Министерство науки и образования РФ

федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

|  |  |
| --- | --- |
| Факультет (институт) | *Информационных технологий и компьютерных систем* |
|  |  |
| Кафедра | *Информатики и вычислительной техники* |
|  |  |

Расчетно-графическая работа

|  |  |
| --- | --- |
| по дисциплине | *Операционные системы* |
|  |  |
| на тему | Разработка программы с использованием многопоточных программ в Linux |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Студента (ки) | | | | Шмидта Антона Владиславовича | | | | | |
|  |  |  |  | | | | фамилия, имя, отчество полностью | | | | | |
|  |  |  | Курс | *2* |  | | | Группа | | **ИВТ-244** | |  |
|  |  |  |  |  | |  | | |  |  | |  |
|  |  |  | Направление (специальность) | | | | | | | *09.03.01 –* | | |
|  |  |  | *Информатика и вычислительная техника* | | | | | | | | | |
|  |  |  | код, наименование | | | | | | | | | |
|  |  |  | Проверил | | | | *ассистент* | | | | | |
|  |  |  |  | | | | ученая степень, звание | | | | | |
|  |  |  | *Кононова В.В.* | | | | | | | | | |
|  |  |  | фамилия, инициалы | | | | | | | | | |
|  |  |  | Выполнил (а) | | | |  | | | | | |
|  |  |  |  | | | | дата, подпись студента (ки) | | | | | |
|  |  |  |  | | | | | | | | | |
|  |  |  | Работа защищен (а) с количеством баллов | | | | | | | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | дата, подпись руководителя |  |  |  |

Омск 2025

**Цель работы**:

Изучение средств и методов использования многопоточных программ в Linux.

**Задание**:

Разработать программу с тремя дополнительными нитями (threads) относительно главной нити. Каждая из нитей должна использовать общие для всех нитей данные, представленные массивом символов, в которых записаны 20 первых букв латинского алфавита. Каждая из этих нитей на своем k-м шаге выводит со своей случайной задержкой на место «своего» столбца экрана k-ю букву из указанного массива латинских букв, причем с числом повторений, равному условному номеру нити, умноженному на два. Каждая из используемых нитей должен осуществлять вывод своим цветом, отличным от остальных нитей. На 6-м шаге главная нить делает попытку отмены первой из дополнительных нитей, а на 11-м делает попытку отмены третьей из дополнительных нитей. Первая и третья дополнительная нити в начале своей работы запрещают свою отмену. Третья нить на 13 шаге разрешает отмену, но в отложенном режиме. Точку отмены эта нить устанавливает между 16 и 17-м шагом своей работы. Все управляющие указания должны отображаться сообщениями без прокрутки экрана (в фиксированные позиции экрана).

**Исходный код:**

#include <iostream>

#include <chrono>

#include <random>

#include <string>

#include <unistd.h>

#include <pthread.h>

using namespace std;

using namespace std::chrono;

const string ALPHABET = "ABCDEFGHIJKLMNOPQRST";

const string RED = "\033[31m";

const string GREEN = "\033[32m";

const string CYAN = "\033[36m";

const string YELLOW = "\033[33m";

const string BOLD = "\033[1m";

const string RESET = "\033[0m";

const int COL1 = 15, COL2 = 48, COL3 = 81; // расположение столбцов

// переход в координату

void gotoxy(int y, int x) {

cout << "\033[" << y << ";" << x << "H";

}

// очищение строки

void clear\_line(int row) {

gotoxy(row, 1);

cout << "\033[2K";

}

// вывод текста

void print(int row, int col, const string& text, const string& color = "") {

gotoxy(row, col);

cout << color << text << RESET << flush;

}

// функция первого потока

void\* thread1(void\*) {

pthread\_setcancelstate(PTHREAD\_CANCEL\_DISABLE, nullptr);

clear\_line(23); print(23, 0, "Нить 1: отмена запрещена (на всё время работы)", RED);

random\_device rd; mt19937 gen(rd());

uniform\_int\_distribution<int> d(100, 900);

for (int k = 1; k <= 20; ++k) {

usleep(1000 \* d(gen));

char letter = ALPHABET[k-1];

print(k + 1, COL1, string(2, letter), RED); // 2 раза

}

clear\_line(23); print(23, 0, "Нить 1: завершена нормально (неотменяемая)", RED);

return nullptr;

