Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное учреждение высшего образования  
«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет  
Кафедра «Информационные технологии и автоматизированные системы»

**О Т Ч Ё Т**

Дискретная математика

Семестр 3

Лабораторная работа 1

Выполнил  
студент группы РИС 22-2б  
Баяндин К. С.

Проверила  
старший преподаватель кафедры ИТАС  
Рустамханова Г.И.

Пермь, 2023

Постановка задачи

Создать калькулятор множеств со следующим функционалом:

1. Создание универсума из целых чисел, заданием границ
2. Создание множеств, входящих в универсум:

* Случайными элементами из универсума по заданной мощности
* Случайными элементами из универсума по одному из условий: четность, знак, кратность.
* Перечислением элементов с клавиатуры

1. Меню со всеми возможными операциями
2. Ввод формулы для вычисления строкой

Алгоритмы работы программы

Множества в программе хранятся в формате массив в массиву (int[][] mnog), где все числа множества заносятся в массив, который в свою очередь заносится в массив множеств.

Программа состоит из 4 основных методов:

1. Создание универсума (диапазона)
2. Создание множеств (условием, перечислением или случайно)
3. Ввод строки, которую надо вычислить
4. Завершение программы

В программе реализованы все операции над множествами:

* Объединение – добавляющее в новое множество элементы сначала из первого множества, а потом из второго.
* Пересечение – добавляющее в новое множество элементы из первого множества, если они также находятся и во втором.
* Разность – добавляющая в новое множество элементы из первого множества, если они не содержатся во втором.
* Симметрическая разность – добавляющая в новое множество сначала разность первого и второго множества, а потом – второго и первого.
* Дополнение – добавляющее в новое множество все элементы универсума, не содержащиеся в старом множестве.

Все эти функции возвращают новое множество, что позволяет сохранять полученный результат как новое множество в программе. Так как множество не может хранить одинаковые элементы, то предусмотрена функция удаления дубликатов *RemoveDuplicates()*

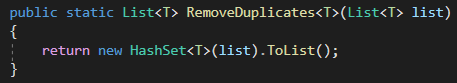


Рисунок 1 – функция удаления дубликатов

Остановимся на 3 этапе более подробно. Ввод строки происходит по шаблону, который задается регулярным выражением: *(@"^([\(d]\*[A-Z]\s[-+/t]\s[\(d]\*[A-Z]\)\*\s\*(\s\*[-+/t]\s\*[\(d]\*[A-Z]\)\*\s\*)\*){1,24}$").* В случае если введенная строка не удовлетворяет шаблону, программа вернется на главное меню. Далее строка передается в функцию *UnidentifiedExpression()*, в которой строка перебирается с помощью цикла *foreach().* Далее возможны 3 случая.

1. Встретилась *)* или *(*.

Логика работы со скобками: если есть закрывающая скобка *)*, то есть и открывающая скобка *(*. В таком случае, индексы всех *(* заносятся в список позиций этих скобок, а как только встречается *)*, создается подстрока от индекса последней *(* до первой *)*.

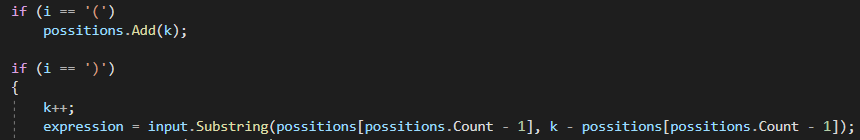


Рисунок 2 – логика работы со скобками

1. Встретился знак операции над множествами

Если встречается знак операции над множествами (-+/td), то берутся множества слева (индекс на 2 элемента влево) и справа (индекс на 2 элемента вправо) и выполняются операции над множествами через функции.

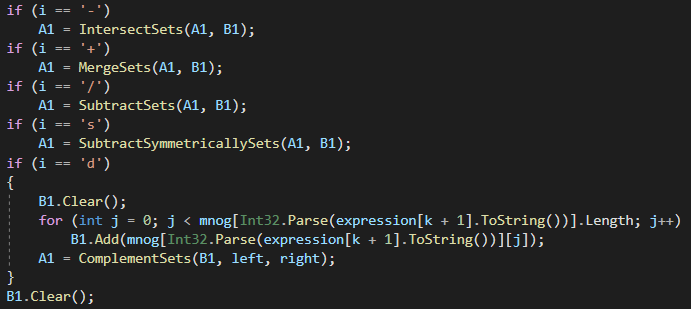


Рисунок 3 – операции над множествами

1. Встретился \_ или множество

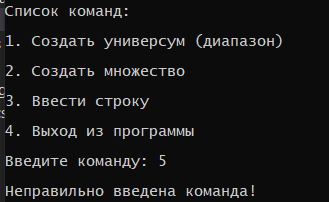
Цикл пропускает данные элементы и работает дальше.

