Министерство образования Республики Беларусь

Учреждение образования

«Брестский Государственный технический университет»

Кафедра ИИТ

**Лабораторная работа №4**

По дисциплине «Математические основы интеллектуальных систем»

Тема: «Группы»

**Выполнил:**

Студент 2 курса

Группы ИИ-21

Карагодин Д.Л.

**Проверил:**

Козинский А.А.

Брест 2022

**Цель работы:** Изучение и решение заданий по подстановкам, группам.

**Ход работы:**

**Задание 1.**

1. Представить заданную подстановку произведением независимых циклов. Варианты заданий указаны в таблице 1. Подстановка из 6 элементов задана в таблице 1 второй строкой подстановки.
2. Определить степень заданной подстановки.
3. Определить четность подстановки:  
   - через число транспозиций в разложении подстановки в произведение транспозиций;  
   - через число инверсий в подстановке.
4. Задания реализовать программно.

**Задание 2.**

Для заданной в таблице 1 группы:

1. Построить таблицу Кэли. Определить нейтральный элемент. Для каждого элемента группы указать обратный элемент.
2. Определить является ли группа циклической, указать порядок каждого элемента в группе.
3. Указать все подгруппы заданной группы и их порядок. Определить порождающие элементы для подгрупп. Для одной из подгрупп построить таблицу Кэли.
4. Разложить группу на левые смежные классы по каждой из подгрупп. Построить фактор множества для группы по каждой из подгрупп.
5. Построить таблицу Кэли для фактор-группы по одной из подгрупп.

**Задание 3.**

Построить гомоморфизм групп . Дать гомоморфизму характеристику.

**Вариант 4**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Задание 1 | Задание 2,3 | |
| Подстановка | Группа | Множество *A* |
| 4. |  |  |  |

**Код программы:**

#include "1\_lab.hpp"

class Podstanovka {

private:

short kolvo\_vershin;

vector<short> podst\_second\_line;

public:

Podstanovka() {

kolvo\_vershin = 0;}

Podstanovka(string NameOfFile) {

ifstream ifs(NameOfFile);

short i = 0;

while (!ifs.eof()) {

ifs >> i;

podst\_second\_line.push\_back(i);}

kolvo\_vershin = podst\_second\_line.size();

ifs.close();}

Podstanovka(const Podstanovka& Pod2) {

kolvo\_vershin = Pod2.kolvo\_vershin;

podst\_second\_line = Pod2.podst\_second\_line;}

void ToShow() {

cout << "-------------------------\n";

cout <<"Count of elements: " << kolvo\_vershin << endl;

for (size\_t j = 0; j < kolvo\_vershin;j++) { cout << j+1 << " "; }

cout << endl;

for (auto j : podst\_second\_line) { cout << j << " "; }

cout << "\n-------------------------\n";}

vector<short> GetPodstLine() { return podst\_second\_line; }

short GetKolVo() { return kolvo\_vershin; }

short operator[](short i) { return podst\_second\_line[i];}};

void CyclesOfPodstanovka(Podstanovka& Pod) {

cout << "Cycles: ";

vector<bool> visited(Pod.GetKolVo(), false);

vector<short> cycle;

short i,start;

for (size\_t j = 0; j < Pod.GetKolVo(); j++)

{

if (!visited[j]) {

start = i = j;

visited[start] = true;

for (size\_t k = 0; k < Pod.GetKolVo(); k++)

{

i = Pod[i]-1;

visited[i] = true;

cycle.push\_back(i+1);

if (i == start) {

cout << "( ";

for (auto at : cycle) cout << at << " ";

cout << ");";

break;}}

cycle.clear();}}

cout << endl;}

bool ParityTransposition(Podstanovka& Pod) {

cout << "Transposition: ";

vector<bool> visited(Pod.GetKolVo(), false);

vector<short> cycle;

short i, start,count=0;

for (size\_t j = 0; j < Pod.GetKolVo(); j++)

{

if (!visited[j]) {

start = i = j;

visited[start] = true;

for (size\_t k = 0; k < Pod.GetKolVo(); k++)

{

i = Pod[i] - 1;

visited[i] = true;

cycle.push\_back(i + 1);

if (i == start) {

count+=cycle.size()-1;

for (size\_t n = 1; n < cycle.size(); n++)

{

printf("(%d,%d) ", cycle[0], cycle[n]);

}

cout << endl;

break;}}

cycle.clear();}}

if(!(count % 2)) { cout << "True: " << count << endl; return true; }

else { cout << "False: " << count << endl; return false; }}

bool ParityInversion(Podstanovka& Pod) {

short parity = 0;

for (short j = Pod.GetKolVo() - 1; j >= 0; j--)

for (short i = j; i >=0 ; i--)

if (Pod[j] < Pod[i]) parity++;

if (!(parity % 2)) { cout << "True: "<<parity<<endl; return true;}

else { cout << "False: "<<parity<<endl; return false; }}

int main() {

Podstanovka Pod("pods\_line.txt");

Pod.ToShow();

CyclesOfPodstanovka(Pod);

ParityInversion(Pod);

ParityTransposition(Pod);

return 228;}

**Результат программы:**

-------------------------

Count of elements: 6

1 2 3 4 5 6

3 6 4 1 2 5

-------------------------

Cycles: ( 3 4 1 );( 6 5 2 );

True: 8

Transposition: (3,4) (3,1)

(6,5) (6,2)