}

// функция второго потока

void\* thread2(void\*) {

clear\_line(24); print(24, 0, "Нить 2: отмена разрешена по умолчанию", GREEN);

random\_device rd; mt19937 gen(rd());

uniform\_int\_distribution<int> d(120, 950);

for (int k = 1; k <= 20; ++k) {

usleep(1000 \* d(gen));

char letter = ALPHABET[k-1];

print(k + 1, COL2, string(4, letter), GREEN); // 4 раза

}

clear\_line(24); print(24, 0, "Нить 2: завершена нормально", GREEN);

return nullptr;

}

// функция третьего потока

void\* thread3(void\*) {

pthread\_setcancelstate(PTHREAD\_CANCEL\_DISABLE, nullptr);

clear\_line(25); print(25, 0, "Нить 3: отмена запрещена в начале работы", CYAN);

random\_device rd; mt19937 gen(rd());

uniform\_int\_distribution<int> d(100, 880);

for (int k = 1; k <= 20; ++k) {

usleep(1000 \* d(gen));

char letter = ALPHABET[k-1];

print(k + 1, COL3, string(6, letter), CYAN);

if (k == 13) {

pthread\_setcanceltype(PTHREAD\_CANCEL\_DEFERRED, nullptr);

pthread\_setcancelstate(PTHREAD\_CANCEL\_ENABLE, nullptr);

clear\_line(25); print(25, 0, "Нить 3: на 13-м шаге разрешила отмену (отложенный режим)", CYAN);

}

if (k == 16) {

clear\_line(25); print(25, 0, "Нить 3: точка отмены между 16 и 17 шагом - pthread\_testcancel()", CYAN);

pthread\_testcancel();

}

}

return nullptr;

}

int main() {

system("clear");

print(1, COL1, "Нить 1", RED);

print(1, COL2, "Нить 2", GREEN);

print(1, COL3, "Нить 3", CYAN);

pthread\_t tid1, tid2, tid3;

pthread\_create(&tid1, nullptr, thread1, nullptr);

pthread\_create(&tid2, nullptr, thread2, nullptr);

pthread\_create(&tid3, nullptr, thread3, nullptr);

random\_device rd; mt19937 gen(rd());

uniform\_int\_distribution<int> main\_d(400, 1100);

for (int step = 1; step <= 20; ++step) {

usleep(1000 \* main\_d(gen));

if (step == 6) {

pthread\_cancel(tid1);;

clear\_line(26); print(26, 0, "Главная нить: 6-й шаг - пытается отменить Нить 1 - ОТКАЗАНО", YELLOW);

}

if (step == 11) {

pthread\_cancel(tid3);

clear\_line(26); print(26, 0, "Главная нить: 11-й шаг - пытается отменить Нить 3 - запрос принят", YELLOW);

}

}

pthread\_join(tid1, nullptr);

pthread\_join(tid2, nullptr);

pthread\_join(tid3, nullptr);

print(28, 0, "Нажмите Enter для выхода...", BOLD);

cin.get();

cout << RESET;

return 0;

}

**Результат выполнения:**

На рисунке 1 представлен вывод программы после запуска. Нити 1, 2 и 3 начали своё выполнение.

