Донецкий Национальный Технический Университет

Лабораторная работа № 4

«Механизм обратного логического вывода

в продукционных интеллектуальных системах»

Выполнил:

ст. группы ИПЗ -13

Лысенко А. С.

Проверил:

асс. каф ПМИ

Павловский Е.В.

Покровск 2017

Название: механизм обратного логического вывода в продукционных интеллектуальных системах.

Цель работы: получить навыки в проектировании и программировании интерпретатора продукционных правил, работающего по принципу обратного логического вывода, а также научиться применять эффективные стратегии поиска решения в древовидном пространстве состояний.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Стратегия поиска | Место расположения базы знаний | Система программирования |
| 4 | В глубину | Оперативная память | Си |

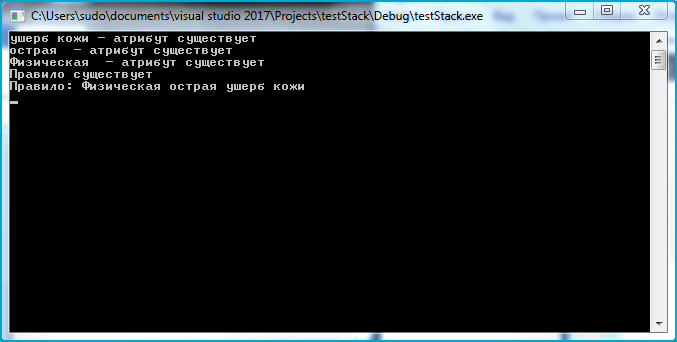
Логический вывод представляет собой механизм общения с пользователем системы в моем случае, данные заданы статически и изменить их можно только из программы.

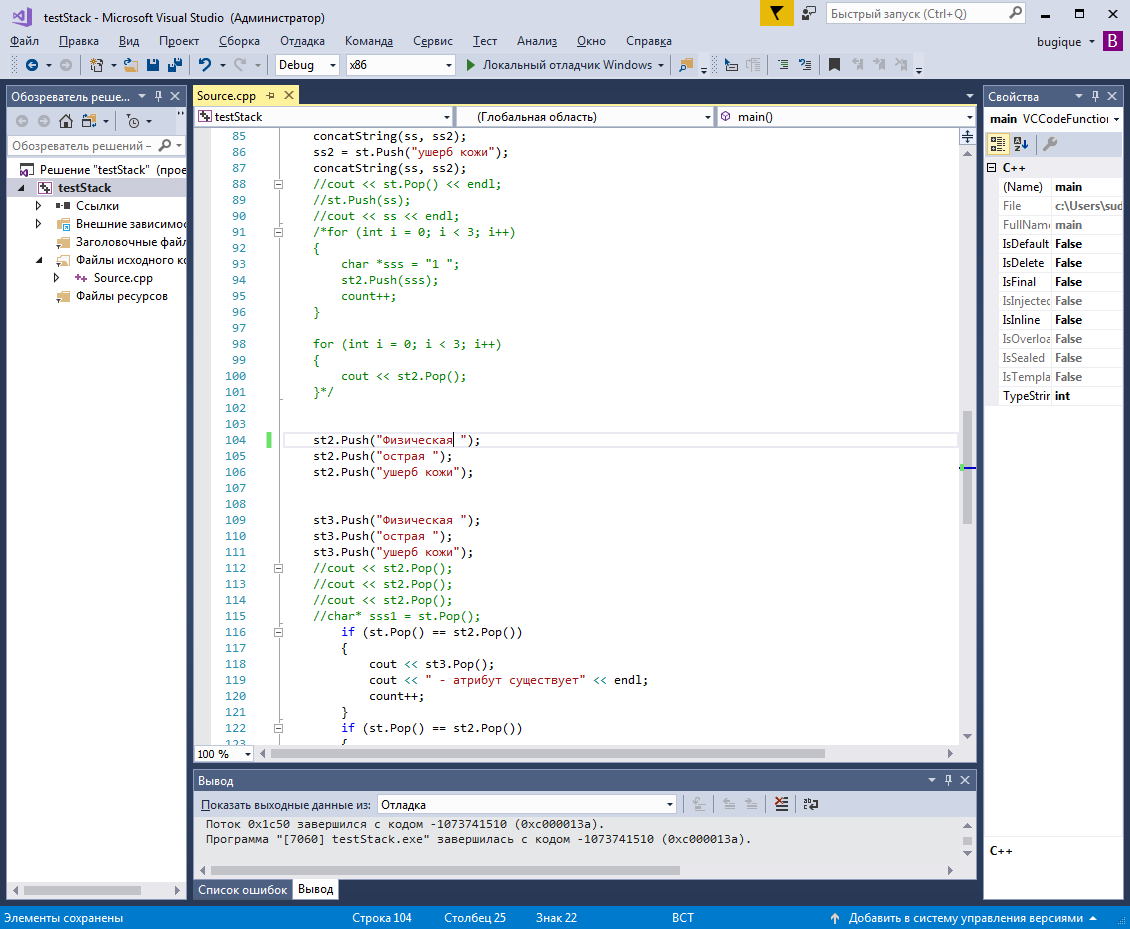
Основные структуры данных в ЭС это стек, в программе использую 3 стека и для формирования атрибутов правила, 2-ой для формирования продукционного правила, 3-й для вывода данных, т.к. данных выводятся с помощью функции Pop() а вводятся с помощью функции Push(), то вызывать их можно только ограниченное число раз, поэтому создаем доп. Стек для вывода.