**Тестирование программы**

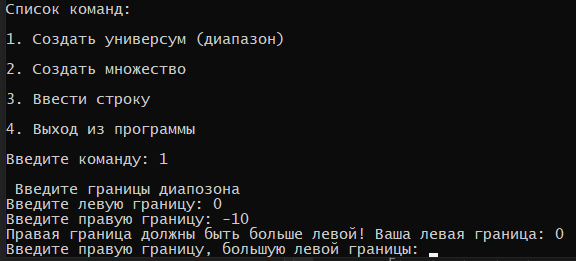
Для проверки правильности работы программы проведены тесты, направленные на выявление критических ошибок и ошибок вычисления. Результаты тестирования приведены в таблице 1.

Таблица 1 – Результаты тестирования

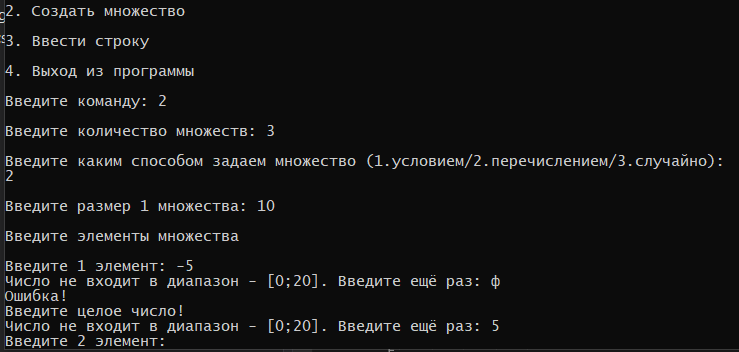
|  |  |  |  |
| --- | --- | --- | --- |
| Описание теста | Ввод | Вывод | Соответствие ожидаемому ответу |
| а) Введена несуществующая команда | Буквы, дробные числа, enter, цифры не входящие в диапазон [0-4] | Неправильно введена команда! | Программа не завершает работу, а запрашивает повторный ввод значения |
| б) Неправильно введен универсум | Правая граница больше левой, буквы, дробные числа, enter | Правая граница должны быть больше левой! Введите правую границу, большую левой границы: | Программа не завершает работу, а запрашивает повторный ввод правой границы |
| в) При создании множества введено не число или число, выходящее за рамки диапазона | Буквы, дробные числа, enter, цифры не входящие в универсум | 1) Ошибка! Введите целое число!  2) Число не входит в заданный диапазон, введите ещё раз: | Программа не завершает работу, а запрашивает повторный ввод значения |
| г) При заполнении множества случайными элементами размер множества введен больше размера диапазона | Ввод числа, большего размера диапазона | Множество не может быть больше (размер больше заданного диапазона) | Программа не завершает работу, а запрашивает повторный ввод значения |
| д) Не верный формат строки для вычисления | Строка несоответствующая шаблону строки | Выражение задано неверно! | Программа переходит на главное меню |



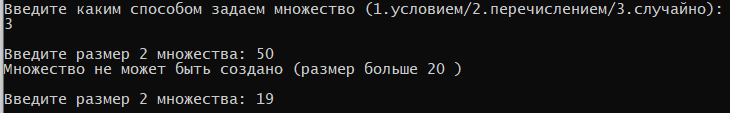
а) Введена несуществующая команда



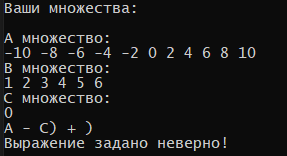
б) Неправильно введен универсум



в) При создании множества введено не число или число, выходящее за рамки диапазона

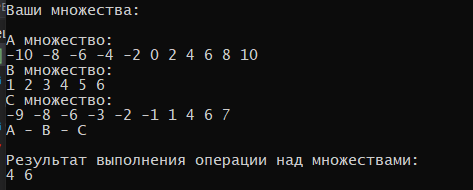


г) При заполнении множества случайными элементами размер множества введен больше размера диапазона

