МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра теоретических основ компьютерной безопасности и криптографии

Комбинаторная теория полугрупп

ОТЧЁТ

ПО ДИСЦИПЛИНЕ

«ПРИКЛАДНАЯ УНИВЕРСАЛЬНАЯ АЛГЕБРА»

студентки 3 курса 331 группы

специальности 10.05.01 Компьютерная безопасность

факультета компьютерных наук и информационных технологий

Зиминой Ирины Олеговны

Преподаватель

профессор, д.ф.-м.н. ___________ В. А. Молчанов

подпись, дата

Саратов 2022

СОДЕРЖАНИЕ

1. Цель работы и порядок выполнения

Цель работы — изучение основных понятий теории полугрупп.

Порядок выполнения работы:

1. Рассмотреть понятие полугруппы, подполугруппы и порождающего множества. Разработать алгоритм построения подполугрупп по таблице Кэли.
2. Разработать алгоритм построения полугруппы бинарных отношений по заданному порождающему множеству.
3. Рассмотреть понятия подгруппы, порождающего множества и определяющих соотношений. Разработать алгоритм построения полугруппы по порождающему множеству и определяющим соотношениям.

2. Теоретический материал.

- **Полугруппа.**

Полугруппой $S = (S, \cdot)$ называется непустое множество S с бинарной операцией $\cdot$, удовлетворяющей ассоциативному закону: $(x \cdot y) \cdot z = x \cdot (y \cdot z)$ для $\forall x, y, z \in S$.

- **Подполугруппа.**

Подмножество X полугруппы S называется подполугруппой, если X устойчиво относительно операции умножения, т.е $\forall x, y \in X$ выполняется свойство: $x \cdot y \in X$. В этом случае множество X с ограничением на нём операции умножения исходной полугруппы S образует полугруппу.

- **Порождающее множество.**

В силу общего свойства подалгебр пересечение любого семейства X_i $(i \in I)$ подполугрупп полугруппы S является подполугруппой S и, значит, множество $Sub(S)$ всех подполгрупп полугруппы S является системой замыканий. Следовательно, для любого подмножества X полугруппы S существует наименьшая подполугруппа S, содержащая множество X. Такая полугруппа обозначается символом $\langle X \rangle$ и называется подполугруппой S, порождённой множеством X. При этом множество X называется также порождающим множеством подполугруппы $\langle X \rangle$.
В частности, если $\langle X \rangle = S$, то X называется порождающим множеством полугруппы S и говорят, что множество X порождает полугруппу S.

- **Определяющие соотношения.**

Для любой конечной полугруппы S найдется такой конечный алфавит A, что для некоторого отображения $\emptyset : A \to S$ выполняется равенство $\langle \emptyset(A) \rangle = S$, и следовательно $S \cong A^+ / \ker \emptyset$ в этом случае множество A называется множество порождающих символов полугруппы S (относительно изображения $\emptyset : A \to S$). Очевидно, что в общем случае множество таких соотношений $w_1 = w_2$ для всех пар $(w_1, w_2) \in \ker \emptyset$ будет бесконечным и не представляется возможности эффективно описать полугруппу S в виде полугруппы классов конгруэнции

$\ker\emptyset$. Однако в некоторых случаях можно выбрать такое сравнительно простое подмножество $\rho \subset \ker\emptyset$, которое однозначно определяет конгруэнцию $\ker\emptyset$, как наименьшую конгруэнцию полугруппы A^+, содержащую отношение ρ, т.е $\ker\emptyset = f_{con}(\rho) = f_{eq}(f_{reg}(\rho))$. Так как в случае $(w_1, w_2) \in \rho$ по прежнему выполняется равенство $\emptyset(w_1) = \emptyset(w_2)$, то будем писать $w_1 = w_2$ и называть такие выражения определяющими соотношениями. Из таких соотношений конгруэнция $\ker\emptyset$ строится с помощью применения следующих процедур к словам $u, v \in A^+$:

- слово v непосредственно выводится из слова u, если v получается из u заменой некоторого подслова w_1 на слово w_2, удолетворяющее определяющему соотношению $w_1 = w_2$, т.е $(u, v) = (xw_1y, xw_2y)$ для некоторых $x, y \in A^*$;
- слово v выводится из слова u, если v получается из u с помощью конечного числа применения процедуры 1.

