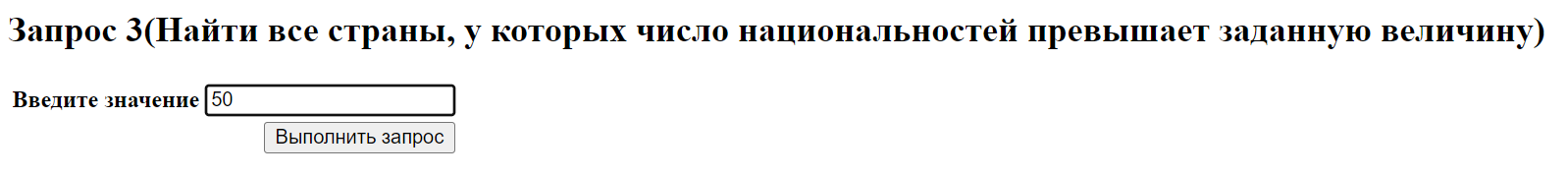


Рисунок 9 – Пример поиска стран, у которых число национальностей превышает заданную величину

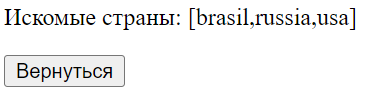


Рисунок 10 – Результат третьего запроса

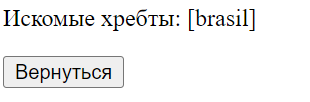


Рисунок 11 – Результат четвёртого запроса

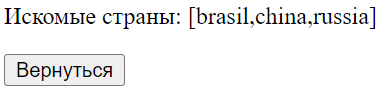


Рисунок 12 – Результат пятого запроса

**Листинг программы**

:-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)).

% Country(Название, Площадь, Геогр.положение(часть света, материк,

% океаны, моря, горный хребты), Население(Численность, список

% гос.языков, национальный состав(национальность, численность,

% процент населения)))

% хэндлер для корневой страницы

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

%хэндлер страницы для добавления новой страны

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

%хэндлер для добавления новой страны

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

%хэндлер для удаления вакансии

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

%хэндлер для ресета БД

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

%хэндлер для страницы обновления соискателя

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

%хэндлер для обновления соискателя

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

%хэндлер для запроса 1(Найти страну, которую омывает больше всего морей)

:- http\_handler(root(find\_country1), find\_country1, []).

% хендлер для запроса 2(Найти все страны, на территории которых

% находится указанный горный хребет)

:- http\_handler(root(find\_count2), find\_count2, []).

% хендлер для запроса 3( Найти все страны, в которых численность

% превышает указаное значение)

:- http\_handler(root(find\_country\_count), find\_country\_count, [method(post)]).

% хендлер для запроса 4(Найти все горные хребты, находящиеся на территории указанной

% страны)

:- http\_handler(root(find\_countries\_with\_mountains), find\_countries\_with\_mountains, [method(post)]).

% хендлер для запроса 5(Найти все страны, у которых численность населения меньше заданной

% величины

:- http\_handler(root(find\_country5), find\_country5, [method(post)]).

% Country(Название, Площадь, Геогр.положение(часть света, материк,

% океаны, моря, горный хребты), Население(Численность, список

% гос.языков, национальный состав(национальность, численность,

% процент населения)))

:- dynamic country/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).

%инициализация базы фактов

reset\_DB(\_):-

retractall(country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_)))),

%инициализация БД

asserta(country(usa, 9833517, geographical\_location(north\_america, north\_america, pacific\_ocean, ['north\_sea', ' red\_sea'] ,ural\_mountains), population(331449281, english, nationality(american, 3000,76)))),

asserta(country(russia, 4141414, geographical\_location(europa, europa, atlantic\_ocean, ['north\_sea', ' red\_sea', ' dead\_sea'], fargo\_mountains), population(2535351, russian, nationality(russia, 1500,56)))),

asserta(country(china, 636351, geographical\_location(china, china, indian\_ocean, [black\_sea] , himalayas\_mountains), population(43523535, chinas, nationality(chinas, 4500,23)))),

asserta(country(brasil, 681731, geographical\_location(brasil, brasil, pacific\_ocean, [dead\_sea] , appenies\_mountains), population(514561, brasil, nationality(brasil, 3200,81)))),

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

%Обработка главной страницы

home\_page(\_Request):-

findall(Name, country(Name,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_))), Names),

findall(Area, country(\_,Area,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_))), Areas),

findall(Part, country(\_,\_,geographical\_location(Part,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_))), Parts),

findall(Continent, country(\_,\_,geographical\_location(\_,Continent,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_))), Continents),

findall(Ocean, country(\_,\_,geographical\_location(\_,\_,Ocean,\_,\_),population(\_,\_,nationality(\_,\_,\_))), Oceans),

findall(Sea, country(\_,\_,geographical\_location(\_,\_,\_,Sea,\_),population(\_,\_,nationality(\_,\_,\_))), Seas),

findall(Mountain, country(\_,\_,geographical\_location(\_,\_,\_,\_,Mountain),population(\_,\_,nationality(\_,\_,\_))), Mountains),

findall(Number, country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(Number,\_,nationality(\_,\_,\_))), Numbers),

findall(Language, country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,Language,nationality(\_,\_,\_))),Languages),

findall(Nationality, country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(Nationality,\_,\_))), Nationalitys),

findall(Count, country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,Count,\_))), Counts),

findall(Percent, country(\_,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,Percent))), Percents),