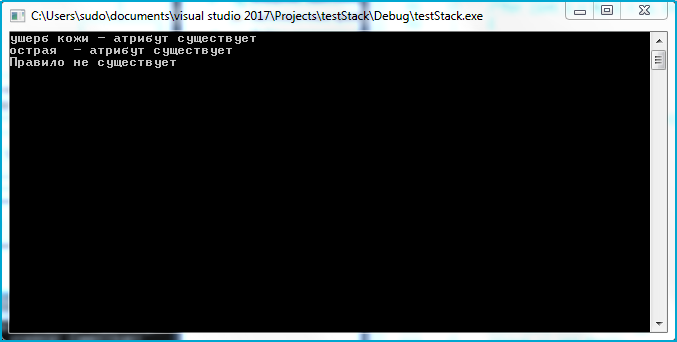
Помещаем правило в стек целей, затем извлекаем и сравниваем со стеком атрибутов если все атрибуты из стека целей соответствуют всем атрибутам из стека контекста, тогда правило существует, иначе нет.

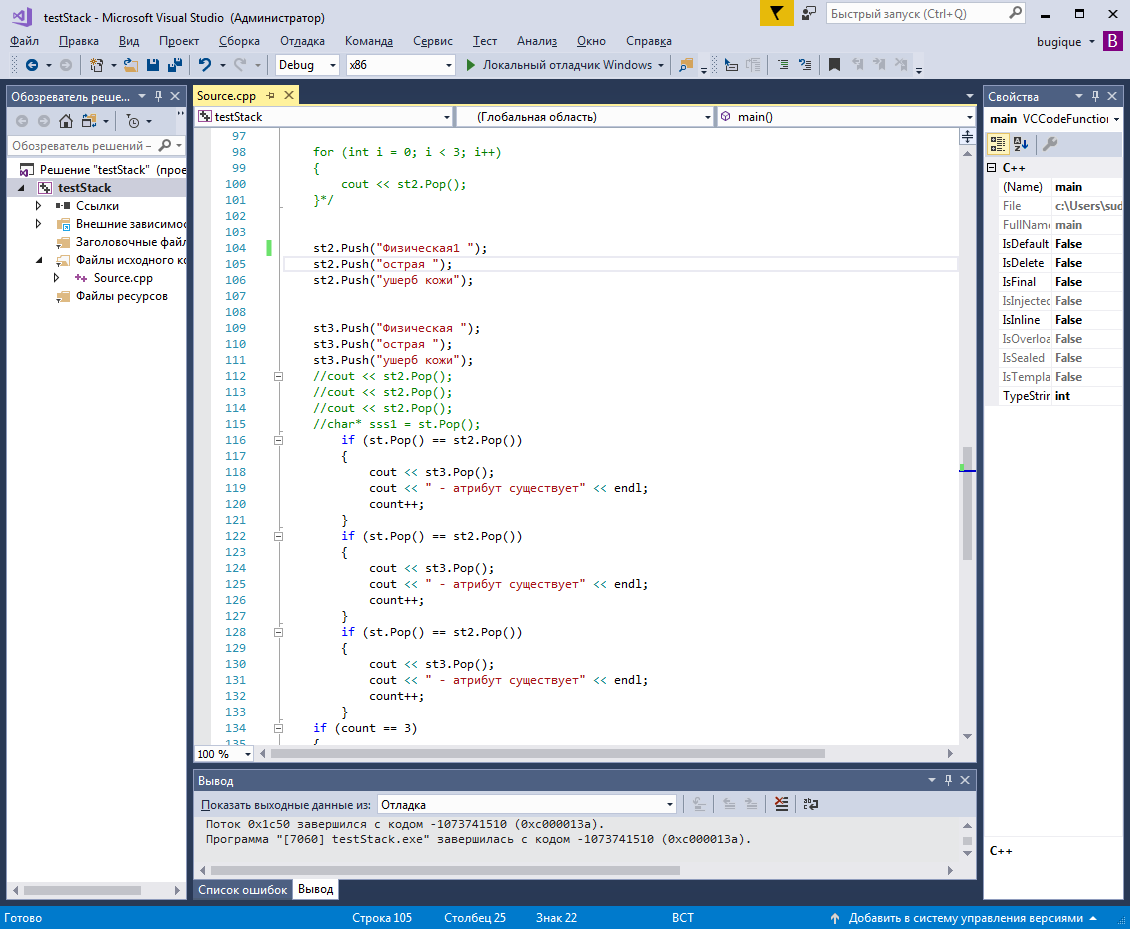
Деревья решения задач











Source.cpp

#include <iostream>

#include <conio.h>

using namespace std;

struct Stek

{

char\* key;

Stek \*next;

};

class Stack2

{

private:

Stek \*next;

public:

Stack2() {};

char\* Push(char\* d)

{

Stek \*pv = new Stek; // объявляем новую динамическую переменную типа Stek

pv->key = d; // записываем значение, которое помещается в стек

pv->next = next; // связываем новый элемент стека с предыдущим

next = pv; // новый элемент стека становиться его вершиной

return d;

}

char\* Pop()

{

char\* temp = next->key; // извлекаем в переменную temp значение в вершине стека

Stek \*pv = next; // запоминаем указатель на вершину стека, чтобы затем

// освободить выделенную под него память

next = next->next; // вершиной становится предшествующий top элемент

delete pv; // освобождаем память, тем самым удалили вершину

return temp; // возврощаем значение, которое было в вершине

}

};

void concatString(char \*str, const char \*str2)

{

int i = 0, j = 0;

while (1)

{

if (str[i] != '\0')

{

i++;

continue;

}

if (str2[j] != '\0')

{

str[i] = str2[j];

i++;

j++;

}

else