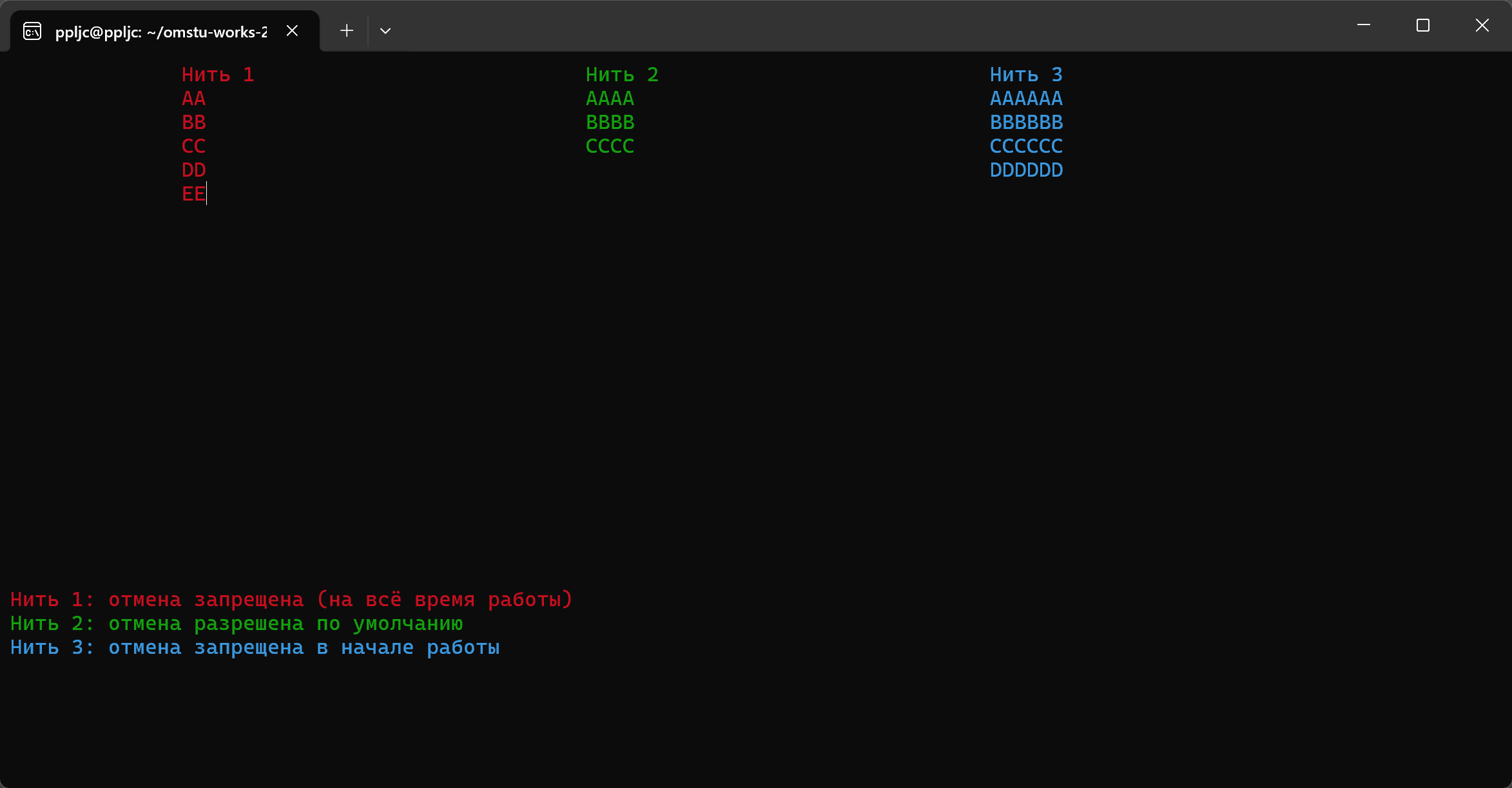


Рисунок 1 – Вывод программы в начале выполнения.

Рисунок 2 показывает, что главная нить пытается отменить первую нить, но та является неотменяемой и поэтому продолжает своё выполнение. В то же время третья нить на 13-м шаге разрешает свою отмену.

