

МИНОБРАЗОВАНИЯ РОССИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
(ФГАОУ ВО «СПбПУ»)
Университетский политехнический колледж

Утверждаю
Зам. директора по УМР
Е.Г.Конакина
«___» ____ 2017 г.

ОТЧЕТ
по учебной практике УП.01.01
«Системное программирование»

По профессиональному модулю ПМ.01 «Разработка программных модулей
программного обеспечения для компьютерных систем»

Специальность 09.02.03. «Программирование в компьютерных системах»

Студента III курса ____ 32928/2 ____ группы

Тимушев Фёдор Алексеевич
(Фамилия, имя, отчество)

Место прохождения практики: УПК, пр. Энгельса д.23

Период прохождения практики
с «25» сентября 2017 г. по «14» октября 2017 г.

Руководитель(и) практики:

(подпись)

Девятко Н.С.
(расшифровка подписи)

Итоговая оценка по практике _____

Санкт-Петербург
2017

Утверждаю
Зам. директора по УМР
_____ Е.Г.Конакина
«___» _____ 2017 г.

Задание на учебную практику УП.01.01

по профессиональному модулю

ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем»

Специальность 09.02.03 «Программирование в компьютерных системах»
(код и наименование специальности)

Студенту 3 курса 32928/2 группы

Тимушев Фёдор Алексеевич

(фамилия, имя, отчество)

Место прохождения практики Университетский политехнический колледж
(наименование и адрес организации)

Период прохождения практики
с «25» сентября 2017 г. по «14» октября 2017 г.

Виды работ, обязательные для выполнения

Использование функций
Работа с указателями
Работа с файлами
Динамические структуры данных
Работа с файловой системой
Организация многопоточной обработки данных

Индивидуальное задание *(заполняется в случае необходимости дополнительных видов работ для решения практикоориентированных задач и т.д.)*

ВАРИАНТ 8

Задание выдал «25» сентября 2017 г.

(подпись)

Девятко Н.С

(Ф.И.О.)

С заданием ознакомлен

«25» сентября 2017 г.

(подпись)

(Ф.И.О. студента)

МИНОБРАЗОВАНИЯ РОССИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
(ФГАОУ ВО «СПбПУ»)
Университетский политехнический колледж

ДНЕВНИК
прохождения учебной практики
«Системное программирование»

По профессиональному модулю ПМ.01 «Разработка программных модулей
программного обеспечения для компьютерных систем»

Специальность 09.02.03. «Программирование в компьютерных системах»

Студент(ка) III курса 32928/2 группы

Тимушев Фёдор Алексеевич

Место прохождения практики УПК, пр. Энгельса д.23

Период прохождения практики
с «25» сентября 2017 г. по «14» октября 2017 г.

Руководитель(и) практики:

(подпись)

Девятко Н.С.
(расшифровка подписи)

Итоговая оценка по практике _____

2017

Санкт-Петербург

Содержание дневника

Дата	Виды выполненных работ и заданий по программе практики	Подпись руководителя практики
1	2	3
25.09	Разработка алгоритмов решения вычислительных задач.	
26.09	Реализация программ по типовым алгоритмам.	
27.09	Разработка спецификаций отдельных компонент программного обеспечения. Создание и использование функций.	
28.09	Кодирование, отладка, тестирование, оптимизация программного модуля обработки простых данных с помощью функций программиста.	
29.09	Организация динамического распределения памяти. Разработка функций для обработки, преобразования, сортировки данных с помощью адресной арифметики.	
30.09	Кодирование, отладка, тестирование, оптимизация программного модуля обработки массивов с использованием указателей.	
02.10	Разработка спецификаций функций анализа и обработки строчных величин. Создание статической библиотеки.	
03.10	Кодирование, отладка, тестирование, оптимизация программного модуля обработки строк с использованием собственной библиотеки.	
04.10	Разработка функций для работы с текстовыми файлами, организации ввода-вывода текстовой информации и ее хранения на внешних носителях.	
05.10	Разработка программ для работы с файлами и папками.	
06.10	Разработка программ обработки двоичных файлов.	
07.10	Динамические структуры данных: односвязная очередь, стек, дерево. Разработка функций по организации работы с динамическими структурами данных.	
09.10	Кодирование, отладка, тестирование, оптимизация программного модуля обработки динамических структур данных.	
10.10	Структура каталога. Разработка программ для просмотра каталога диска, поиска файлов, определения и изменения атрибутов файлов.	
11.10	Кодирование, отладка, тестирование, оптимизация программного модуля по работе с файловой системой.	
12.10	Создание параллельных потоков и процессов обработки информации.	
13.10	Синхронизация потоков и процессов.	
14.10	Кодирование, отладка, тестирование, оптимизация программного модуля многопоточной обработки информации.	

Тема 1: «Разработка программ по типовым алгоритмам»

Цель: получить практические навыки по разработке программ по типовым алгоритмам.

Задача 1.1.

Составить *алгоритм* и программу, которая по номеру дня в году выводит число и месяц в общепринятой форме. Например, 33-ий день в году – 2 февраля.

Входные данные:

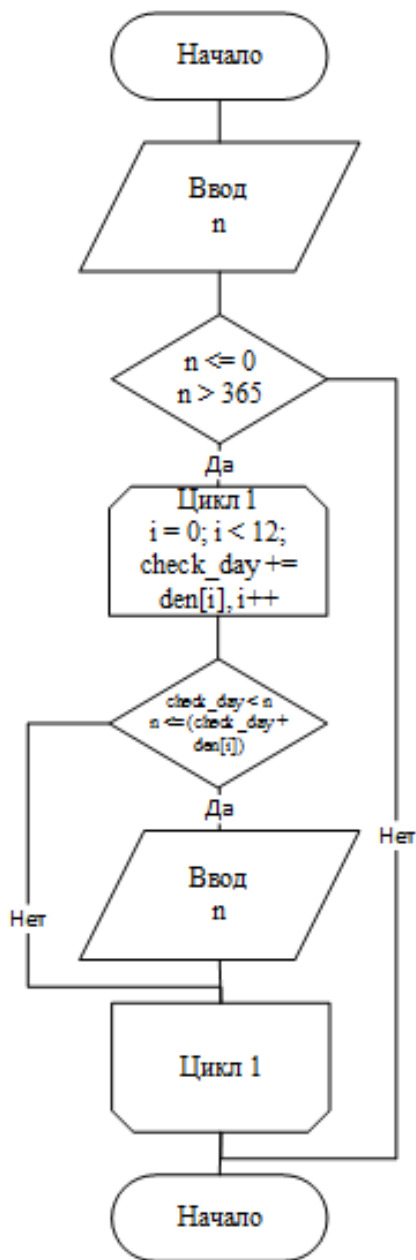
const string mes[12] – *строковый массив, содержащий названия месяцев года;*
const int den[12] – *числовой массив, содержащий числа с кол-вом дней в месяцах;*
int n – *переменная целого типа, означающая номер дня в году, вводится с клавиатуры;*
int check_day – *переменная целого типа, означающая номер дня в месяце, с которым она будет сопоставляться;*

