**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа № 3

дисциплина: Математическая логика и теория алгоритмов

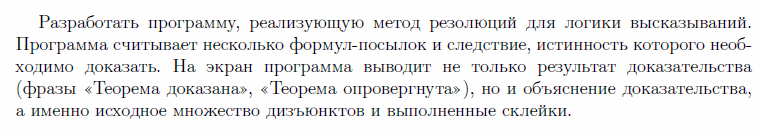
тема: «Формальные теории»

|  |  |
| --- | --- |
|  | Выполнил: ст. группы ВТ-22  Воскобойников Илья Сергеевич  Проверил: Бондаренко Т.В. |

Белгород 2019

**Практическая часть:**

**Вариант 1**



#include **<stdio.h>**#include **<stdlib.h>**#include **<math.h>**#include **"malloc.h"***//выделение памяти массиву***int** \*getMemoryArr(**int** size) {  
 **return** (**int** \*)malloc(size \* **sizeof**(**int**));  
}  
  
*//очищенине памяти массива***void** freeMemoryArr(**int** \*arr) {  
 free(arr);  
}  
  
*//выделение памяти матрице***int** \*\*getMemoryMatr(**int** str, **int** tab) {  
 **int** \*\*matr = (**int** \*\*)malloc(str \* **sizeof**(**int** \*));  
 **for** (**int** i = 0; i < str; i++)  
 matr[i] = (**int** \*)malloc(tab \* **sizeof**(**int**));  
 **return** matr;  
}  
*//ввод КНФ пользователем в заданном программой виде***void** input\_knf(**int**\*\* a, **int**\* b, **int** m, **int** n) {  
 **int** i, j;  
 printf(**"Ваши элементы: "**);  
 **for** (i = 0; i < n; i++) {  
 b[i] = **'A'** + i;  
 printf(**"%c "**, b[i]);  
 }  
 printf(**"Ввод КНф:\n"**);  
 printf(**"Элемент есть: 1"**);  
 printf(**"\nОтрицание элемента : -1"**);  
 printf(**"\nЭлемента нет : 0 \n"**);  
 **for** (i = 0; i < m; i++)  
 **for** (j = 0; j < n; j++)  
 scanf(**"%i"**, &a[i][j]);  
}  
*//вывод КНФ на экран***void** output\_form(**int** \*\*a, **int**\* b, **int** m, **int** n) {  
 **int** i = 0, j = 0;  
 **for** (i = 0; i < m; i++) {  
 printf(**"("**);  
 j = 0;  
 **while** ((a[i][j] == 0) && (j < n))  
 j++;  
 **if** (j < n) {  
 **if** (a[i][j] == -1)  
 printf(**"!%c"**, b[j]);  
 **else** printf(**"%c"**, b[j]);  
 }  
 j++;  
 **for** (j; j < n; j++) {  
 **if** (a[i][j] == -1)  
 printf(**" v !%c"**, b[j]);  
 **if** (a[i][j] == 1)  
 printf(**" v %c"**, b[j]);  
 }  
 printf(**")"**);  
 **if** (i + 1 < m) printf(**"^"**);  
 }  
 printf(**"\n"**);  
}  
*//вывод полученной таблицы истинности а для данной КНФ***int**\*\* output\_table(**int** \*\*a, **int** \*b, **int** m, **int** n) {  
 **int** i, j, mask = 1;  
 **for** (i = 0; i < n; i++)  
 printf(**"%c \t"**, b[i]);  
 printf(**"f\t\n"**);  
 **int** all = pow(2, n);  
  
 **int** \*\*mas = getMemoryMatr(all, n + 1);  
  
 **for** (i = 0; i < all; i++) {  
 **for** (j = 0; j < n; j++) {  
 mas[i][j] = (mask & (i >> (n - 1 - j)));  
 printf(**"%i\t"**, mas[i][j]);  
 }  
 **int** f = 1, x;  
 **int** j1 = 0, k;  
 **while** (j1 < m) {  
 x = 0;  
 k = 0;  
 **while** (k < n) {  
 **if** (a[j1][k] == 1)  
 x |= mas[i][k];  
 **if** (a[j1][k] == -1)  
 x |= !mas[i][k];  
 k++;  
 }  
 f = f && x;  
 j1++;  
 }  
 mas[i][n] = f;  
 printf(**"%i\n"**, mas[i][n]);  
 }  
 **return** mas;  
}  
  