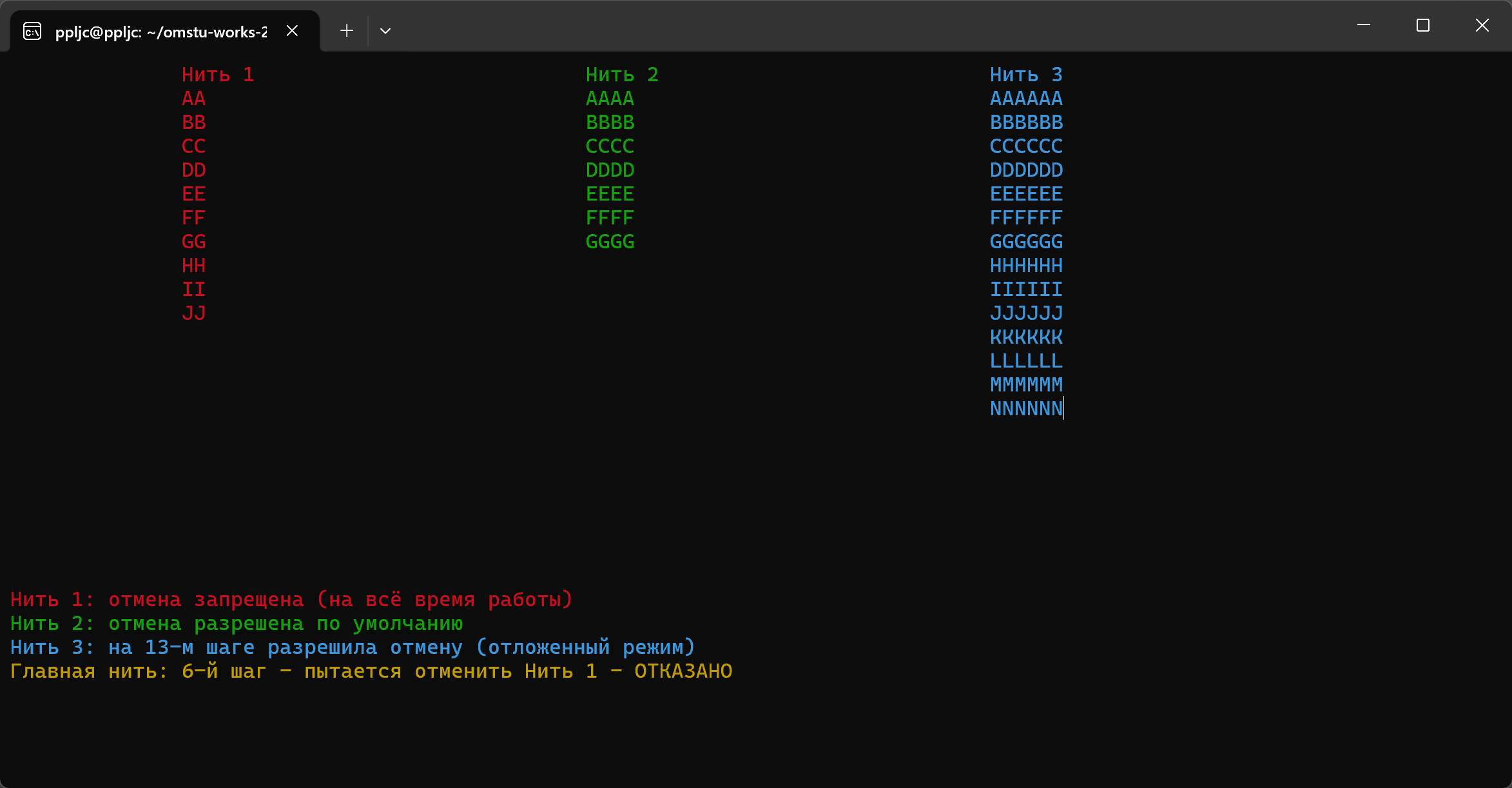


Рисунок 2 – Вывод программы после отдачи команд на отмену первой нити и на разрешение отмены третьей нити.

На рисунке 3 видно, что третья нить устанавливает свою точку отмену между 16-м и 17-м шагом.