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры;
- 2 - вычисление дня в году, в соответствии с введенными пользователем данными;
- 3 - вывод найденного дня в общепринятой форме на экран;

Выходные данные:

n – *вывод дня, введенного пользователем ранее;*
(n – check_day) – *вывод искомого дня;*
mes[i] – *вывод сопоставимого искомому дню месяца;*



```

#include <conio.h>
#include <stdio.h>
#include <iostream>
#include <string>
using namespace std;
int main()
{
    setlocale(LC_ALL, "Russian");
    const string mes[12] = { "января", "февраля", "марта",
    "апреля", "мая", "июня", "июля", "августа", "сентября", "октября",
    "ноября", "декабря" }; //строковый массив, содержащий названия
    месяцев года
    const int den[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31,
    30, 31}; //числовой массив, содержащий числа с кол-вом дней в
    месяцах
    int n; //номер дня в году
    cout << "Введите номер дня в году от 1 до 365: ";
    cin >> n;
    int check_day = 0; //номер дня в месяце
    if (n <= 0 || n > 365) //проверка на правильность ввода номера
    дня
    {
        cout << "Введите день в правильном диапазоне!";
        system("Pause");
    }
    for (int i = 0; i < 12; check_day += den[i], i++)
    // "наращивание" месяцев к номеру дня в них
    {
        if (check_day < n && n <= (check_day + den[i])) //если
        номер дня в месяце меньше введенного пользователем и меньше
        "наращенного" то переход к выводу
        {
            cout << n << " день в году - " << n - check_day <<
            " " << mes[i]; //вывод
            break;
        }
    }
    _getch();
}

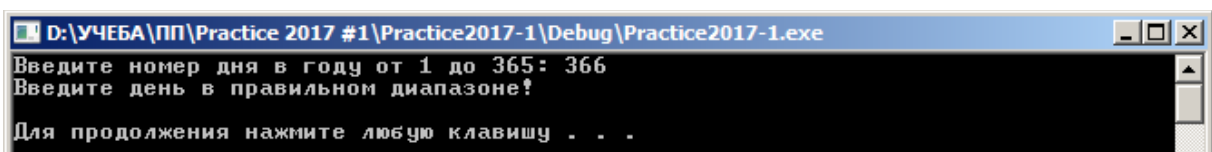
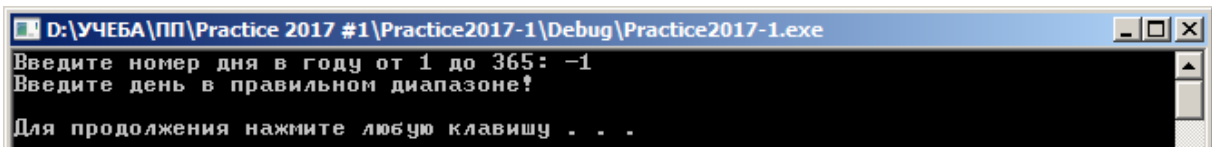
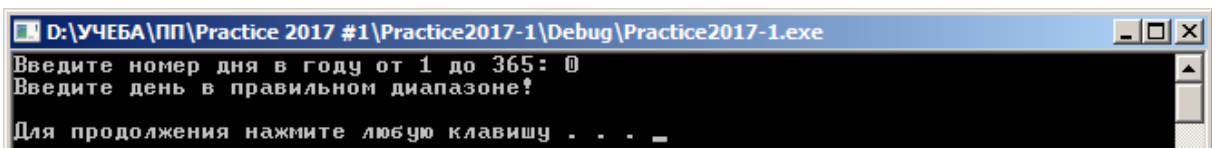
```

План тестирования:

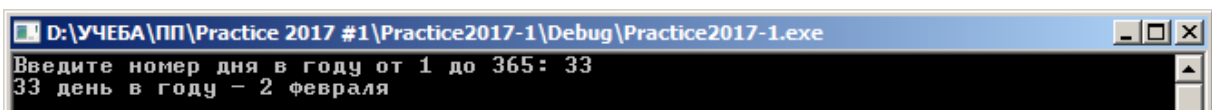
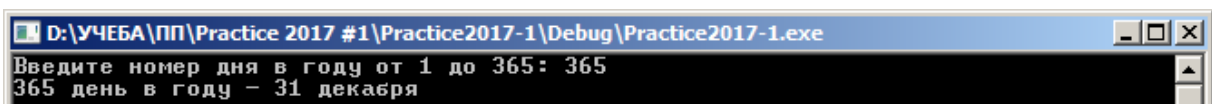
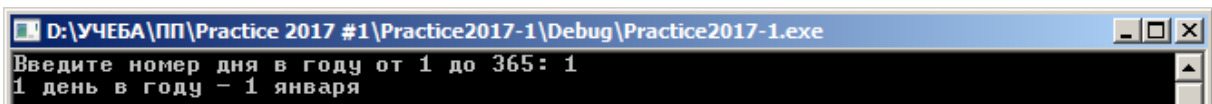
№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in [1;365], x > 0$	$n=33$ Ожидаемые ответы 2 февраля
2	Класс допустимых значений $n \in [1;365], x > 0$ <i>Граничные условия</i>	$n=1$ $n=365$ Ожидаемые ответы: 1 января, 31 декабря
3	Класс недопустимых значений $n \in (-\infty;0], x > 0$	$n=-1$ $n=0$ Ожидаемые ответы: сообщение об ошибке
4	Класс недопустимых значений $n \in [366;+\infty), x > 0$	$n=366$ Ожидаемые ответы: сообщение об ошибке

Результаты тестирования:

1 – Проверка на ввод значений вне диапазона:



2 – Проверка на ввод крайних значений (минимального и максимального):



Задача 1.2.

Составить *алгоритм* и программу, которая имитирует работу секундомера. Пользователь задает количество минут и секунд, затем программа показывает обратный отсчёт времени на чистом экране по центру, по окончании отсчёта выдаёт сообщение «Время истекло!»