**int** next(**int** \*a, **int** n, **int** ind) {  
 **for** (**int** i = ind; i < n; i++)  
 **if** (a[i])  
 **return** 1;  
 **return** 0;  
}  
  
**int** scobka(**int**\* a, **int** n) {  
 **int** i, f = 1;  
 **for** (i = 0; i < n; i++) {  
 **if** ((a[i] == 1)) {  
 printf(**"%c"**, **'A'** + i);  
 **if** (next(a, n, i + 1)) printf(**" v "**);  
 f = 0;  
 }  
 **if** ((+a[i] == -1)) {  
 printf(**"!%c"**, **'A'** + i);  
 **if** (next(a, n, i + 1)) printf(**" v "**);  
 f = 0;  
 }  
 }  
 **if** (f) {  
 printf(**"%c"**, **'0'**);  
 **return** 0;  
 }  
 **return** 1;  
}  
  
**int** one\_dif(**int** \*a, **int** \*b, **int** n) {  
 **int** flag = 0, i;  
 **for** (i = 0; i < n; i++)  
 **if** (a[i] == -b[i] && a[i] != 0) flag++;  
  
 **return** (flag == 1) ? 1 : 0;  
}  
  
**void** skleika(**int** \*a, **int** \*b, **int** \*rez, **int** n) {  
 **for** (**int** i = 0; i < n; i++) {  
 **if** (a[i] \* b[i] == -1 || (!a[i] && !b[i]))  
 rez[i] = 0;  
 **else if** ((a[i] \* b[i] == 1 && a[i] == 1) || (a[i] \* b[i] == 0 && ((a[i] == 0 && b[i] == 1) || (a[i] == 1 && b[i] == 0))))  
 rez[i] = 1;  
 **else** rez[i] = -1;  
 }  
}  
  
**void** rez(**int** \*\*a, **int** m, **int** n) {  
 **int**\* rez1 = getMemoryArr(n);  
  
 **int** i, j, f = 1;  
 **for** (j = 0; j < n; j++)  
 rez1[j] = a[0][j];  
  
 **for** (i = 0; i < m && f; i++) {  
 **if** (a[i][0] != 5) {  
 **for** (j = 0; j < m && f; j++)  
 **if** (one\_dif(rez1, a[j], n) && a[j][0] != 5) {  
 printf(**"("**);  
 scobka(rez1, n);  
 printf(**")"**);  
 printf(**" ^ "**);  
 printf(**"("**);  
 scobka(a[j], n);  
 printf(**")"**);  
 printf(**" => "**);  
 skleika(rez1, a[j], rez1, n);  
 f = scobka(rez1, n);  
 printf(**"\n"**);  
 }  
 }  
 }  
}  
  
  
  
*//очищенине памяти матрицы***void** freeMemoryMatr(**int** \*\*matr, **int** str) {  
 **for** (**int** i = 0; i < str; i++)  
 free(matr[i]);  
 free(matr);  
}  
  
  
 **int** main() {  
  
  
 **int** i = 0;  
 printf(**"Введите количество скобок: "**);  
 **int** m;  
 scanf(**"%i"**, &m);  
 printf(**"Введите количество переменных: "**);  
 **int** n;  
 scanf(**"%i"**, &n);  
 **int** all = pow(2, n);  
  
 **int** \*\*mas = getMemoryMatr(all, n + 1);  
 **int** \*\*a = getMemoryMatr(m, n);  
 **int** \*name = getMemoryArr(n);  
  
 input\_knf(a, name, m, n);  
 output\_form(a, name, m, n);  
 mas = output\_table(a, name, m, n);  
 printf(**"\n"**);  
 rez(a, m, n);  
  
 freeMemoryArr(name);  
 freeMemoryMatr(mas, all);  
 freeMemoryMatr(a, m);  
 **return** 0;  
}