{

str[i] = '\0';

return;

}

}

}

int main()

{

setlocale(LC\_ALL, "Rus");

Stack2 st;

Stack2 st2;

Stack2 st3;

int count = 0;

char ss[100] = {};

char\* ss2;

ss2 = st.Push("Физическая ");

concatString(ss, ss2);

//cout << st.Pop() << endl;

ss2 = st.Push("острая ");

//cout << st.Pop() << endl;

concatString(ss, ss2);

ss2 = st.Push("ушерб кожи");

concatString(ss, ss2);

//cout << st.Pop() << endl;

//st.Push(ss);

//cout << ss << endl;

/\*for (int i = 0; i < 3; i++)

{

char \*sss = "1 ";

st2.Push(sss);

count++;

}

for (int i = 0; i < 3; i++)

{

cout << st2.Pop();

}\*/

st2.Push("Физическая ");

st2.Push("острая ");

st2.Push("ушерб кожи");

st3.Push("Физическая ");

st3.Push("острая ");

st3.Push("ушерб кожи");

//cout << st2.Pop();

//cout << st2.Pop();

//cout << st2.Pop();

//char\* sss1 = st.Pop();

if (st.Pop() == st2.Pop())

{

cout << st3.Pop();

cout << " - aтрибут существует" << endl;

count++;

}

if (st.Pop() == st2.Pop())

{

cout << st3.Pop();

cout << " - aтрибут существует" << endl;

count++;

}

if (st.Pop() == st2.Pop())

{

cout << st3.Pop();

cout << " - aтрибут существует" << endl;

count++;

}

if (count == 3)

{

cout << "Правило существует" << endl;

cout << "Правило: " << ss << endl;

}

else

cout << "Правило не существует" << endl;

/\*if (st.Pop() == st2.Pop())

cout << "ok";

if (st.Pop() == st2.Pop())

cout << "ok";\*/

\_getch();

return 0;

}