Если все выполняющиеся на S соотношения выводятся из определяющих соотношений совокупности ρ и выражение $\langle A : w_1 = w_2 : (w_1, w_2) \in \rho \rangle$ называется копредставлением полугруппы S.

3. Алгоритм построения подполугрупп по таблице Кэли.

Входные данные: полугруппа S с таблицей Кэли $cayley\_table$ размерности $(N * N)$ и множество $X \subset S$.

Выходные данные: построенная подполугруппа $\langle X \rangle \subset S$.

Описание алгоритма:

1) Положим $i = 0, X_0 = X$.
2) Для X_i вычисляется $\bar{X}_i = \{x \cdot y : x \in X_i \wedge y \in X_i\}$ и положим $X_{i+1} = X_i \cup \bar{X}_i$.
3) Вычисляется:

$$\langle X \rangle = \bigcup_{i=0}^{\infty} X_i$$

4) Выводится построенная подполугруппа $\langle X \rangle \subset S$.

Сложность алгоритма: $O(N^3)$.

4. Алгоритм построения полугруппы бинарных отношений по заданному порождающему множеству.

Входные данные: M – количество матриц в порождающем множестве, булевые матрицы размерности $N * N$.

Выходные данные: построенная полугруппа $\langle X \rangle$.

Описание алгоритма:

1) Создается сет $sets$ – вектор векторов с числами, в который добавляется очередная введенная матрица.
2) Создается контейнер символов и матриц in_matrix, в который добавляется буква очередной матрицы и сама эта матрица.
3) Создается контейнер матриц и символов out_matrix, в который добавляется очередная матрица и её буква.
4) Создается сет матриц $group$ по сути копия сета $sets$.
5) С помощью прохода по каждому элементу $sets$ и дополнительной функции $multy\_matrix$ (выполняющую логическое умножение

элементов матрицы) создается новая булевая матрица new_matrix. Если эта матрица еще не присутствует в сете матриц $group$, то new_matrix добавляется в контейнеры in_matrix и out_matrix и добавляется в сет матриц $group$.

6) После этого начинается вывод результата:

   С помощью использования функции $insert\_matrix$ сет $sets$ теперь хранит исходное матричное множество и матрично перемноженные подмножества множества. Выводится итоговая полугруппа: булевые матрицы. И таблица Кэли.

Сложность алгоритма: $O(N^3 * 2^{N^2})$.

5. Алгоритм построения полугруппы по порождающему множеству и определяющим соотношениям.

Входные данные: N – количество букв в множестве, сет строк A – алфавит, M – количество определяющих отношений, контейнер строк R – определяющие отношения.

Выходные данные: построенная полугруппа и её таблица Кэли.

Описание алгоритма:

1) Создается переменная $count$, её значение равно одному.
2) Создается сет строк S_cur, и заполняется значением алфавита.
3) Запускается цикл пока S_cur не пустой сет:

   Значение переменой $count$ увеличивается на один. Для каждого элемента из сета S_cur и для каждого элемента сета A используется функция $new\_element$ – нахождение нового элемента для полугруппы по сумме элементов и сетов S_cur и A.

   После этого из сета S_cur удаляются все элементы. И запускается цикл прохода по всем элементам полугруппы, если размер элемента равен переменной $count$, то этот элемент добавляется в сет S_cur.

4) Полученная полугруппа добавляется в вектор строк $halfgroup$ и сортируется. Размер вектора $halfgroup$ будет соответствовать размерности таблице Кэли.
5) Выводится построенная полугруппа и её таблица Кэли.

Сложность алгоритма: $O(N^N)$, потому что может быть бесконечное определение не эквивалентных конгруэнции слов.

