**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ

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 №1.4

дисциплина: Дискретная математика

тема: «Теоретико-множественные уравнения»

Выполнил: ст. группы ПВ-201

Машуров Дмитрий Русланович

Проверил: Бондаренко Т.В.

Белгород 2021

**Лабораторная работа №1.4**

**Теоретико-множественные уравнения**

**Цель работы:** научиться решать теоретико-множественные уравнения с применением ЭВМ.

**Задания**

1. Преобразовать исходное уравнение (см. ―Варианты заданий‖) в уравнение с пустой правой частью.
2. Преобразовать левую часть уравнения к виду , используя разложение Шеннона по неизвестному множеству X.
3. Написать программу, вычисляющую значения множеств ϕ ∅ и U ϕ при заданных исходных множествах.
4. Вычислить значения множеств ϕ ∅ и U ϕ и сделать вывод о существовании решения уравнения. Если решения уравнения не существует, то выполнить п.п. 1—4 для следующего (предыдущего) варианта.
5. Определить мощность общего решения, найти некоторые (или все) частные решения, в том числе частные решения наименьшей и наибольшей мощности. 6. Написать программу для проверки найденных решений

**Задание варианта №17**

**Выполнение:**

1. Преобразую уравнение своего варианта к уравнению с пустой правой частью:
2. Выполню разложение Шеннона левой части уравнения по неизвестному множеству :

Преобразую их

1. Программа

#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
  
#define MIN(a, b) (a < b ? a : b)  
#define MAX(a, b) (a > b ? a : b)  
  
struct set {  
 int \*values;  
 int min;  
 int max;  
};  
  
int is\_elem\_in(struct set \*s, int elem) {  
 int \*a = s->values;  
  
 if (a[elem] == 1) {  
 return 1;  
 }  
  
 return 0;  
}  
  
struct set create\_set(int min, int max) {  
 int n = max - min,  
 \*a = (int \*) calloc(sizeof(int), n);  
 struct set temp;  
  
 temp.max = max;  
 temp.min = min;  
  
 temp.values = a;  
  
 return temp;  
}  
  
void delete\_set(struct set \*s) {  
 free(s->values);  
 free(s);  
}  
  
struct set complemept\_set(struct set \*s) {  
 int \*a = s->values;  
 struct set temp = create\_set(s->min, s->max);  
 int \*b = temp.values;  
  
 for (int i = s->min; i <= s->max; i++) {  
 b[i] = (a[i] == 0 ? 1 : 0);  
 }  
  
 return temp;  
}  
  
void output\_set(struct set \*s) {  
 int \*values = s->values;  
  
 printf("{ ");  
  
 for (int i = s->min; i <= s->max - 1; i++) {  
 if (values[i] == 1)  
 printf("%d ", i);  
 }  
 if (values[s->max] == 1)  
 printf("%d}\n", s->max);  
 else  
 printf("}");  
}  
  
void add(struct set \*s, int value) {  
 int \*values = s->values;  
 values[value] = 1;  
}  
  
void delete(struct set \*s, int value) {  
 int \*values = s->values;  
  
 if (values[value] == 1) {  
 values[value] = 0;  
 }  
}  
  
struct set union\_set(struct set \*s1, struct set \*s2) {  
 int new\_min = MIN(s1->min, s2->min),  
 new\_max = MAX(s1->max, s2->max),  
 \*a = s1->values,  
 \*b = s2->values;  
 struct set temp = create\_set(new\_min, new\_max);  
 int \*t = temp.values;  
  
 for (int i = s1->min; i < s1->max; i++) {  
 if (t[i] == 0) t[i] = a[i];  
 }  
  
 for (int i = s2->min; i < s2->max; i++) {  
 if (t[i] == 0) t[i] = b[i];  
 }  
  
 temp.max = new\_max;  
 temp.min = new\_min;  
 return temp;  
}  
  
struct set intersection\_set(struct set \*s1, struct set \*s2) {  
 int new\_min = MAX(s1->min, s2->min),  
 new\_max = MIN(s1->max, s2->max),  
 \*a = s1->values, \*b = s2->values;  
 struct set temp = create\_set(new\_min, new\_max);  
 int \*t = temp.values;  
  
 for (int i = new\_min; i < new\_max; i++) {  
 if ((a[i] == 1) && (b[i] == 1))  
 t[i] = 1;  
 }  
  
 temp.max = new\_max;  
 temp.min = new\_min;  
 return temp;  
}  
  
int is\_subset(struct set \*s1, struct set \*s2) {  
 int \*a = s2->values;  
  
 for (int i = s2->min; i < s2->max; i++) {  
 if (a[i] == 1) {  
 if ((!is\_elem\_in(s1, i))) {  
 return 0;  
 }  
 }  
 }  
  
 return 1;  
}  
  
struct set subtraction\_set(struct set \*s1, struct set \*s2) {  
 int \*a = s1->values,  
 \*b = s2->values;  
 struct set temp = create\_set(s1->min, s1->max);  
 int \*t = temp.values;  
  