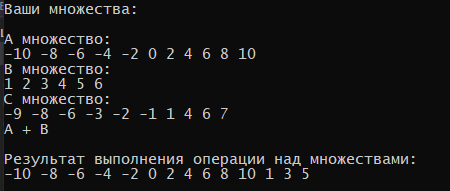


д) Формула введена неверно

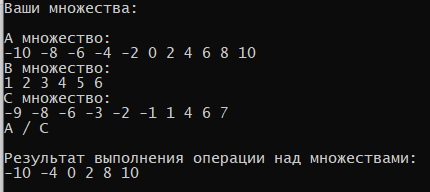
**Результаты работы программы**

****

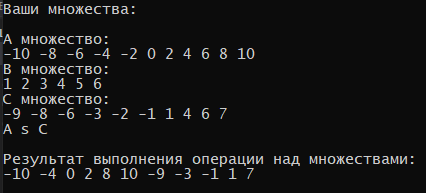
а) Пересечение множеств



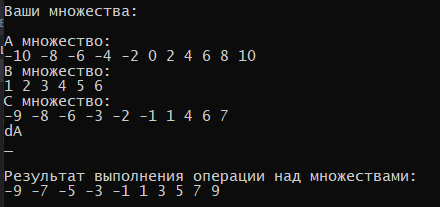
б) Объединение множеств



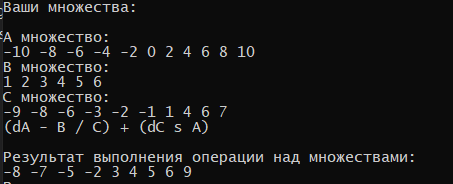
в) Разность множеств



г) Симметрическая разность



д) Дополнение



е) Вычисление сложной строки, на которой представлены все операции

Листинг

**Функции операций:**

//пересечение множеств

public static List<int> IntersectSets(List<int> mnog1, List<int> mnog2)

{

List<int> res = new List<int>();

for (int i = 0; i < mnog1.Count; i++)

{

for (int j = 0; j < mnog2.Count; j++)

{

if (mnog1[i] == mnog2[j])

res.Add(mnog1[i]);

}

}

res = RemoveDuplicates(res);

return res;

}

//объединение множеств

public static List<int> MergeSets(List<int> mnog1, List<int> mnog2)

{

List<int> res = new List<int>();

for (int i = 0; i < mnog1.Count; i++)

res.Add(mnog1[i]);

for (int i = 0; i < mnog2.Count; i++)

res.Add(mnog2[i]);

res = RemoveDuplicates(res);

return res;

}

//дополнить множества

public static List<int> ComplementSets(List<int> mnog, int left, int right)

{

List<int> res = new List<int>();

//заполнение листа числами в диапозоне

for (int i = left; i <= right; i++)

{

res.Add(i);

}

for (int i = 0; i < mnog.Count; i++)

{

res.Remove(mnog[i]);

}

return res;

}

//разность множеств

public static List<int> SubtractSets(List<int> mnog1, List<int> mnog2)

{

List<int> res = new List<int>();

for (int i = 0; i < mnog1.Count; i++)

{

res.Add(mnog1[i]);

}

for (int i = 0; i < mnog2.Count; i++)

{

res.Remove(mnog2[i]);

}

return res;

}

//симметрическая разность множеств

public static List<int> SubtractSymmetricallySets(List<int> mnog1, List<int> mnog2)

{

List<int> res = new List<int>();

//добавление всех элементов в лист и удаление повторяющихся элементов

for (int j = 0; j < mnog1.Count; j++)

{

int indexElem = res.IndexOf(mnog1[j]);

//если такое число уже есть, то удаляем его по индексу

if (indexElem != -1)

res.RemoveAt(indexElem);

//если такого числа нет, то добавляем его в конечное множество

else

res.Add(mnog1[j]);

}

for (int j = 0; j < mnog2.Count; j++)

{

int indexElem = res.IndexOf(mnog2[j]);

//если такое число уже есть, то удаляем его по индексу

if (indexElem != -1)

res.RemoveAt(indexElem);

//если такого числа нет, то добавляем его в конечное множество

else

res.Add(mnog2[j]);

}

return res;

}

**Главная функция:**

static void Main(string[] args)

{

string[] menu = {

"\n\nСписок команд: \n",

"1. Создать универсум (диапазон)\n",

"2. Создать множество\n",

"3. Ввести строку\n",

"4. Выход из программы\n",

"Введите команду: "