%Добавляем заголовки таблицы

generate\_rows(Names, Areas, Parts, Continents, Oceans, Seas, Mountains, Numbers, Languages, Nationalitys, Counts, Percents, Rows),

ins(Rows, tr(

[

th('Название'),

th('Площадь'),

th('Часть света'),

th('Материк'),

th('Океан'),

th('Море'),

th('Гора'),

th('Численность'),

th('Языки'),

th('Национальность'),

th('Численность'),

th('Процентный состав')

]

), Rows\_with\_Headers),

reply\_html\_page(

title('Страны'),

[

h1('Страны'),

table(

[border(12)],

Rows\_with\_Headers

),

form(

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

p(button([type(submit), formaction('/add\_country\_page')], 'Добавить Страну'))

),

form(

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

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

),

form(

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

[

h2('Удаление страны'),

table([

tr([th('Введите страну'), td(input([name(name)]))]),

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

])

]

),

form(

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

[

h2('Обновление страны'),

table([

tr([th('Введите страну'), td(input([name(name)]))]),

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

])

]

),

h2('Запрос 1(Найти страну, которую омывает больше всего морей)

'),

form(

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

p(button([type(submit), formaction('/find\_country1')], 'Выполнить запрос'))

),

form(

[action=location\_by\_id(find\_count2), method(post)],

[

h2('Запрос 2(Найти все страны, на территории которых находится указанный

горный хребет)

'),

table([

tr([th('Введите название горного хребта'), td(input([name(name)]))]),

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

])

]

),

form(

[action=location\_by\_id(find\_country\_count), method(post)],

[

h2('Запрос 3(Найти все страны, у которых число национальностей превышает

заданную величину)

'),

table([

tr([th('Введите значение'), td(input([name(name)]))]),

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

])

]

),

form(

[action=location\_by\_id(find\_countries\_with\_mountains), method(post)],

[

h2('Запрос 4(Найти все горные хребты, находящиеся на территории указанной

страны)

'),

table([

tr([th('Введите название страны'), td(input([name(name)]))]),

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

])

]

),

form(

[action=location\_by\_id(find\_country5), method(post)],

[

h2('Запрос 5(Найти все страны, у которых численность населения меньше заданной

величины)

'),

table([

tr([th('Введите значение'), td(input([name(name)]))]),

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

])

]

)

]

).

%Добавление элемента в начало списка

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

%Генерация строк таблицы

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

generate\_rows([A|A1], [B|B1], [C|C1],[D|D1],[E|E1],[F|F1],[G|G1],[H|H1],[I|I1],[J|J1],[K|K1],[L|L1], [tr([td(A),td(B),td(C),td(D),td(E),td(F),td(G),td(H),td(I),td(J),td(K),td(L)])|Rows]):-

generate\_rows(A1, B1, C1, D1, E1, F1, G1, H1, I1, J1, K1, L1, Rows).

%страница с добавлением новой страны

add\_country\_page(\_Request):-

reply\_html\_page(

title('Добавление страны'),

[form(

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

[

table([

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

tr([th('Площадь'), td(input([name(area)]))]),

tr([th('Часть света'), td(input([name(part)]))]),

tr([th('Материк'), td(input([name(continent)]))]),

tr([th('Океан'), td(input([name(ocean)]))]),

tr([th('Море'), td(input([name(sea)]))]),

tr([th('Гора'), td(input([name(mountain)]))]),

tr([th('Численность'), td(input([name(number)]))]),

tr([th('Языки'), td(input([name(language)]))]),

tr([th('Национальность'), td(input([name(nationality)]))]),

tr([th('Численность'), td(input([name(count)]))]),

tr([th('Процентный состав'), td(input([name(percent)]))]),

tr(td([colspan(12), align(right)], input([type=submit, value='Добавить'])))

] )

]

)]

).

%Добавление новой страны

add\_country(Request):-

http\_parameters(

Request,

[

name(Name, []),

area(Area, []),

part(Part, []),

continent(Continent, []),

ocean(Ocean, []),

sea(Sea, []),

mountain(Mountain, []),

number(Number, []),

language(Language, []),

nationality(Nationality, []),

count(Count, []),

percent(Percent,[])

]

),

assertz(country(Name,Area,geographical\_location(Part,Continent,Ocean,Sea,Mountain),population(Number,Language,nationality(Nationality,Count,Percent)))),

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

%Удаление страны

delete\_country(Request):-

http\_parameters(

Request,

[

name(Name, [])

]

),

retract(country(Name,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_)))),

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

%Страница обновления страны

update\_country(Request):-

http\_parameters(

Request,

[

name(Name, [])