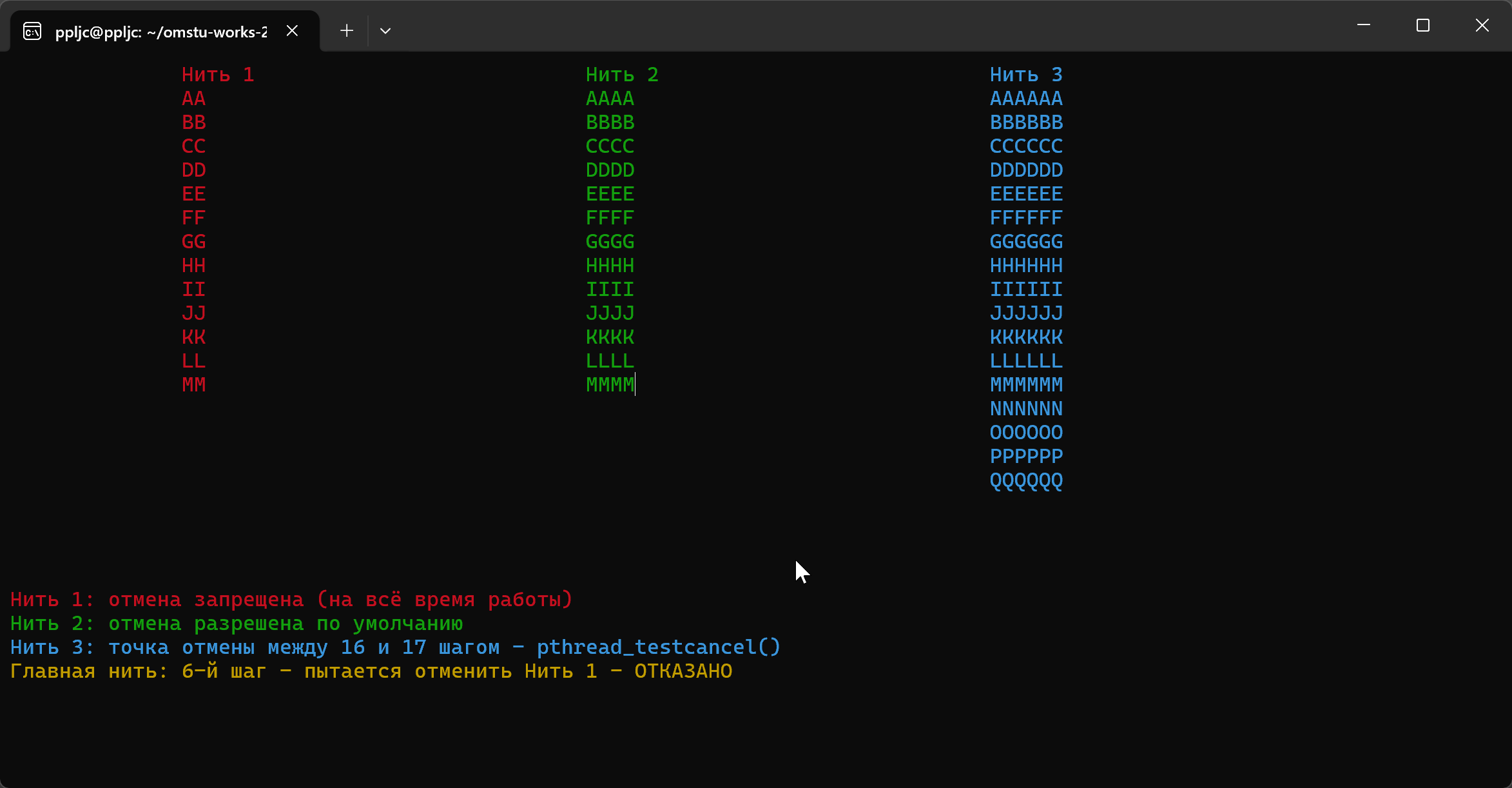


Рисунок 3 – Вывод программы после установки точки отмены для третьей нити.

По рисунку 4 видно, что главная нить пытается отменить третью нить и успешно останавливает её выполнение, так как ранее третья нить установила точку отмены.



Рисунок 4 – Вывод программы после новой попытки главной нити отменить третью.

Рисунок 5 показывает финальное состояние программы, на котором первая и вторая нить завершили своё выполнение, а выполнение третьей нити было отменено главной нитью.