6. Реализация рассмотренных алгоритмов.

6.1 Алгоритм построения подполугрупп по таблице Кэли.

Программный код:

```
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>
#include <windows.h>

using namespace std;

bool check_associativity(const vector<pair<int, int>>& result, const
vector<pair<int, int>>& prov) {

    if (size(result) != size(prov)) {
        return true;
    }

    int count_elements = 0;
    for (auto& [first, junk1] : result) {
        for (auto& [second, junk2] : prov) {
            if (first == second) {
                ++count_elements;
            }
        }
    }

    return size(result) != count_elements;
}

auto unite(const vector<pair<int, int>>& result, const vector<pair<int, int>>&
resSt) {
    vector<pair<int, int>> temp_res(result);

    for (int i = 0; i < size(resSt); ++i) {
        int count_elements = count_if(begin(temp_res), end(temp_res), [&resSt,
i](auto element) {
            return resSt[i].first == element.first && resSt[i].second ==
element.second;
        });

        if (count_elements == 0) {
            temp_res.push_back(resSt[i]);
        }
    }
```

```
    return temp_res;
}

vector<pair<int, int>> subpool_group(vector<vector<int>>& cayley_table,
vector<pair<int, int>>& subset, vector<int>& half_group) {
    vector<pair<int, int>> res = subset;
    vector<pair<int, int>> check;

    while (true) {
        vector<pair<int, int>> temp;
        for (int i = 0; i < size(subset); ++i) {
            for (int j = 0; j < size(res); ++j) {
                int element = cayley_table[subset[i].second][res[j].second];

                temp.push_back({element, find(begin(half_group),
end(half_group), element) - begin(half_group)});
            }
        }

        check = res;
        res = unite(res, temp);

        if (!check_associativity(res, check)) {
            break;
        }
    }

    return res;
}

int main() {

    SetConsoleOutputCP(CP_UTF8);

    int n;
    cout << "Количество элементов в полугруппе:" << endl;
    cin >> n;

    vector<int> half_group(n);
    int element;
    cout << "элементы полугруппы:" << endl;
    for (int i = 0; i < n; ++i) {
        cin >> element;
        half_group[i] = element;
    }

    vector<vector<int>> cayley_table(n, vector<int>(n));
    cout << "Таблица Кэли:" << endl;
    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cin >> cayley_table[i][j];
        }
    }

    cout << "Количество элементов в подмножестве: " << endl;
    int m;
    cin >> m;

    vector<pair<int, int>> subset;
    cout << "Элементы подмножества: " << endl;
    for (int i = 0; i < m; ++i) {
        int element;
```

```
        cin >> element;
        subset.push_back({element, find(begin(half_group), end(half_group),
element) - begin(half_group)});
    }

    vector<pair<int, int>> result;
    result = subpool_group(cayley_table, subset, half_group);
    sort(begin(result), end(result));

    cout << "Подполугруппа: ";
    bool is_first = true;
    cout << "{";
    for (auto [first, junk] : result) {
        cout << (is_first ? "" : ", ") << first;
        is_first = false;
    }
    cout << "}" << endl;

    return 0;
}
```

Результаты тестирования:

```
Количество элементов в полугруппе:
5
элементы полугруппы:
1 2 3 4 5
Таблица Кэли:
1 1 1 1 1
1 2 3 4 5
1 3 5 2 4
1 4 2 5 3
1 5 4 3 2
Количество элементов в подмножестве:
2
Элементы подмножества:
2 3
Подполугруппа: {2, 3, 4, 5}
```

6.2 Алгоритм построения полугруппы бинарных отношений по заданному порождающему множеству.