 for (int i = s1->min; i < s2->max; i++) {  
 if (s2->min <= i && i <= s2->max) {  
 if (a[i] == 1 && b[i] == 1) {  
 t[i] = 0;  
 } else t[i] = a[i];  
 } else t[i] = a[i];  
 }  
  
 temp.min = s1->min;  
 temp.max = s1->max;  
 return temp;  
}  
  
struct set sym\_subtraction(struct set \*s1, struct set \*s2) {  
 struct set t1, t2, temp;  
  
 t1 = subtraction\_set(s1, s2);  
 t2 = subtraction\_set(s2, s1);  
 temp = union\_set(&t1, &t2);  
  
 delete\_set(&t1);  
 delete\_set(&t2);  
  
 return temp;  
}  
  
struct set get\_set\_from\_array(int min, int max, const int \*a, int n) {  
 struct set temp = create\_set(min, max);  
 int \*t = temp.values;  
  
 for (int i = 0; i < n; ++i) {  
 t[a[i]] = 1;  
 }  
  
 temp.max = max;  
 temp.min = min;  
 return temp;  
}  
  
void print\_set\_of\_solves(struct set \*s, struct set \*emp) {  
 int \*a = s->values, r;  
  
 for (int i = 0; i < 10; ++i) {  
 if (a[i] == 0) {  
 continue;  
 } else  
 r = i;  
  
 if (is\_subset(s, emp) && is\_subset(emp, s)) {  
 output\_set(emp);  
 } else {  
 add(emp, r);  
 delete(s, r);  
 print\_set\_of\_solves(s, emp);  
 }  
  
  
 }  
}  
  
int main() {  
 struct set t1, t2, t3, t4, t5, t6, t7, te1, res\_emp, res\_uni,  
 u = create\_set(1, 10),  
 e = create\_set(1, 10),  
 a = create\_set(1, 10),  
 b = create\_set(1, 10),  
 c = create\_set(1, 10);  
  
 //init U - universum  
 for (int i = 1; i <= 11; ++i) {  
 add(&u, i);  
 }  
  
 //init A  
 add(&a, 3);  
 add(&a, 7);  
 add(&a, 9);  
 add(&a, 10);  
  
 //init B  
 add(&b, 1);  
 add(&b, 3);  
 add(&b, 8);  
 add(&b, 9);  
 add(&b, 10);  
  
 //init C  
 add(&c, 2);  
 add(&c, 4);  
 add(&c, 5);  
 add(&c, 6);  
 add(&c, 7);  
  
 print\_set\_of\_solves(&a, &b);  
  
 //expression 1  
 te1 = complemept\_set(&e);  
  
 t1 = subtraction\_set(&c, &e);  
  
 t2 = subtraction\_set(&a, &te1);  
 t3 = complemept\_set(&t2);  
  
 t4 = subtraction\_set(&a, &e);  
  
 t5 = intersection\_set(&b, &e);  
  
 t6 = sym\_subtraction(&t1, &t3);  
 t7 = sym\_subtraction(&t4, &t5);  
 res\_emp = sym\_subtraction(&t6, &t7);  
  
 printf("fi^empty = ");  
 output\_set(&res\_emp);  
  
 delete\_set(&te1);  
 delete\_set(&t1);  
 delete\_set(&t2);  
 delete\_set(&t3);  
 delete\_set(&t4);  
 delete\_set(&t5);  
 delete\_set(&t6);  
 delete\_set(&t7);  
  
 printf("\n");  
  
 //expression 2  
 te1 = complemept\_set(&u);  
  
 t1 = subtraction\_set(&c, &u);  
  
 t2 = subtraction\_set(&a, &te1);  
 t3 = complemept\_set(&t2);  
  
 t4 = subtraction\_set(&a, &u);  
  
 t5 = intersection\_set(&b, &u);  
  
 t6 = sym\_subtraction(&t1, &t3);  
 t7 = sym\_subtraction(&t4, &t5);  
 res\_uni = sym\_subtraction(&t6, &t7);  
 res\_uni = complemept\_set(&res\_uni);  
  
 printf("fi^u = ");  
 output\_set(&res\_uni);  
  
 delete\_set(&te1);  
 delete\_set(&t1);  
 delete\_set(&t2);  
 delete\_set(&t3);  
 delete\_set(&t4);  
 delete\_set(&t5);  
 delete\_set(&t6);  
 delete\_set(&t7);  
  
 printf("\n");  
  
 //equals  
  
 if (is\_subset(&res\_uni, &res\_emp)) {  
 t1 = subtraction\_set(&res\_uni, &res\_emp);  
 printf("fi^empty is subset of fi^u\n");  
  
 if (is\_subset(&e, &t1) && is\_subset(&t1, &e)) {  
  
 printf("General solve is {");  
 output\_set(&res\_emp);  
 printf("}\n");  
 printf("Its power is 1");  
 }  
  
 } else printf("No any sets");  
  
  
}

1. Вычислю значения множеств

Множество является подмножеством , следовательно – уравнение имеет решения

1. Определю мощность общего решения, найду некоторые частные решения

Частное решение –

Поскольку , то частное решение будет является единственным решением данного теоретико-множественного уравнения