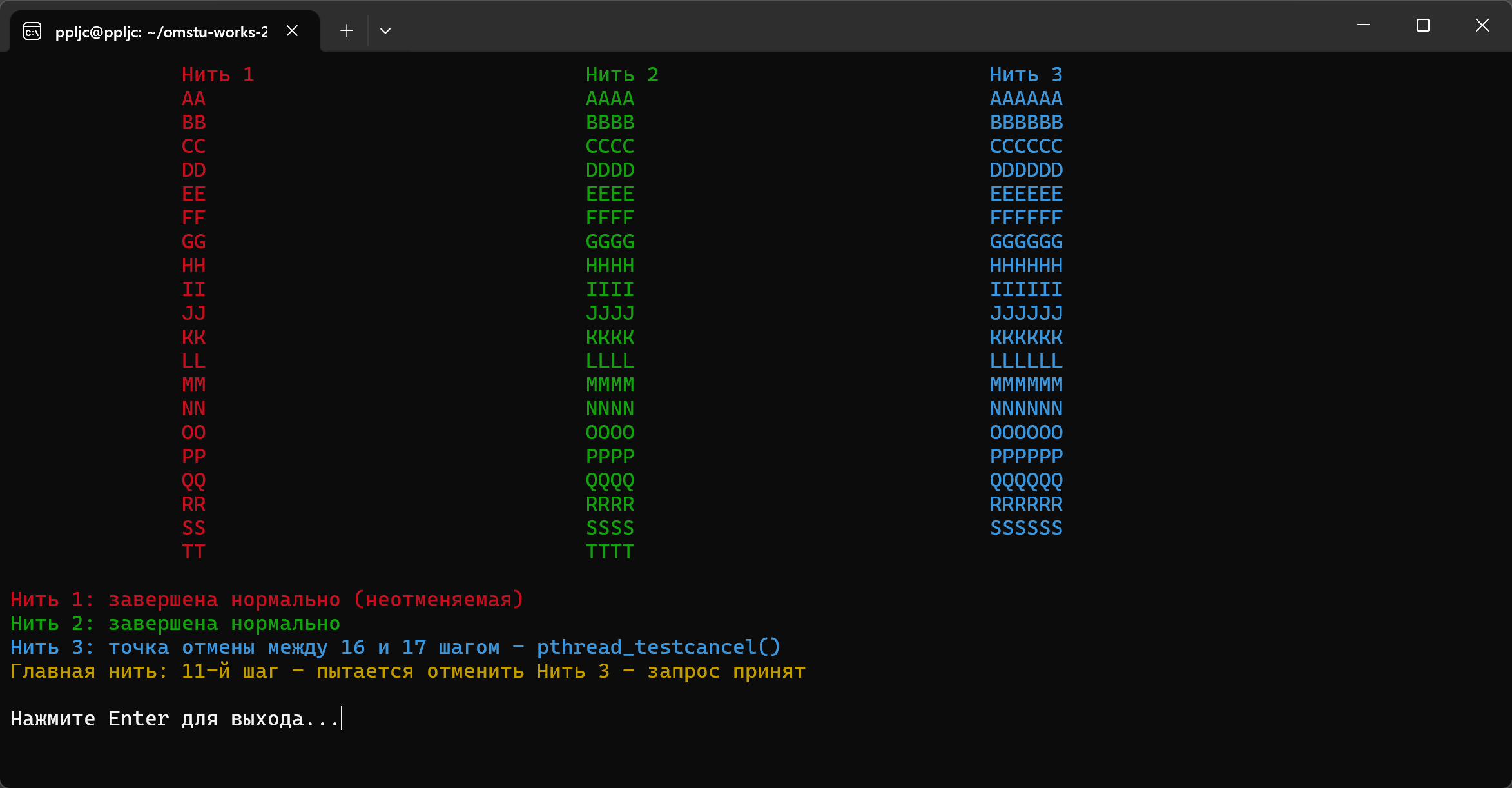


Рисунок 5 – Вывод программы в конце.

**Вывод**:

В ходе работы было изучено создание и управление многопоточными программами в Linux, включая работу с общими данными, синхронизацию потоков и организацию цветного вывода на экран. Была разработана программа с тремя дополнительными нитями, реализованы механизмы отмены потоков, включая запрет и отложенную отмену, что позволило на практике закрепить навыки управления и контроля многопоточных процессов.