Программный код:

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <iomanip>
#include <windows.h>

using namespace std;
```

```
vector<vector<int>> multy_matrix(const vector<vector<int>>& sets, const
vector<vector<int>>& result, int n) {
    vector <vector <int>> temp_result(n, vector<int>(n));

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            for (int k = 0; k < n; ++k) {
                if (sets[i][k] == 1 && result[k][j] == 1) {
                    temp_result[i][j] = 1;
                    break;
                }
                else temp_result[i][j] = 0;
            }
        }
    }
    return temp_result;
}

set<vector<vector<int>>> insert_matrix(set<vector<vector<int>>> sets, int n) {
    for (auto& sets1 : sets)
        for (auto& sets2 : sets) {
            sets.insert(multy_matrix(sets1, sets2, n));
        }
    return sets;
}

int main() {

    SetConsoleOutputCP(CP_UTF8);

    int m;
    cout << "Количество матриц в порождающем множестве: " << endl;
    cin >> m;

    int n;
    cout << "Размер матриц: " << endl;
    cin >> n;

    vector<vector<int>> matrix(n, vector<int>(n));
    map<char, vector<vector<int>>> in_matrix;
    map<vector<vector<int>>, char> out_matrix;

    set<vector<vector<int>>> sets;
    int q = 0;
    for (int i = 1; i <= m; i++) {
        cout << "Матрица " << char(65 + q) << ":" << endl;
        for (int i = 0; i < n; ++i) {
            for (int j = 0; j < n; j++)
                cin >> matrix[i][j];
        }
        sets.insert(matrix);
        in_matrix.insert({char(65 + q), matrix});
        out_matrix.insert({matrix, char(65 + q++)});
    }

    set<vector<vector<int>>> group;
    group = sets;
    int count = 0;
    while (true) {
        for (auto& i: sets) {
            for (auto& j : sets) {
                vector<vector<int>> new_matrix = multy_matrix(i, j, n);
```

```
                if (!group.count(new_matrix)) {
                     in_matrix.insert({char(65 + q + count), new_matrix});
                     out_matrix.insert({new_matrix, char(65 + q + count++)});
                    group.insert(new_matrix);
                }
            }
        }

        if (group == sets) {

            group = sets;
            sets = insert_matrix(sets, n);

            cout << endl << "Итоговая полугруппа:" << endl;
            for (const vector<vector<int>>& temp_result : sets) {
                cout << out_matrix[temp_result];
                for (int i = 0; i < size(temp_result); ++i) {
                    cout << endl;
                    for (int j = 0; j < size(temp_result); ++j)
                        cout << temp_result[i][j] << ' ';
                }
                cout << endl;
            }

            cout << endl << "Таблица Кэли: " << endl << ' ';
            for (int i = 0; i < size(out_matrix); ++i){
                cout << setw(4) << char(65 + i);
            }

            cout << endl;

            for (int i = 0; i < size(out_matrix); ++i) {
                cout << char(65 + i) << setw(4);
                for (int j = 0; j < size(out_matrix); ++j){
                    cout << setw(4) << out_matrix[multy_matrix(in_matrix[char(65
+ i)], in_matrix[char(65 + j)], n)];
                }
                cout << endl;
            }
            return 0;

        } else {
            sets = group;
        }
    }

    return 0;
}
```

Результаты тестирования:

```
Количество матриц в порождающем множестве:
2
Размер матриц:
3
Матрица A:
0 1 0
1 1 1
0 0 0
Матрица B:
1 0 1
0 1 0
1 0 1
```

```
Итоговая полугруппа:
A
0 1 0
1 1 1
0 0 0
D
0 1 0
1 1 1
0 1 0
B
1 0 1
0 1 0
1 0 1
C
1 1 1
1 1 1
0 0 0
E
1 1 1
1 1 1
1 1 1