]

),

retract(country(Name,\_,geographical\_location(\_,\_,\_,\_,\_),population(\_,\_,nationality(\_,\_,\_)))),

reply\_html\_page(

title('Обновление страны'),

[form(

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

[

table([

tr([th('Название'), td(input([name(name), value(Name), readonly]))]),

tr([th('Площадь'), td(input([name(area)]))]),

tr([th('Часть света'), td(input([name(part)]))]),

tr([th('Материк'), td(input([name(continent)]))]),

tr([th('Океан'), td(input([name(ocean)]))]),

tr([th('Море'), td(input([name(sea)]))]),

tr([th('Гора'), td(input([name(mountain)]))]),

tr([th('Численность'), td(input([name(number)]))]),

tr([th('Языки'), td(input([name(language)]))]),

tr([th('Национальность'), td(input([name(nationality)]))]),

tr([th('Численность'), td(input([name(count)]))]),

tr([th('Процентный состав'), td(input([name(percent)]))]),

tr(td([colspan(12), align(right)], input([type=submit, value='Обновить'])))

] )

]

)]

).

%Обновление страны

update\_c(Request):-

http\_parameters(

Request,

[

name(Name, []),

area(Area, []),

part(Part, []),

continent(Continent, []),

ocean(Ocean, []),

sea(Sea, []),

mountain(Mountain, []),

number(Number, []),

language(Language, []),

nationality(Nationality, []),

count(Count, []),

percent(Percent,[])

]

),

assertz(country(Name,Area,geographical\_location(Part,Continent,Ocean,Sea,Mountain),population(Number,Language,nationality(Nationality,Count,Percent)))),

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

%Запрос 1. Найти страну, которую омывает больше всего морей

find\_country1(\_Request):-

country(Country, \_, geographical\_location(\_, \_, \_, Seas, \_), population(\_,\_,nationality(\_,\_,\_))), % Получаем информацию о стране

length(Seas, NumSeas), % Вычисляем количество омываемых морей

\+ (country(\_, \_, geographical\_location(\_, \_, \_, OtherSeas, \_), population(\_,\_,nationality(\_,\_,\_))), % Проверяем, что нет страны с большим количеством омываемых морей

length(OtherSeas, OtherNumSeas),

OtherNumSeas > NumSeas),

reply\_html\_page(

title('Запрос 1'),

[

h1('Запрос 1'),

div('Страна, которую омывает больше всего морей ~w' - Country),

form(

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

p(button([type(submit), formaction('/')], 'Вернуться'))

)

]

).

%Запрос 2. Найти все страны, с указанным горным хребтом

find\_count2(Request):-

http\_parameters(

Request,

[

name(Name,[])

]

),

findall(Country, country(Country, \_, geographical\_location(\_, \_, \_, \_, Name), population(\_,\_,nationality(\_,\_,\_))), Countries),

reply\_html\_page(

title('Запрос 2'),

[

style(h1, 'color: red'),

div('Искомые страны: ~w' - [Countries]),

form(

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

p(button([type(submit), formaction('/')], 'Вернуться'))

)

]

).

% Запрос 3. Найти все страны, у которых число национальностей превышает

% заданную величину

%

find\_country\_count(Request) :-

http\_parameters(

Request,

[

name(NameAtom,[])

]

),

atom\_number(NameAtom, Name),

findall(Country,(country(Country, \_, geographical\_location(\_,\_,\_,\_,\_), population(\_, \_, nationality(\_, \_, NumNationalities))),NumNationalities > Name),Countries),

reply\_html\_page(

title('Запрос 3'),

[

style(h1, 'color: red'),

div('Искомые страны: ~w' - [Countries]),

form(

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

p(button([type(submit), formaction('/')], 'Вернуться'))

)

]

).

% Запрос 4. Найти все горные хребты, находящиеся на территории указанной

% страны

find\_countries\_with\_mountains(Request) :-

http\_parameters(

Request,

[

name(Name,[])

]

),

findall(Country, country(Country, \_, geographical\_location(\_, \_, \_, \_, Name), population(\_, \_, nationality(\_, \_, \_))), Countries),

reply\_html\_page(

title('Запрос 4'),

[

style(h1, 'color: red'),

div('Искомые хребты: ~w' - [Countries]),

form(

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

p(button([type(submit), formaction('/')], 'Вернуться'))

)

]

).

% Запрос 5. Найти все страны, у которых численность населения меньше

% заданной величины

find\_country5(Request):-

http\_parameters(

Request,

[

name(NameAtom,[])

]

),

atom\_number(NameAtom, Name),

findall(Country,(country(Country, \_, geographical\_location(\_,\_,\_,\_,\_), population(NumNationalities, \_, nationality(\_, \_, \_))),NumNationalities < Name),Countries),

reply\_html\_page(

title('Запрос 5'),

[

style(h1, 'color: red'),

div('Искомые страны: ~w' - [Countries]),

form(

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

p(button([type(submit), formaction('/')], 'Вернуться'))

)

]

).