Пермский национальный исследовательский политехнический университет

Кафедра информационные технологии и автоматизированные системы

Лабораторная работа

“**Ханойская башня**”

|  |  |
| --- | --- |
|  | Выполнил:  студент группы: ИВТ-24-2б  К.Д. Горбушин  Проверил:  доцент кафедры ИТАС  О.А. Полякова |

Пермь 2025

**Постановка задачи**

Задача состоит в том, чтобы перенести пирамиду из восьми колец за наименьшее число ходов на другой стержень. За один раз разрешается переносить только одно кольцо, причём нельзя класть большее кольцо на меньшее.

**Код программы**

#include <iostream>

#include <iomanip>

using namespace std;

// Количество дисков (константа)

const int NUM\_DISKS = 3;

// Функция для отображения башни

void displayTower(int disks[]) {

for (int i = NUM\_DISKS - 1; i >= 0; --i) { // Итерируем в обратном порядке

if (disks[i] == 0) {

cout << setw(NUM\_DISKS) << "|" << setw(NUM\_DISKS) << endl; // Пустое место

}

else {

int width = 2 \* disks[i] - 1; // Ширина диска

int padding = NUM\_DISKS - disks[i]; // Отступ для центровки

// Выводим пробелы для отступа

for (int j = 0; j < padding; ++j) {

cout << " ";

}

// Выводим диск

for (int j = 0; j < width; ++j) {

cout << "\*";

}

// Выводим пробелы для отступа справа

for (int j = 0; j < padding; ++j) {

cout << " ";

}

cout << endl;

}

}

}

// Функция для решения головоломки Ханойская башня с визуализацией

void hanoi(int n, int source, int destination, int auxiliary,

int tower1[], int tower2[], int tower3[]) {

if (n > 0) {

hanoi(n - 1, source, auxiliary, destination, tower1, tower2, tower3);

int diskToMove = 0; // Инициализируем значением по умолчанию

int sourceIndex = -1; // Инициализируем значением по умолчанию

int destIndex = -1; // Инициализируем значением по умолчанию

// Находим диск для перемещения в исходной башне

int\* sourceTower;

if (source == 1) sourceTower = tower1;

else if (source == 2) sourceTower = tower2;

else sourceTower = tower3;

for (int i = 0; i < NUM\_DISKS; ++i) {

if (sourceTower[i] == n) {

diskToMove = sourceTower[i];

sourceIndex = i;

sourceTower[i] = 0;

break;

}

}

// Находим место для диска в целевой башне

int\* destTower;

if (destination == 1) destTower = tower1;

else if (destination == 2) destTower = tower2;

else destTower = tower3;

for (int i = 0; i < NUM\_DISKS; ++i) {

if (destTower[i] == 0) {

destIndex = i;

destTower[i] = diskToMove;

break;

}

}

cout << "Перемещаем диск " << n << " с башни " << source << " на башню " << destination << endl;

cout << "Башня 1:\n";

displayTower(tower1);

cout << "Башня 2:\n";

displayTower(tower2);

cout << "Башня 3:\n";

displayTower(tower3);

hanoi(n - 1, auxiliary, destination, source, tower1, tower2, tower3);

}

}

int main() {

setlocale(LC\_ALL, "Ru");

// Инициализация башен

int tower1[NUM\_DISKS];

int tower2[NUM\_DISKS];

int tower3[NUM\_DISKS];

// Изначально все диски находятся на первой башне

for (int i = 0; i < NUM\_DISKS; ++i) {

tower1[i] = NUM\_DISKS - i; // Размеры дисков (от самого большого к самому маленькому)

tower2[i] = 0;

tower3[i] = 0;

}

cout << "Начальное состояние:\n";

cout << "Башня 1:\n";

displayTower(tower1);

cout << "Башня 2:\n";

displayTower(tower2);

cout << "Башня 3:\n";

displayTower(tower3);

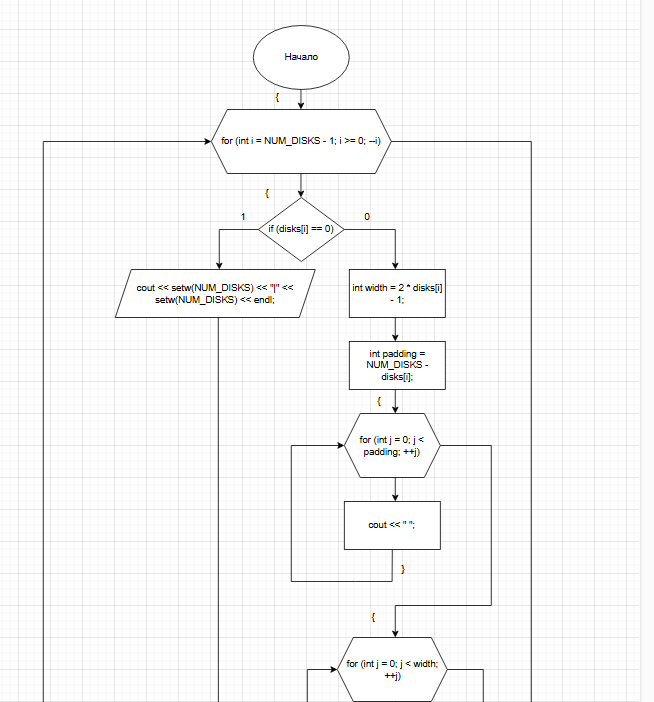
cout << "Решение для " << NUM\_DISKS << " дисков:\n";