Таблица Кэли:
    A   B   C   D   E
A   C   A   C   C   C
B   D   B   E   D   E
C   C   C   C   C   C
D   E   D   E   E   E
E   E   E   E   E   E
```

6.3 Алгоритм построения полугруппы по порождающему множеству и определяющим соотношениям.

Программный код:

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <set>
#include <map>
#include <iomanip>
#include <windows.h>

using namespace std;

set<string> set_of_R;
map<string, string> R;
set<string> S_cur, S, copy_S;
int maximum = 0;
string cur_word;
vector<vector<string>> cayley_table;


vector<string> split(string word) {
    vector <string> splited_word;

    for (int i = 0; i < size(word); ++i) {
        splited_word.push_back(word.substr(i, 1));
    }

    return(splited_word);
}


bool comp(string A, string B) {
    return (size(A) < size(B)) || (size(A) == size(B) && A < B);
}


void new_element(string word, bool flag = false, int row = 0, int size = -1) {
    if (set_of_R.find(word) != end(set_of_R) && !flag)
        return;

    vector<string> temp_word = split(word);

    int substr = 2;
    while (substr <= maximum) {
        for (int i = temp_word.size() - 1; i >= 0; --i) {
            if (substr > word.size() || substr > i + 1)
                break;

            string part = "";

            for (int j = i; j > i - substr; --j) {
                part = temp_word[j] + part;
            }

            if (R.find(part) != end(R)) {
                cur_word = "";
                vector<string> temp_part = split(R[part]);

                if (!flag) {
```

```
                    for (int q = 0; q < i - substr; ++q) {
                        cur_word = cur_word + temp_word[q];
                    }
                } else {
                    for (int q = 0; q < i - substr + 1; ++q) {
                        cur_word = cur_word + temp_word[q];
                    }
                }

                cur_word = cur_word + R[part];

                for (int q = i + 1; q < temp_word.size(); ++q) {
                    cur_word = cur_word + temp_word[q];
                }

                if (cur_word.size() < word.size() && !flag) {
                    set_of_R.insert(word);
                    set_of_R.insert(cur_word);
                    return;
                } else if (set_of_R.find(cur_word) != set_of_R.end() && !flag) {
                    return;
                } else if (flag && copy_S.find(cur_word) != copy_S.end()) {
                    if (cayley_table[row].size() < size) {
                        cayley_table[row].push_back(cur_word);
                    }
                    return;
                } else {
                    int check = cayley_table[row].size();
                    new_element(cur_word, flag, row, size);
                    if (flag && cayley_table[row].size() > check) {
                        return;
                    }
                }
            }

            if (set_of_R.find(word) != end(set_of_R) && !flag) {
                return;
            }

            if (flag && copy_S.find(word) != end(copy_S)) {
                if (cayley_table[row].size() <= size)
                    cayley_table[row].push_back(word);
                return;
            }
        }

        ++substr;
    }

    if (cur_word.size() == word.size() && cur_word < word && set_of_R.find(word)
== set_of_R.end() && set_of_R.find(cur_word) == set_of_R.end()) {
        S.insert(cur_word);
        set_of_R.insert(word);
        set_of_R.insert(cur_word);
    } else {
        S.insert(word);
        set_of_R.insert(word);
        set_of_R.insert(cur_word);
    }
}

int main() {
```

```
SetConsoleOutputCP(CP_UTF8);

int n;
cout << "Количество букв в множестве: " << endl;
cin >> n;

set<string> A;
cout << "Алфавит: " << endl;
for (int i = 0; i < n; ++i) {
    string a;
    cin >> a;
    A.insert(a);
    S.insert(a);
    set_of_R.insert(a);
}

int m;
cout << "Количество определяющих отношений: " << endl;
cin >> m;

cout << "Определяющие отношения: " << endl;
for (int i = 0; i < m; ++i) {
    string r1, r2;
    cin >> r1 >> r2;

    if (size(r1) > maximum)
        maximum = size(r1);
    if (r2.size() > maximum)
        maximum = size(r2);

    if (size(r1) < size(r2))
        R[r2] = r1;
    else if (size(r1) > size(r2))
        R[r1] = r2;
    else {
        if (r1 < r2)
            R[r2] = r1;
        else
            R[r1] = r2;
    }
}

int count = 1;
S_cur = A;
while (!empty(S_cur)) {
    ++count;

    for (string s : S_cur) {
        for (string letter : A) {
            new_element(s + letter);
        }
    }

    S_cur.clear();
    for (auto s : S) {
        if (size(s) == count) {
            S_cur.insert(s);
        }
    }

}

vector<string> halfgroup;
```

```
    for (string s : S) {
        halfgroup.push_back(s);
    }
    sort(begin(halfgroup), end(halfgroup), comp);

    cout << " Полугруппа: {";
    for (int i = 0; i < size(halfgroup); ++i) {
        if (i == size(halfgroup) - 1) {
            cout << halfgroup[i] << "}" << endl;
        } else {
            cout << halfgroup[i] << " ";
        }
    }

    cayley_table.resize(size(halfgroup));
    copy_S = S;
    for (int i = 0; i < size(halfgroup); ++i) {
        for (int j = 0; j < size(halfgroup); ++j) {
            new_element(halfgroup[i] + halfgroup[j], true, i, size(halfgroup));
        }
    }

    cout << endl << "Таблица Кэли: " << endl;
    cout << "     ";
    for (int i = 0; i < size(halfgroup); ++i){
        cout << setw(5) << halfgroup[i];
    }
    cout << endl;
    for (int i = 0; i < size(halfgroup); ++i) {
        cout << setw(5) << halfgroup[i] << setw(5);
        for (int j = 0; j < size(halfgroup); ++j) {
            cout << cayley_table[i][j] << setw(5);
        }
        cout << endl;
    }

    cout << endl;

    return 0;
}
```

