Приложение 2

**program** z\_8;

**uses** or\_arr;

**var** a, b, c, x, d, e, f, g: t\_arr;

n, m, k, j, y, z, w, h: byte;

**begin**

writeln ('Введите количество элементов множества A');

read (n);

writeln ('Введите элементы множества А');

input (a, n);

writeln ('Введите количество элементов множества B');

read (m);

writeln ('Введите элементы множества B');

input (b, m);

writeln ('Введите количество элементов множества C');

read (k);

writeln ('Введите элементы множества C');

input (c, k);

j := difference(b, c, m, k, x);

y := intersection(a, x, n, j, d);

j := complement(d, y, x);

y := difference(c, b, k, m, d);

z := complement(d, y, e);

y := intersection (x, e, j, z, d);

j := complement(d, y, x);

y := difference(b, c, m, k, d);

h := complement(d, y, g);

y := complement(a, n, d);

z := unification(d, g, y, h, e);

y := complement(e, z, d);

w := difference(c, b, k, m, f);

z := unification(d, f, y, w, e);

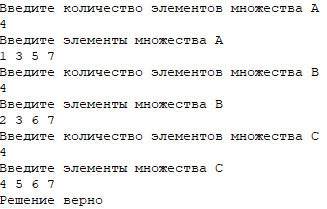
**if** equality(x, e, j, z) **then**

writeln('Решение верно')

**else**

writeln('Решение не верно');

**end**.



Приложение 1

#include <stdio.h>

#include <math.h>

#include <locale.h>

#define HOWSETS 3

#define HOWVEC (int)pow(2, HOWSETS)

void form\_code (unsigned int a[][HOWSETS])

{

for (unsigned int i = 0; i < HOWVEC; i++){

unsigned int mask = 1 << HOWSETS-1;

for (unsigned int j = 0; j < HOWSETS; j++){

a[i][j] = (i&mask)!=0?1:0;

mask >>= 1;

}

}

}

void write\_multi\_arr (unsigned int a[][HOWSETS])

{

for (int i = 0; i < HOWVEC; i++){

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

printf ("%d ", a[i][j]);

printf ("\n");

}

printf ("\n");

}

void write\_arr (int a[])

{

for (int i = 0; i < HOWVEC; i++){

printf ("%d ", a[i]);

}

printf ("\n");

}

int main()

{

setlocale (LC\_ALL, "rus");

unsigned int binary\_vec[HOWVEC][HOWSETS];

form\_code (binary\_vec);

printf ("Значение всех двоичных векторов для заданного кол-ва исходных множеств:\n");

write\_multi\_arr (binary\_vec);

unsigned int truth\_table\_first[HOWVEC];

for (int i = 0; i < HOWVEC; i++){

int f\_op = binary\_vec[i][0] & binary\_vec[i][1];

int s\_op = binary\_vec[i][0] & !binary\_vec[i][2];

truth\_table\_first[i] = f\_op | s\_op | binary\_vec[i][2] | !binary\_vec[i][1];

}

printf ("Таблица истинности первого выражения:\n");

write\_arr (truth\_table\_first);

unsigned int truth\_table\_second [HOWVEC];

for (int i = 0; i < HOWVEC; i++){

int f\_op = binary\_vec[i][0] & binary\_vec[i][1];

int s\_op = binary\_vec[i][0] & !binary\_vec[i][2];

truth\_table\_second[i] = f\_op | s\_op | binary\_vec[i][2] | !binary\_vec[i][1];

}

printf ("Таблица истинности второго выражения:\n");

write\_arr (truth\_table\_second);

}