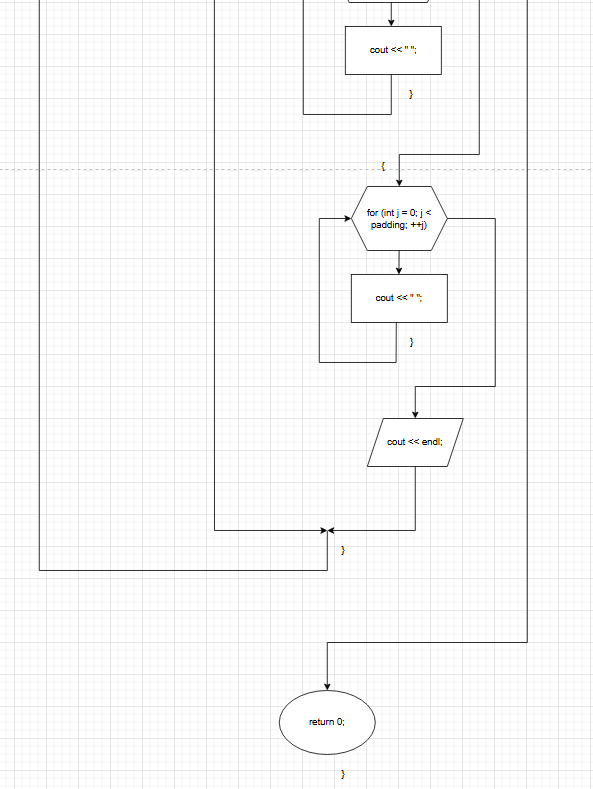
hanoi(NUM\_DISKS, 1, 3, 2, tower1, tower2, tower3);

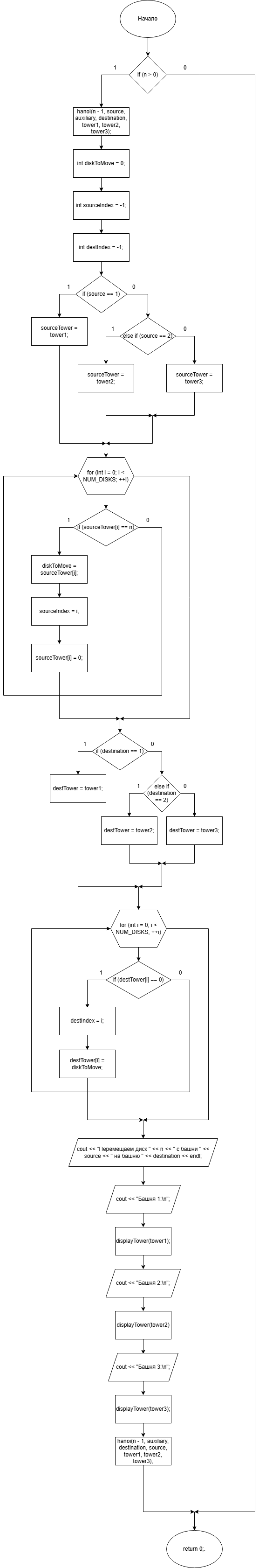
return 0;

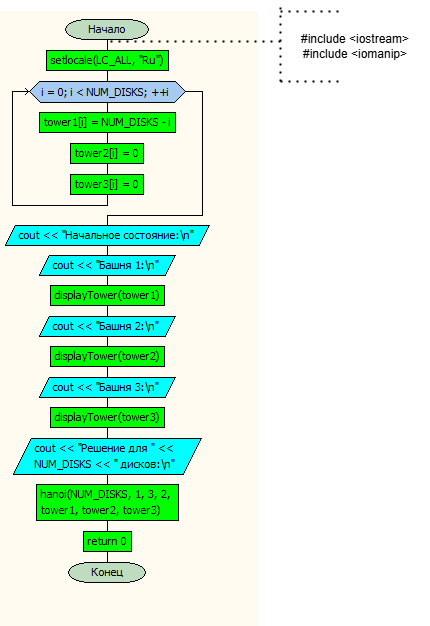
}

**Блок-схема**









**Ссылка на GITHUB:** [https://github.com/kirill2209](https://github.com/kirill2209%20)