Результаты тестирования:

```
Количество букв в множестве:
2
Алфавит:
a b
Количество определяющих отношений:
3
Определяющие отношения:
ab ba
aaa b
bb a
 Полугруппа: {a b aa ab aab}

Таблица Кэли:
        a    b   aa   ab  aab
    a   aa   ab    b  aab    a
    b   ab    a  aab   aa    b
   aa    b  aab   ab    a   aa
   ab  aab   aa    a    b   ab
  aab    a    b   aa   ab  aab
```

7. Решение задач по варианту 3.

Задание 1.

Найдите полугруппу $S=\langle f,g\rangle$ преобразований мн-ва $X=\{1,2,3\}$, порождённую следующими преобразованиями f,g. в симметричной полугруппе $T(X)$ преобразований мн-ва X:

$$f=\begin{pmatrix}1&2&3\\2&2&1\end{pmatrix}\quad g=\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}$$

Решение:

Мн-во преобразований f,g порождает полугруппу $S=\langle f,g\rangle$ преобразований множества X, которая состоит из элементов: $f, g, f^2, fg, gf, g^2, \ldots$ и является подполугруппой конечной полугруппы $T(X)$.

$$f^2=\begin{pmatrix}1&2&3\\2&2&1\end{pmatrix}\cdot\begin{pmatrix}1&2&3\\2&2&1\end{pmatrix}=\begin{array}{ccc}1&2&3\\\downarrow&\downarrow&\downarrow\\2&2&1\\\downarrow&\downarrow&\downarrow\\2&2&2\end{array}=\begin{pmatrix}1&2&3\\2&2&2\end{pmatrix}$$

$$fg=\begin{pmatrix}1&2&3\\2&2&1\end{pmatrix}\cdot\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}=\begin{array}{ccc}1&2&3\\\downarrow&\downarrow&\downarrow\\2&2&1\\\downarrow&\downarrow&\downarrow\\3&3&3\end{array}=\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}$$

$$gf=\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}\cdot\begin{pmatrix}1&2&3\\2&2&1\end{pmatrix}=\begin{array}{ccc}1&2&3\\\downarrow&\downarrow&\downarrow\\3&3&3\\\downarrow&\downarrow&\downarrow\\1&1&1\end{array}=\begin{pmatrix}1&2&3\\1&1&1\end{pmatrix}$$

$$g^2=\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}\cdot\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}=\begin{array}{ccc}1&2&3\\\downarrow&\downarrow&\downarrow\\3&3&3\\\downarrow&\downarrow&\downarrow\\3&3&3\end{array}=\begin{pmatrix}1&2&3\\3&3&3\end{pmatrix}$$

Таким образом получаем полугруппу $S=\langle f,g,f^2,gf\rangle$, т.к. $fg=g^2=g$, то $fg, g^2\notin S$.

