ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ НАЦИОНАЛЬНЫЙ**

**ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

(НИУ «БелГУ»)

ИНСТИТУТ ИНЖЕНЕРНЫХ И ЦИФРОВЫХ ТЕХНОЛОГИЙ

КАФЕДРА МАТЕМАТИЧЕСКОГО И ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ИНФОРМАЦИОННЫХ СИСТЕМ

**Отчет по дополнительному заданию**

**по дисциплине: «Интеллектуальные системы»**

студента очного отделения

4 курса 12001801 группы

Капустина Виктора Сергеевича

Проверил(а):

Константинов Игорь Сергеевич

Белгород 2022

**Цель работы:** Разработка программы, которая будет выводить формулы по заданной таблице истинности

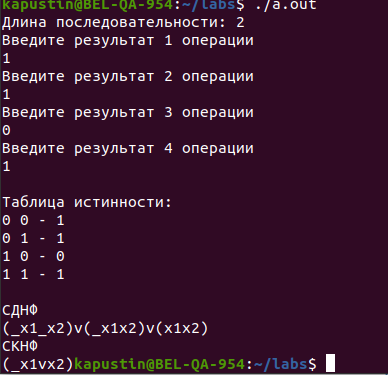
Для разработки программы обозначим требования к её выполнению. Для начала программы необходимо задать саму таблицу истинности для выбранной операции и после этого программы составит формулу по заполненным данным. В качестве вывода программы выдаст формулы по следующему шаблону: (\_x1x2)v(x3\_x4) - Дизъюнкция 2 конъюнктов, где “\_x1” и “\_x4” являются **⌝**X1 и **⌝**X4; “x2” и “x3” является X2 и X3; v - обозначение дизъюнкции, а нахождение двух переменных вместе подразумевает под собой конъюнкцию.

Для первого примера рассмотрим импликацию, как формулу с двумя переменными.

Таблица 1. Таблица истинности для импликации

| A | B | F ( A→B ) |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

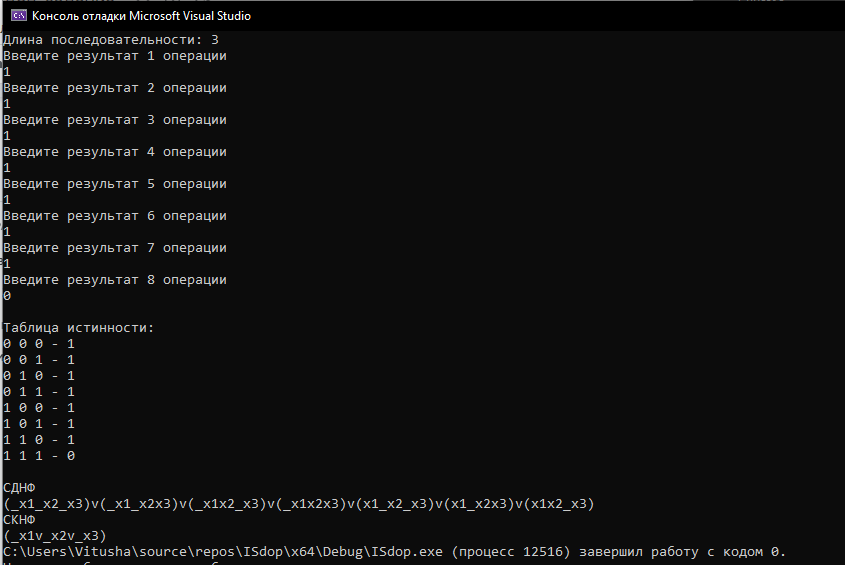
**Тестирование программы**

  
Рис. 1 Тестирование программы для импликации

Для достижения большей универсальности, попробуем промасштабировать таблицу истинности и проверить работоспособность программы на больших по масштабу таблицах истинности. В качестве формулы возьмем ⌝(A&B&C)

Таблица 2. Таблица истинности для трёх переменных

| A | B | C | F (⌝(A&B&C) ) |
| --- | --- | --- | --- |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 |

  
Рис. 2 Тестирование программы для формулы с 3 переменными

**Приложение**

**Листинг программы 1**

#include <iostream>

#include <experimental/vector>

#include <vector>

#include <cmath>

using namespace std;

int main()

{

////// Создание таблицы истинности

int n;

cout << "Длина последовательности: ";

cin >> n;

int nn=pow(2,n);

int Main\_array[nn][n];

int Sub\_Array[nn];

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

{

cout << "Введите результат " << i+1 << " операции"<< endl;

cin >> Sub\_Array[i];

}

int k=1,i,j,p,p1=n,N=2,p3=1,t;

for(i=1;i<=n;i++)

p3=N\*p3;

for(i=0;i<nn;i++)

for(j=0;j<n;j++)

Main\_array[i][j]=0;

for(j=p1-1;j>=0;j--)

{

for(i=0+k;i<p3;i++)

{

for(t=0;t<k;t++)

{

Main\_array[i][j]=1;

i++;

}

}

k\*=2;

}

////////////////// Вывод таблицы истинности

vector<int> v1;

vector<int> v2;

cout << endl << "Таблица истиннocти:" << endl;

for(i=0;i<nn;i++)

{

for(j=0;j<n;j++){

cout<<Main\_array[i][j]<<" ";

}

cout << "- "<<Sub\_Array[i];

cout<<endl;

}

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

{

if (Sub\_Array[i]==1)

{//////СДНФ

v1.push\_back(0);

for(int j=0; j<n; j++)

{

if (Main\_array[i][j]==0)

{

v1.push\_back((-1)\*(j+1));

}

else v1.push\_back(j+1);

}

}

else

{//////СКНФ

v2.push\_back(0);

for(int j=0; j<n; j++)

{

if (Main\_array[i][j]==1)

{

v2.push\_back((-1)\*(j+1));

}

else v2.push\_back(j+1);

if(j!=n-1)

{

v2.push\_back(0);

}

}

}

}

cout << endl << "СДНФ " << endl;

cout <<"(";

for (vector<int>::iterator i=v1.begin()+1; i<v1.end(); ++i)

{

if((\*i)<0)

{

cout <<"\_x"<<abs(\*i);

}

else if(\*i>0)

{

cout<<"x"<<abs(\*i);

}

else cout<<")v(";

}

cout <<")";

cout << endl << "СКНФ" << endl;

cout <<"(";

for (vector<int>::iterator i=v2.begin()+1; i<v2.end(); ++i)

{

if((\*i)<0)

{

cout <<"\_x"<<abs(\*i);

}

else if(\*i>0)

{

cout<<"x"<<abs(\*i);

}

else cout<<"v";

}

cout <<")";

return 0;

}

**Конец листинга 1**