True: 4

**Задание 2:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **(+)mod10** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **0** | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* |
| **1** | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* |
| **2** | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* |
| **3** | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* |
| **4** | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* |
| **5** | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* |
| **6** | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* |
| **7** | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* |
| **8** | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
| **9** | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* |

|  |  |
| --- | --- |
| Обратные | |
| 0 | 0 |
| 1 | 9 |
| 2 | 8 |
| 3 | 7 |
| 4 | 6 |
| 5 | 5 |
| 3 | 4 |
| 7 | 3 |
| 8 | 2 |
| 9 | 1 |

Нейтральный элемент это e=0

**2.** Данная группа является циклической и её порождающий элемент это 1.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Элемент** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| **Порядок** | 1 | 10 | 5 | 10 | 5 | 2 | 5 | 10 | 5 | 10 |

**3.**<0>={0}; <5>={0,5};

<1>,<3>,<7>,<9> = {0,1,2,3,4,5,6,7,8,9};

<2>,<4>,<6>,<8>= {0,2,4,6,8};

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **(+)mod10** | **0** | **2** | **4** | **6** | **8** |
| **0** | *0* | *2* | *4* | *6* | *8* |
| **2** | *2* | *4* | *6* | *8* | *0* |
| **4** | *4* | *6* | *8* | *0* | *2* |
| **6** | *6* | *8* | *0* | *2* | *4* |
| **8** | *8* | *0* | *2* | *4* | *6* |

**4.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **gH** | **{0}** | **{0,1,2,3,4,5,6,7,8,9}** | **{0,2,4,6,8}** | **{0,5}** |
| **1** | {1} | {1,2,3,4,5,6,7,8,9,0} | {1,3,5,7,9} | {1,6} |
| **2** | {2} | {2,3,4,5,6,7,8,9,0,1} | {2,4,6,8,0} | {2,7} |
| **3** | {3} | {3,4,5,6,7,8,9,0,1,2} | {3,5,7,9,1} | {3,8} |
| **4** | {4} | {4,5,6,7,8,9,0,1,2,3} | {4,6,8,0,2} | {4,9} |
| **5** | {5} | {5,6,7,8,9,0,1,2,3,4} | {5,7,9,1,3} | {5,0} |
| **6** | {6} | {6,7,8,9,0,1,2,3,4,5} | {6,8,0,2,4} | {6,1} |
| **7** | {7} | {7,8,9,0,1,2,3,4,5,6} | {7,9,1,3,5} | {7,2} |
| **8** | {8} | {8,9,0,1,2,3,4,5,6,7} | {8,0,2,4,6} | {8,3} |
| **9** | {9} | {9,0,1,2,3,4,5,6,7,8} | {9,1,3,5,7} | {9,4} |

S1={{0},{1},{2},{3},{4},{5},{6},{7},{8},{9}}

S2={0,1,2,3,4,5,6,7,8,9}

S3={{0,2,4,6,8},{1,3,5,7,9}}

S4={{0,5},{1,6},{2,7},{3,8},{4,9}}

**5.**

**S1**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **(+)mod10** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| **0** | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* |
| **1** | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* |
| **2** | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* |
| **3** | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* |
| **4** | *4* | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* |
| **5** | *5* | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* |
| **6** | *6* | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* |
| **7** | *7* | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* |
| **8** | *8* | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* |
| **9** | *9* | *0* | *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* |

**S2**

|  |  |
| --- | --- |
| **(+)mod10** | **0,1,2,3,4,5,6,7,8,9** |
| **0,1,2,3,4,5,6,7,8,9** | 0,1,2,3,4,5,6,7,8,9 |

**S3**

|  |  |  |
| --- | --- | --- |
| **(+)mod10** | **0,2,4,6,8** | **1,3,5,7,9** |
| **0,2,4,6,8** | 0,2,4,6,8 | 1,3,5,7,9 |
| **1,3,5,7,9** | 1,3,5,7,9 | 0,2,4,6,8 |

**S4**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **(+)mod10** | **0,5** | **1,6** | **2,7** | **3,8** | **4,9** |
| **0,5** | 0,5 | 1,6 | 2,7 | 3,8 | 4,9 |
| **1,6** | 1,6 | 2,7 | 3,8 | 4,9 | 0,5 |
| **2,7** | 2,7 | 3,8 | 4,9 | 0,5 | 1,6 |
| **3,8** | 3,8 | 4,9 | 0,5 | 1,6 | 2,7 |
| **4,9** | 4,9 | 0,5 | 1,6 | 2,7 | 3,8 |

**Задание 3:**

Так как вторая группа задана неявно, то она будет иметь следующий вид: G`={{-1,1},’\*’};

Отображение групп G→G`  будет выглядеть так: f : четное → 1, нечетное → -1. Условие для проверки, что это действительно гомоморфизм: f(a + b) =f(a) \* f(b) для любых a,b принадлежащим G.

Доказательство:

Таблица Кели для второй группы:

|  |  |  |
| --- | --- | --- |
| **\*** | 1 | -1 |
| 1 | 1 | -1 |
| -1 | -1 | 1 |

Рассмотрим все комбинации четных и нечетных чисел:

f(2+4)=f(2)\*f(4) → 1=1\*1

f(2+1)=f(2)\*f(1) → -1=1\*-1

f(1+3)=f(1)\*f(3) → 1 = -1\*-1

Все комбинации удовлетворяют условиям, поэтому данное отображение является гомоморфизмом.

**Вывод:** Изучил и решил задания по подстановкам, группам.