Таблица Кэли:

	f	g	f^2	gf
f	f^2	g	f^2	f^2
g	gf	g	f^2	f^2
f^2	f^2	g	f^2	gf
gf	f^2	g	f^2	f^2

Задание 2

Найдите индекс и период следующих элементов a полугруппы преобразований множества $X=\{1,2,3,4,5\}$:

$$a=\begin{pmatrix}1&2&3&4&5\\5&4&1&2&2\end{pmatrix}$$

Решение:

$$aa=\begin{array}{ccccc}1&2&3&4&5\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 5&4&1&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&5&4&4\end{array}=\begin{pmatrix}1&2&3&4&5\\2&2&5&4&4\end{pmatrix}$$

$$aaa=\begin{array}{ccccc}1&2&3&4&5\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 5&4&1&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&5&4&4\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 4&4&2&2&2\end{array}=\begin{pmatrix}1&2&3&4&5\\4&4&2&2&2\end{pmatrix}$$

$$aaaa=\begin{array}{ccccc}1&2&3&4&5\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 5&4&1&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&5&4&4\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 4&4&2&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&4&4&4\end{array}=\begin{pmatrix}1&2&3&4&5\\2&2&4&4&4\end{pmatrix}$$

$$aaaaa=\begin{array}{ccccc}1&2&3&4&5\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 5&4&1&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&5&4&4\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 4&4&2&2&2\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 2&2&4&4&4\\ \downarrow&\downarrow&\downarrow&\downarrow&\downarrow\\ 4&4&2&2&2\end{array}=\begin{pmatrix}1&2&3&4&5\\4&4&2&2&2\end{pmatrix}$$

Можно заметить, что $aaaaa \to aaa$. Т.е. на 5 преобразований наблюдается цикличность, тогда, если считать элементы полугруппы $\langle a, aa, aaa, aaaa, \dots \rangle$ начиная с 3-х, то каждый $2k$-й элемент будет иметь преобразование $\begin{pmatrix}1&2&3&4&5\\4&4&2&2&2\end{pmatrix}$, а каждый $(2k+1)$-й элемент равен $\begin{pmatrix}1&2&3&4&5\\2&2&4&4&4\end{pmatrix}$, где $k \in N$. Получается, что период = 2, а индекс = 3.

Задание 3

Найдите полугруппу S по следующему ее копредставлению:

$S = \langle x, y : xy = yx, x^3 = x^2, y^2 = x \rangle$

Решение:

Слова длины 1: x, y

Слова длины 2: x^2, $xy = yx$, $y^2 = x$

Слова длины 3: $x^3 = x^2$, x^2y, $xy^2 = x^2$

Слова длины 4: $x^3y = x^2y$, $x^2y^2 = x^3 = x^2$

Таким образом, $S = \{x, y, x^2, xy, x^2y\}$ – полная система представителей классов конгруэнции ε. Операция умножения $\cdot$ таких слов опр. с точностью до конгруэнции ε по след. таблице Кэли:

$\cdot$	x	y	x^2	xy	x^2y
x	x^2	xy	x^2	x^2y	x^2y
y	xy	x	x^2y	x^2	x^2
x^2	x^2	x^2y	x^2	x^2y	x^2y
xy	x^2y	x^2	x^2y	x^2	x^2
x^2y	x^2y	x^2	x^2y	x^2	x^2

ЗАКЛЮЧЕНИЕ

В данной лабораторной работе были рассмотрены теоретические сведения о подгруппах, полугруппах, подполугруппах и порождающем множестве.

Результатом работы является: алгоритм построения подполугрупп по таблице Кэли, алгоритм построения полугруппы бинарных отношения по заданному порождающему множеству, алгоритм построения полугруппы по порождающему множеству и определяющим соотношениям.

Была осуществлена программная реализация описанных алгоритмов и проведено тестирование программ.