Входные данные:

int m, s; - *минуты и секунды;*

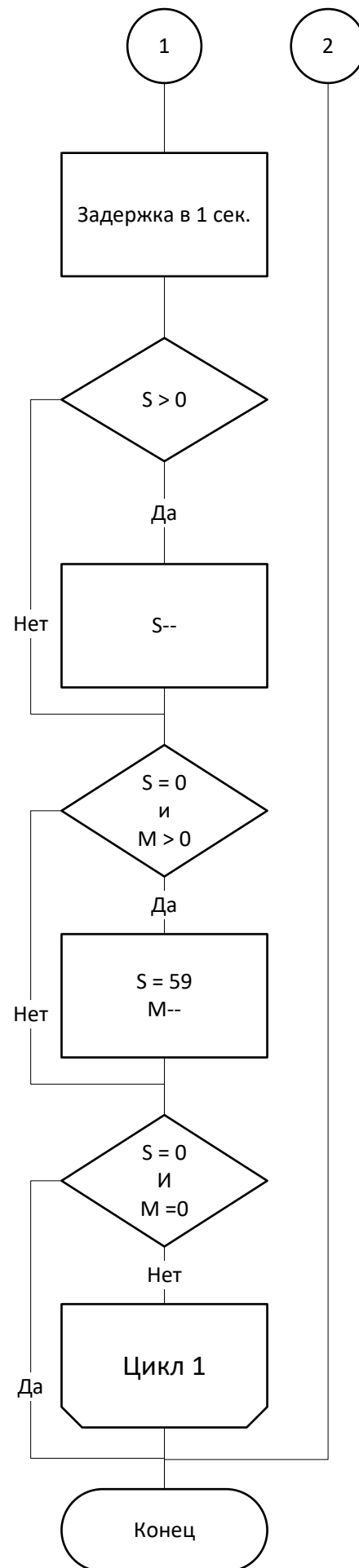
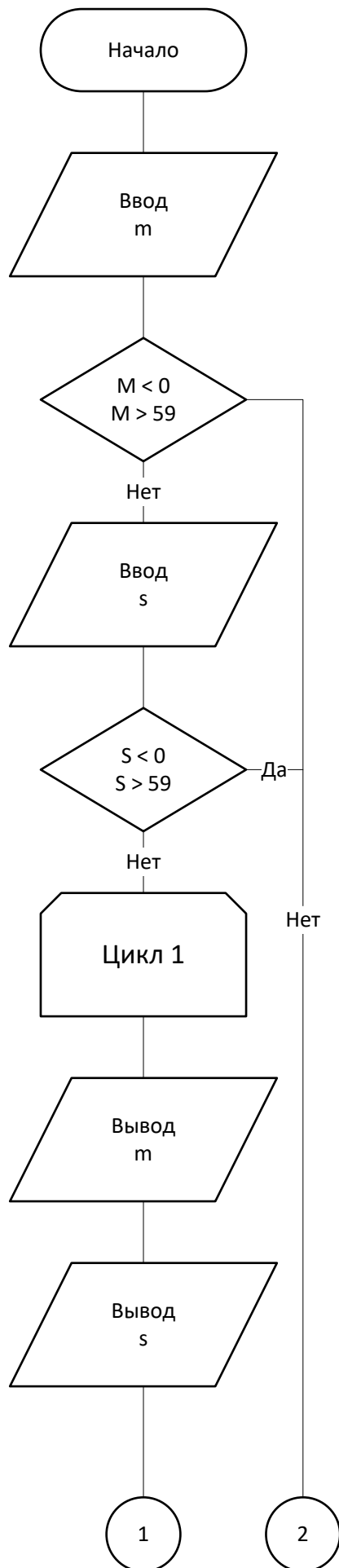
Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры;
- 2 – начало счета таймера;
- 3 - вывод сообщения об окончании времени;

Выходные данные:

int m, s; - *минуты и секунды;*

Сообщение об окончании счета;



```

#include <conio.h>
#include <stdio.h>
#include <iostream>
#include <string>
#include <windows.h>
#include <time.h>
using namespace std;
int main()
{
    setlocale(LC_ALL, "Russian");
    int m, s; // минуты и секунды
    cout << "Введите минуты: ";
    cin >> m;
    if (m < 0 || m > 59) //проверка на корректность ввода минут
    {
        cout << "Введите минуты корректно!";
        _getch();
        exit(0);
    }
    cout << "Введите секунды: ";
    cin >> s;
    if (s < 0 || s > 59) //проверка на корректность ввода секунд
    {
        cout << "Введите секунды корректно!";
        _getch();
        exit(0);
    }
    while (true)
    {
        system("CLS");
        HANDLE hd = GetStdHandle(STD_OUTPUT_HANDLE); //вывод по центру консоли
        COORD cd;
        cd.X = 35;
        cd.Y = 10;
        SetConsoleCursorPosition(hd, cd);

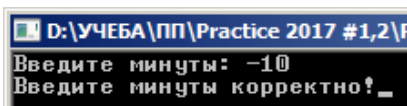
        cout.width(2); //заполнение таймера нулями
        cout.fill('0');
        cout << m << ":";
        cout.width(2);
        cout.fill('0');
        cout << s;
        Sleep(1000); //шаг в одну секунду
        if (s > 0)
            s--;
        if (s == 0 && m > 0)
        {
            s = 59;
            m--;
        }
        if (s == 0 && m == 0) //конец таймера
        {
            HANDLE hd = GetStdHandle(STD_OUTPUT_HANDLE); //вывод по центру
консоли сообщения
            COORD cd;
            cd.X = 30;
            cd.Y = 10;
            SetConsoleCursorPosition(hd, cd);
            cout << "Время истекло!" << endl;
            break;
        }
    }
    system("Pause");
}

```

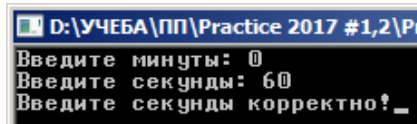
План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $m \in [0;59], m > 0$ $s \in [0;59], s > 0$	$m = 1$ $s = 5$ Ожидаемые ответы: окончание работы таймера выводом сообщения по центру консоли
2	Класс допустимых значений $m \in (-\infty; -1] [60; +\infty), m > 0$ $s \in (-\infty; -1] [60; +\infty), s > 0$	$n = -1$ $n = 60$ Ожидаемые ответы: сообщение об ошибке

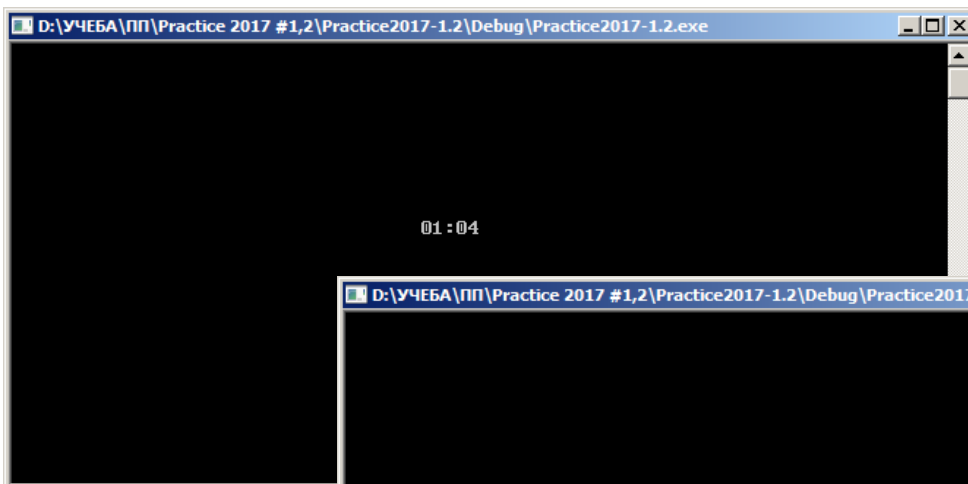
Результаты тестирования:



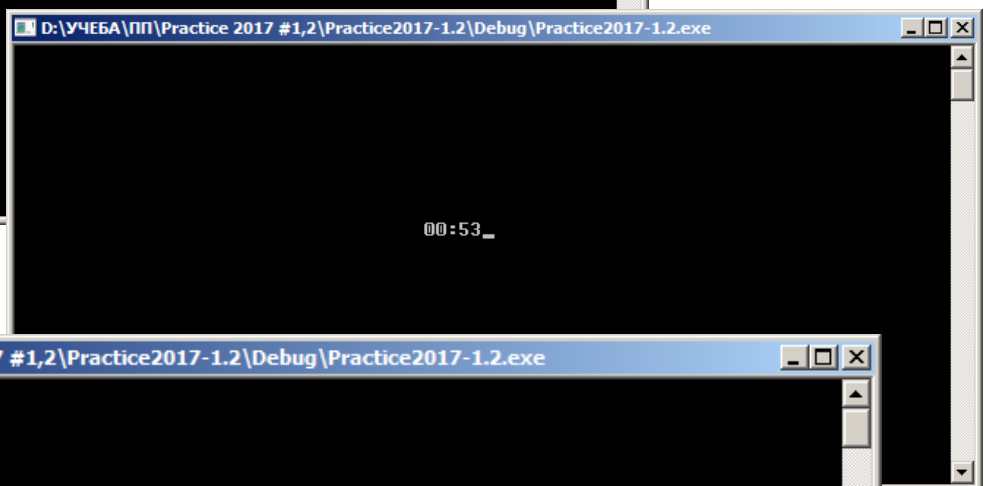
```
D:\УЧЕБА\ПП\Practice 2017 #1,2\Pr
Введите минуты: -10
Введите минуты корректно!_
```



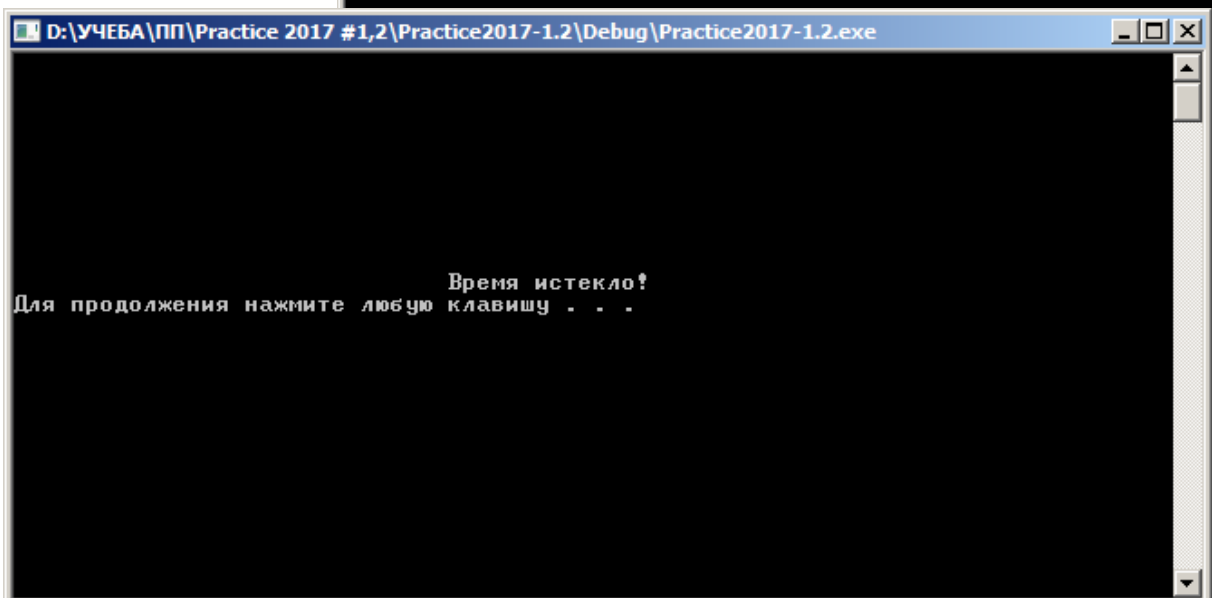
```
D:\УЧЕБА\ПП\Practice 2017 #1,2\Pr
Введите минуты: 0
Введите секунды: 60
Введите секунды корректно!_
```



```
D:\УЧЕБА\ПП\Practice 2017 #1,2\Practice2017-1.2\Debug\Practice2017-1.2.exe
01:04
```



```
D:\УЧЕБА\ПП\Practice 2017 #1,2\Practice2017-1.2\Debug\Practice2017-1.2.exe
00:53_
```



```
D:\УЧЕБА\ПП\Practice 2017 #1,2\Practice2017-1.2\Debug\Practice2017-1.2.exe
Время истекло!
Для продолжения нажмите любую клавишу . . .
```

Тема 2: «Использование функций»

Цель: получить практические навыки по использованию функций;

Задача 2.1:

Разработать спецификации и написать функцию для вычисления значения любого члена арифметической прогрессии по заданному номеру, если известен первый член прогрессии и её разность, а также функцию, проверяющую, принадлежит ли заданное число этой прогрессии. Организовать вызов обеих функций для проверки.

Входные данные:

double a, r, an - *a* - первый член, *r* - разность, *an* - *n*-ный член;

int n - Введите номер члена арифм. прогрессии;