};

bool stop = false; //переменная отвечающая за остановку программы

int left = 0, right = 0;

int numMn = 1; // границы диапазона и количество множеств

int[][] mnog = new int[numMn][]; //множества

for (; ; )

{

//вывод меню

for (int i = 0; i < menu.Length; i++)

{

Console.WriteLine(menu[i]);

}

int command = Convert.ToInt32(Console.ReadLine());

Console.Write("\n");

switch (command)

{

case 1: //Создание универсума (диапазона)

Console.WriteLine(" Введите границы диапозона ");

left = InputInt("Введите левую границу: ");

right = InputInt("Введите правую границу: ");

if (right <= left)

{

Console.WriteLine("Правая граница должны быть больше левой! Ваша левая граница: " + left);

right = InputInt("Введите правую границу, большую левой границы: ");

}

break;

case 2: //Создание множеств

numMn = InputInt("Введите количество множеств: ");

mnog = new int[numMn][];

//цикл перебора всех множеств

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

{

//выбор способа задания множества

Console.WriteLine("\nВведите каким способом задаем множество (1.условием/2.перечислением/3.случайно): ");

string method = Console.ReadLine();

//задание множества перечислением

if (method == "перечислением" || method == "2")

{

int sizeMn = InputInt("\nВведите размер " + (i + 1) + " множества: ");

mnog[i] = new int[sizeMn];

Console.WriteLine(" \nВведите элементы множества\n ");

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

{

int number = InputInt("Введите " + (j + 1) + " элемент: ");

//проверка на вхождение в диапазон

while (number < left || number > right)

number = InputInt("Число не входит в диапазон - " + "[" + left + ";" + right + "]" + ". Введите ещё раз: ");

mnog[i][j] = number;

}

}

//задание множества условием

if (method == "условием" || method == "1")

{

Console.WriteLine("\nВведите условие (1.четность/2.нечетность/3.кратность: ");

string conditions = Console.ReadLine();

List<int> listInd = new List<int>();

//задание множества из четных элементов

if (conditions == "1" || conditions == "четность")

{

//поиск 1 элемента, удовлетворяющего условию

int elem = 0;

if (left % 2 == 0)

elem = left;

if (left % 2 != 0)

elem = left + 1;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += 2;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

//задание множества из нечетных элементов

if (conditions == "2" || conditions == "нечетность")

{

//поиск 1 элемента, удовлетворяющего условию

int elem = 0;

if (left % 2 == 0)

elem = left + 1;

if (left % 2 != 0)

elem = left;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += 2;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

if (conditions == "3" || conditions == "кратность")

{

int numberMultiplicity = InputInt("\nВведите какому числу кратны числа в множестве");

//поиск 1 элемента, удовлетворяющего условию

int elem = left;

while (elem % numberMultiplicity != 0 && elem <= right)

elem++;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += numberMultiplicity;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

}

//заполнение массива случайными числами

if (method == "случайно" || method == "3")

{

int sizeMn = InputInt("\nВведите размер " + (i + 1) + " множества: ");

mnog[i] = new int[sizeMn];

Random rnd = new Random();

byte[] bytes = new byte[100];

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

{

rnd.NextBytes(bytes);

mnog[i][j] = rnd.Next(left, right);

}

}

}

OutputMnog(mnog, numMn);

break;

case 3: //строка

OutputMnog(mnog, numMn);

List<int> res = new List<int>();

Regex regexExpression = new Regex(@"^([\(d]\*[A-Z]\s[-+/t]\s[\(d]\*[A-Z]\)\*\s\*(\s\*[-+/t]\s\*[\(d]\*[A-Z]\)\*\s\*)\*){1,24}$");

string input = Console.ReadLine();

if (!regexExpression.IsMatch(input))

{

System.Console.WriteLine("Выражение задано неверно!");

Console.ReadKey();

break;

}

else if (regexExpression.IsMatch(input))

{

res = UnidentifiedExpression(mnog, input, left, right);

}

OutputResult(res);

break;

case 4: //Выход из программы

stop = true;

break;

default:

Console.Write("Неправильно введена команда!");

break;

}

//Если была введена команда 4, то цикл прервется

if (stop)

break;

}

}static void Main(string[] args)

{

string[] menu = {

"\n\nСписок команд: \n",

"1. Создать универсум (диапазон)\n",

"2. Создать множество\n",

"3. Ввести строку\n",

"4. Выход из программы\n",

"Введите команду: "

};

bool stop = false; //переменная отвечающая за остановку программы

int left = 0, right = 0;

int numMn = 1; // границы диапазона и количество множеств

int[][] mnog = new int[numMn][]; //множества

for (; ; )

{

//вывод меню

for (int i = 0; i < menu.Length; i++)

{

Console.WriteLine(menu[i]);

}

int command = Convert.ToInt32(Console.ReadLine());

Console.Write("\n");

switch (command)

{

case 1: //Создание универсума (диапазона)

Console.WriteLine(" Введите границы диапозона ");

left = InputInt("Введите левую границу: ");

right = InputInt("Введите правую границу: ");

if (right <= left)

{

Console.WriteLine("Правая граница должны быть больше левой! Ваша левая граница: " + left);

right = InputInt("Введите правую границу, большую левой границы: ");

}

break;

case 2: //Создание множеств

numMn = InputInt("Введите количество множеств: ");

mnog = new int[numMn][];

//цикл перебора всех множеств

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

{

//выбор способа задания множества

Console.WriteLine("\nВведите каким способом задаем множество (1.условием/2.перечислением/3.случайно): ");

string method = Console.ReadLine();

//задание множества перечислением

if (method == "перечислением" || method == "2")

{

int sizeMn = InputInt("\nВведите размер " + (i + 1) + " множества: ");

mnog[i] = new int[sizeMn];

Console.WriteLine(" \nВведите элементы множества\n ");

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

{

int number = InputInt("Введите " + (j + 1) + " элемент: ");

//проверка на вхождение в диапазон

while (number < left || number > right)

number = InputInt("Число не входит в диапазон - " + "[" + left + ";" + right + "]" + ". Введите ещё раз: ");

mnog[i][j] = number;

}

}

//задание множества условием

if (method == "условием" || method == "1")

{

Console.WriteLine("\nВведите условие (1.четность/2.нечетность/3.кратность: ");

string conditions = Console.ReadLine();

List<int> listInd = new List<int>();

//задание множества из четных элементов

if (conditions == "1" || conditions == "четность")

{

//поиск 1 элемента, удовлетворяющего условию

int elem = 0;

if (left % 2 == 0)

elem = left;

if (left % 2 != 0)

elem = left + 1;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += 2;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

//задание множества из нечетных элементов

if (conditions == "2" || conditions == "нечетность")

{

//поиск 1 элемента, удовлетворяющего условию

int elem = 0;

if (left % 2 == 0)

elem = left + 1;

if (left % 2 != 0)

elem = left;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += 2;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

if (conditions == "3" || conditions == "кратность")

{

int numberMultiplicity = InputInt("\nВведите какому числу кратны числа в множестве");

//поиск 1 элемента, удовлетворяющего условию

int elem = left;

while (elem % numberMultiplicity != 0 && elem <= right)

elem++;

int k = 0;

//добавление всех элементов в список

while (elem <= right)

{

listInd.Add(elem);

k++;

elem += numberMultiplicity;

}

mnog[i] = new int[listInd.Count];

for (int z = 0; z < listInd.Count; z++)

mnog[i][z] = listInd[z];

listInd.Clear();

}

}

//заполнение массива случайными числами

if (method == "случайно" || method == "3")

{

int sizeMn = InputInt("\nВведите размер " + (i + 1) + " множества: ");

mnog[i] = new int[sizeMn];

Random rnd = new Random();

byte[] bytes = new byte[100];

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

{

rnd.NextBytes(bytes);

mnog[i][j] = rnd.Next(left, right);

}

}

}

OutputMnog(mnog, numMn);

break;

case 3: //строка

OutputMnog(mnog, numMn);

List<int> res = new List<int>();

Regex regexExpression = new Regex(@"^([\(d]\*[A-Z]\s[-+/t]\s[\(d]\*[A-Z]\)\*\s\*(\s\*[-+/t]\s\*[\(d]\*[A-Z]\)\*\s\*)\*){1,24}$");

string input = Console.ReadLine();

if (!regexExpression.IsMatch(input))

{

System.Console.WriteLine("Выражение задано неверно!");

Console.ReadKey();

break;

}

else if (regexExpression.IsMatch(input))

{

res = UnidentifiedExpression(mnog, input, left, right);

}

OutputResult(res);

break;

case 4: //Выход из программы

stop = true;

break;

default:

Console.Write("Неправильно введена команда!");

break;

}

//Если была введена команда 4, то цикл прервется

if (stop)

break;

}

}