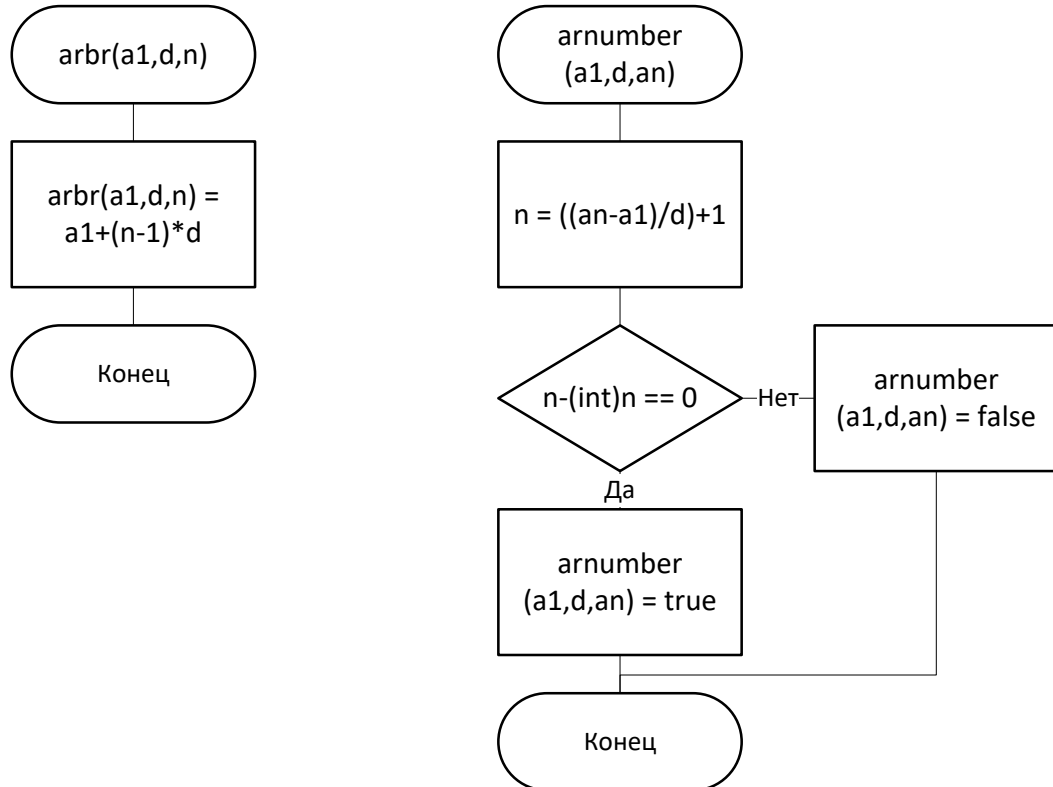
Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры;
- 2 – определение *n*-ного члена арифм. прогрессии;
- 3 – установление принадлежности числа к арифм. прогрессии;
- 3 – вывод вычислений на экран;

Выходные данные:

double an - *an* - *n*-ный член;

bool check – проверка на принадлежность к прогрессии;



```

#include <iostream>
#include <conio.h>
#include <locale>
#include <Windows.h>
using namespace std;
double arpr(double a, double r, int n)//функция вычисления члена арифм. прогрессии по
заданному номеру и возвращает an - n-ый член прогрессии
{return a + (n - 1)*r;}//a - первый член, r - разность, n - номер члена
bool arnumber(double a, double r, double an)//функция проверяет принадлежность числа к
арифметической прогрессии
{
    double n;
    n = ((an - a) / r) + 1;//a - первый член, r - разность, an - n-ый член
    if (n - (int)n == 0)//true - если принадлежит, false - не принадлежит
        return true;
    else
        return false;
}
void main()
{
    setlocale(LC_ALL, "Rus");
    double a, r, an;
    int n;
    bool check;
    cout << "--Опр. n-члена арифм. прогрессии--" << endl;
    cout << "\nВведите первый член арифм. прогрессии: "; cin >> a;
    cout << "\nВведите разность: "; cin >> r;
    do
    {cout << "\nВведите номер члена арифм. прогрессии: "; cin >> n;}
    while (n <= 0);
    an = arpr(a, r, n);
    cout << "\n" << n << "-ый член арифм. прогрессии равен = " << an;
    cout << "\n\nВведите n-ый член арифм. прогрессии: "; cin >> an;
    check = arnumber(a, r, an);
    if (check)
        cout << "\nЧисло принадлежит арифм. прогрессии";
    else
        cout << "\nЧисло не принадлежит арифм. прогрессии";
    _getch();
}

```

План тестирования:

- функции “arpr”

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in [0; 2147483647], n \in \mathbb{Z}$ a – любое число, r – любое число	$n=3$ $a = 2$ $r = 5$ Ожидаемые ответы: 3ий член ариф.прогрессии = 12 <i>// функция высчитывает n-ый член арифметической прогрессии</i>
2	Класс недопустимых значений $n < 0, n \in \mathbb{R}$	$n=-1$ $n=0$ Ожидаемые ответы: повторный ввод

F:\УЧЕБА\ПП\Practice 2017 #2\Practice2017-2\Debug\Prac

C:\Windows\system32\cmd.exe

--Опр. n-члена арифм. прогрессии--
Введите первый член арифм. прогрессии: 2
Введите разность: 5
Введите номер члена арифм. прогрессии: -1
Введите номер члена арифм. прогрессии: 0
Введите номер члена арифм. прогрессии: 3

--Опр. n-члена арифм. прогрессии--
Введите первый член арифм. прогрессии: 2
Введите разность: 5
Введите номер члена арифм. прогрессии: 3
3-ый член арифм. прогрессии равен = 12
Введите n-ный член арифм. прогрессии: 12
Число принадлежит арифм. прогрессии_

- функции “arnumber”

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений a – любое число, r – любое число, an – любое число	an=12 a = 2 r = 5 Ожидаемые ответы: Число принадлежит арифметической прогрессии Функция проверяет принадлежность члена an к арифметической прогрессии

F:\УЧЕБА\ПП\Practice 2017 #2\Practice2017-2\Debug\Prac

--Опр. n-члена арифм. прогрессии--
Введите первый член арифм. прогрессии: 2
Введите разность: 5
Введите номер члена арифм. прогрессии: -1
Введите номер члена арифм. прогрессии: 0
Введите номер члена арифм. прогрессии: 3
3-ый член арифм. прогрессии равен = 12
Введите n-ный член арифм. прогрессии: 12
Число принадлежит арифм. прогрессии

Задача 2.2:

Разработать программу для вычисления интеграла методом трапеций и методом Симпсона, оформив каждый способ в виде отдельной функции. Вывести на экран результаты интегрирования разными методами для сравнения.

$$I = \int_0^{\frac{\pi}{2}} \frac{\sin x \cdot \cos x}{(2 + \sin^2 x - \cos^2 x)} dx$$

Входные данные:

double accuracy – точность;

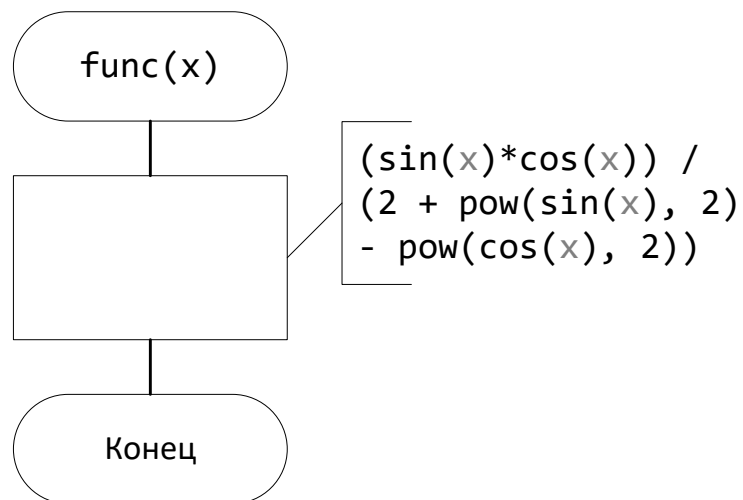
Функциональные характеристики:

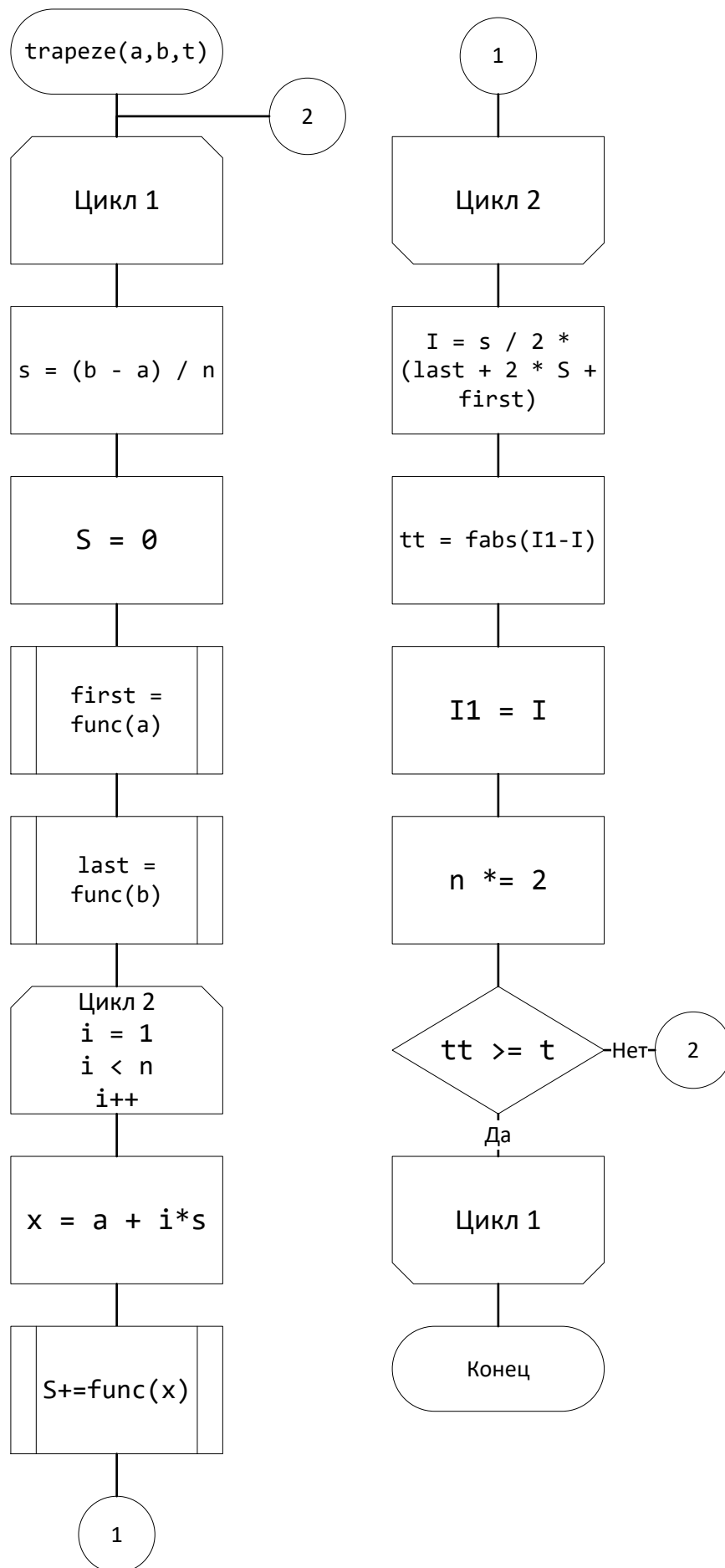
- 1 - ввод данных пользователем с клавиатуры;
- 2 – вычисление интеграла методом трапеций;
- 3 – вычисление интеграла методом Симпсона;
- 3 – вывод вычислений на экран;

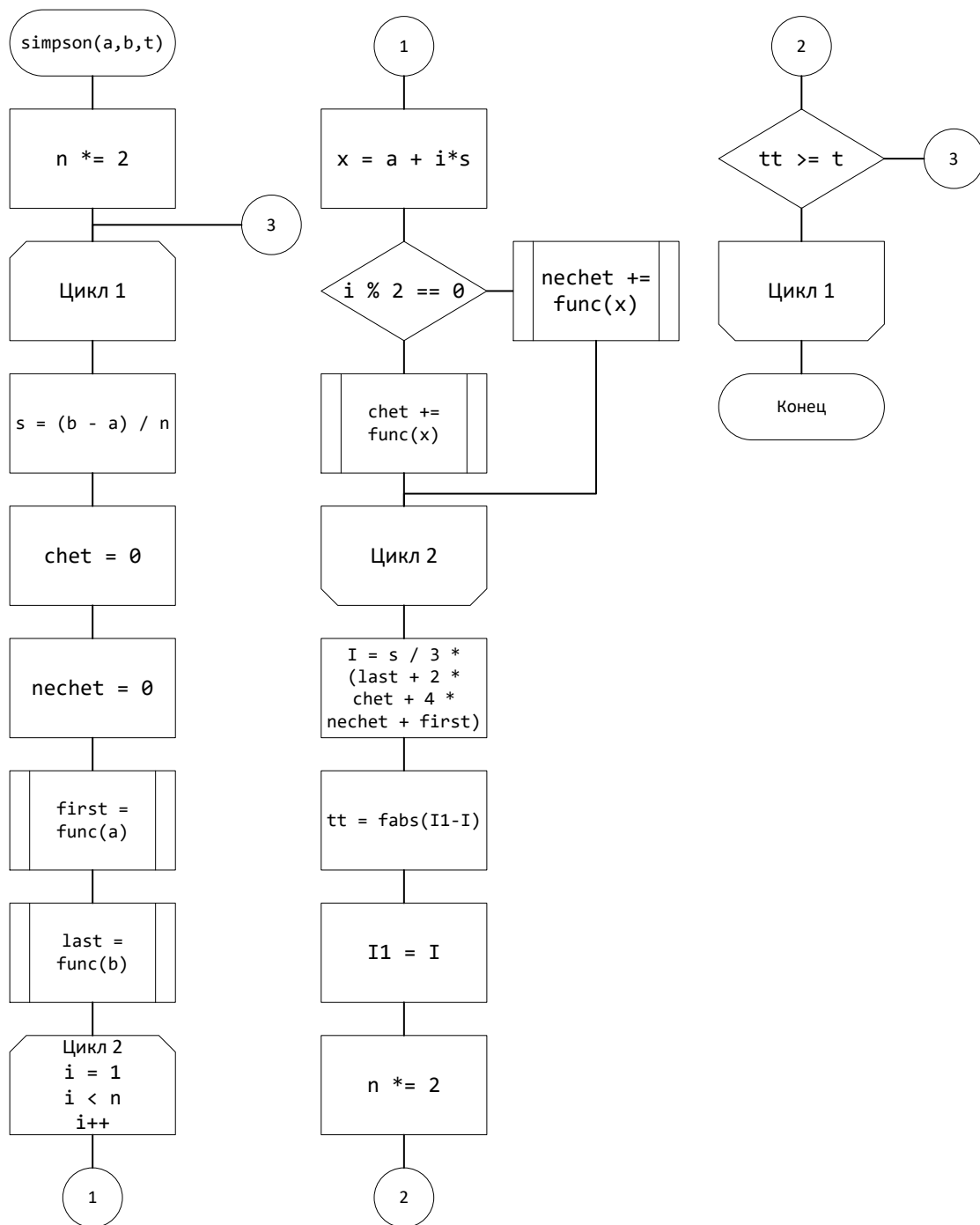
Выходные данные:

double Trapecia – значения от вычисления методом трапеций;

double Simpson – значения от вычисления методом Симпсоном;







```

#include <iostream>
#include <windows.h>
#include <conio.h>
#include <stdlib.h>
#define _USE_MATH_DEFINES
#include <math.h>
using namespace std;
double func(double x)
{
    return (sin(x)*cos(x)) / (2 + pow(sin(x), 2) - pow(cos(x), 2));
}
double simpson(double a, double b, double t) // метод Симпсона
{
    double I = 0, I1 = 0, s, chet, nechet, tt, n = 4; // I - интеграл, I1 - начальное
    значение, s - шаг разбиения, chet/nechet - четн./нечетн. член ряда, tt - разность, n -
    кол-во разбиений

```

```

double x = 0, first, last; //x - аргумент, first/last - первый|последний элемент
ряда
n *= 2;
do
{
    s = (b - a) / n;
    chet = 0; nechet = 0;
    first = func(a);
    last = func(b);
    for (int i = 1; i < n; i++)
    {
        x = a + i*s;
        if (i % 2 == 0)
        {
            chet += func(x);
        }
        else
        {
            nechet += func(x);
        }
    }
    I = s / 3 * (last + 2 * chet + 4 * nechet + first); //нахождение
интеграла
    tt = fabs(I1 - I);
    I1 = I;
    n *= 2; //удваивание отрезков разбиения
}
while (tt >= t);
return I;
}
double trapeze(double a, double b, double t) //метод трапеций
{
    double I = 0, I1 = 0, s, chet, nechet, tt, n = 4;
    double x = 0, first, last;
    do
    {
        s = (b - a) / n;
        double S = 0; //сумма
        first = func(a);
        last = func(b);
        for (int i = 1; i < n; i++)
        {
            x = a + i*s;
            S += func(x);
        }
        I = s / 2 * (last + 2 * S + first); // интеграл
        tt = fabs(I1 - I);
        I1 = I;
        n *= 2; // удваиваем кол-во отрезков разбиения
    }
    while (tt >= t);
    return I;
}
void main()
{
    setlocale(LC_ALL, "Russian");
    int a = 0, b = M_PI/2.; // пределы интегрирования
    double accuracy; // точность
    cout << "Введите точность: ";
    cin >> accuracy;
    if (accuracy <= 0 || accuracy > 1)
        cout << "Точность только от 0 до 1!\n";
    else
    {
        double Trapecia = trapeze(a, b, accuracy);
    }
}

```

```

        cout << "По методу трапеций :" << Trapecia << endl;
        double Simpson = simpson(a, b, accuracy);
        cout << "По методу Симпсона :" << Simpson << endl;
    }
    system("pause");
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $accuracy \in (0;1], accuracy > 0$	<p>$n=0.001$ Ожидаемые ответы: методом трапеций: 0.220437 методом Симпсона: 0.220545</p> <p>$n = 1$ Ожидаемые ответы: методом трапеций: 0.213601 методом Симпсона: 0.220569</p> <p><i>// чем меньше значение, тем более точным получается результат вычисления интеграла.</i></p>
2	Класс недопустимых значений $accuracy < 0$	<p>$accuracy = -1$; Ожидаемые ответы: сообщение об ошибке</p>
3	Класс недопустимых значений $accuracy > 0$	<p>$accuracy = 2$; Ожидаемые ответы: сообщение об ошибке</p>

Результаты тестирования:

```

C:\Windows\system32\cmd.exe
Введите точность: 1
По методу трапеций :0.213601
По методу Симпсона :0.220569
Для продолжения нажмите любую клавишу . . .

```

```

C:\Windows\system32\cmd.exe
Введите точность: 0.001
По методу трапеций :0.220437
По методу Симпсона :0.220545
Для продолжения нажмите любую клавишу . . .

```

```

C:\Windows\system32\cmd.exe
Введите точность: 2
Точность только от 0 до 1!
Для продолжения нажмите любую клавишу . . .

```

```

C:\Windows\system32\cmd.exe
Введите точность: -1
Точность только от 0 до 1!
Для продолжения нажмите любую клавишу . . .

```

Тема 3: «Указатели и динамическое распределение памяти»

Цель: получить практические навыки по работе с указателями и динамическому распределению памяти.

Задача 3.1:

Динамически выделить память под N элементов вещественного типа `float`.
Разработать *алгоритм* и программу обработки одномерного динамического массива. Осуществить циклический сдвиг элементов одномерного массива на k позиций.

Входные данные:

`int k, n` - n - кол-во элементов, k - сдвиг позиций;

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (кол-ва элементов массива и сдвига);
- 2 – заполнение массива;
- 3 – вывод массива на экран;
- 4 – сдвиг элементов массива;
- 5 – вывод сдвинутого массива на экран;

Выходные данные:

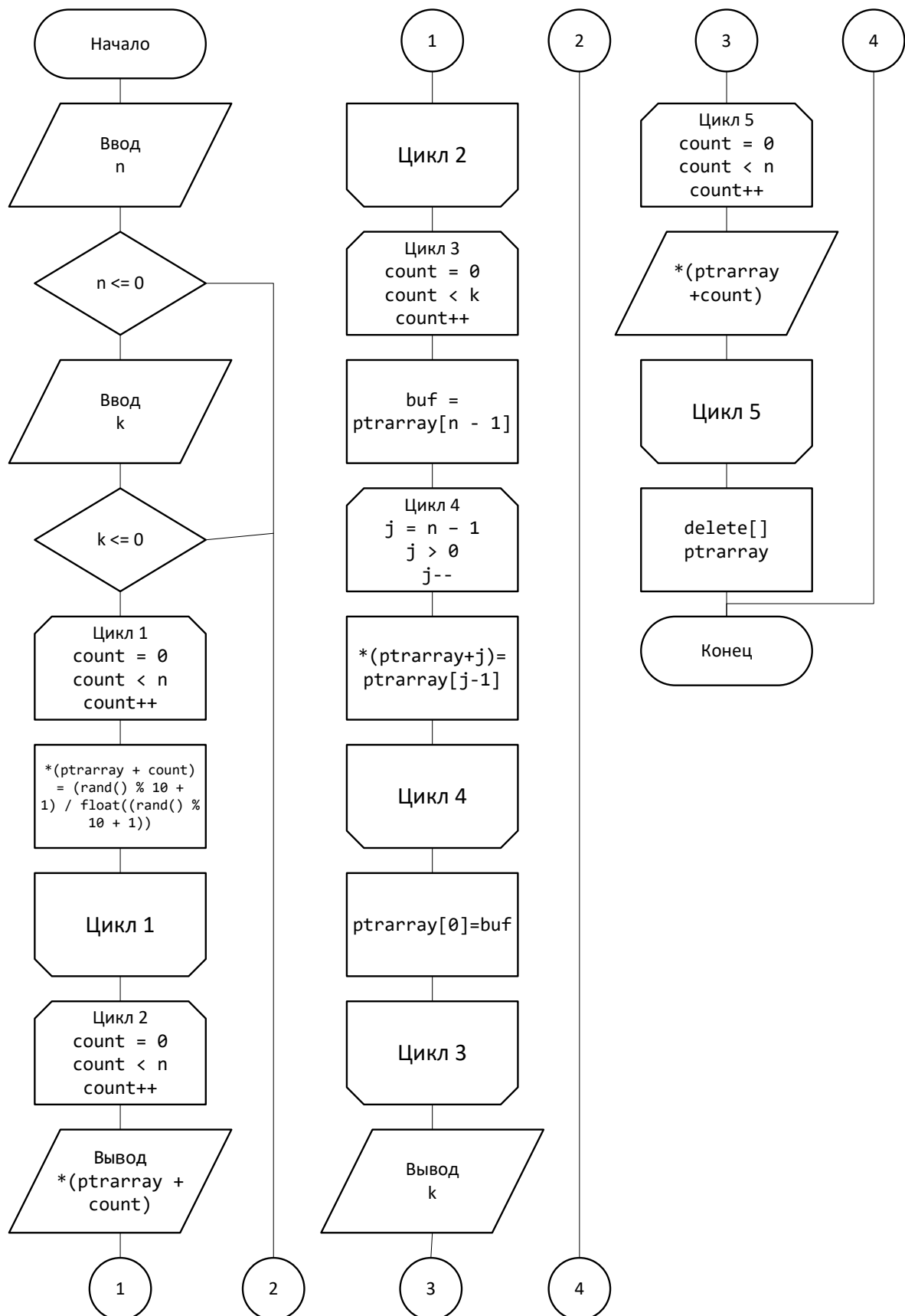
`*(ptrarray + count)` – вывод массива, вывод сдвинутого массива;

```
#include <iostream>
#include <stdio.h>
#include <ctime>
#include <iomanip> // в заголовочном файле <iomanip> содержится прототип функции
setprecision() Задаёт точность для плавающей запятой.
using namespace std;
int main(int argc, char* argv[])
{
    setlocale(LC_ALL, "Russian");
    srand(time(0)); // генерация случайных чисел
    int count, k, j, n; //count - индекс массива, j - индекс для сдвинутого массива,
    n - кол-во элементов, k - сдвиг позиций
    float buf; //буфер, хранящий сортируемый массив
    cout << "Введите кол-во элементов: "; cin >> n;
    if (n <= 0)
    { cout << "Ошибка!" << endl; return 0; }
    cout << "Введите сдвиг: "; cin >> k;
    if (k <= 0)
    { cout << "Ошибка!" << endl; return 0; }
    float *ptrarray = new float[n]; // создание динамического массива вещественных
    чисел на n элементов
    for (count = 0; count < n; count++) //заполнение массива случайными числами с
    масштабированием от 1 до 10
        *(ptrarray + count) = (rand() % 10 + 1) / float((rand() % 10 + 1));
    cout << "/ array / \n";
    for (count = 0; count < n; count++) //вывод массива
        cout << setprecision(2) << *(ptrarray + count) << "\n";
    for (count = 0; count < k; count++) //сдвиг на k позиций
    {
        buf = ptrarray[n - 1]; //запись в буфер
        for (j = n - 1; j > 0; j--)
            *(ptrarray + j) = ptrarray[j - 1]; //заполнение сдвинутого массива
        ptrarray[0] = buf; //очистка буфера
    }
}
```

```

cout << "\nСдвиг элементов массива на " << k << " позиций:/ \n";
for (count = 0; count < n; count++)//Вывод сдвинутого массива
    cout << setprecision(2) << *(ptrarray + count) << "\n";
delete[] ptrarray; // высвобождение памяти
cout << endl; system("pause"); return 0;
}

```



План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in [0; 2147483647], n \in \mathbb{N}$ $k \in [0; 2147483647], k \in \mathbb{N}$	$n = 5$ $k = 2$ Ожидаемые ответы: вывод массива из 5 элементов и сдвиг его на 2 позиции <i>//массив заполняется случайными числами, делаем сдвиг элементов массива вправо(вниз) на k позиций.</i>
2	Класс недопустимых значений $n \in (-\infty; 0], n \leq 0$	$n = 0$ $n = -1$ Ожидаемые ответы: сообщение об ошибке <i>//количество элементов должно быть больше нуля. Сдвиг не должен быть отрицательным.</i>
2	Класс недопустимых значений $k \in (-\infty; 0), k < 0$	$k = -1$ Ожидаемые ответы: сообщение об ошибке <i>//количество элементов должно быть больше нуля. Сдвиг не должен быть отрицательным.</i>

Результаты тестирования:

```

C:\Windows\system32\cmd.exe
Введите кол-во элементов: 5
Введите сдвиг: 2
/ array /
0.88
0.17
5
3
2
/Сдвиг элементов массива на 2 позиций:/
3
2
0.88
0.17
5
Для продолжения нажмите любую клавишу .

```

```

C:\Windows\system32\cmd.exe
Введите кол-во элементов: 0
Ошибка!
Введите сдвиг: -1
Ошибка!
Для продолжения нажмите любую клавишу . . .

```

Задача 3.2:

Динамически выделить память под N^2 элементов целого типа `int`. Разработать алгоритм и программу обработки квадратной матрицы порядка N : найти в матрице такие номера k , что k -ая строка матрицы совпадает с k -ым столбцом.

Входные данные:

`int n` – кол-во элементов массива;

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (кол-во строк и столбцов);
- 2 – заполнение массива;
- 3 – вывод массива на экран;
- 4 – нахождение таких номеров k , что k -ая строка матрицы совпадает с k -ым столбцом;
- 5 – вывод этих k на экран;

Выходные данные:

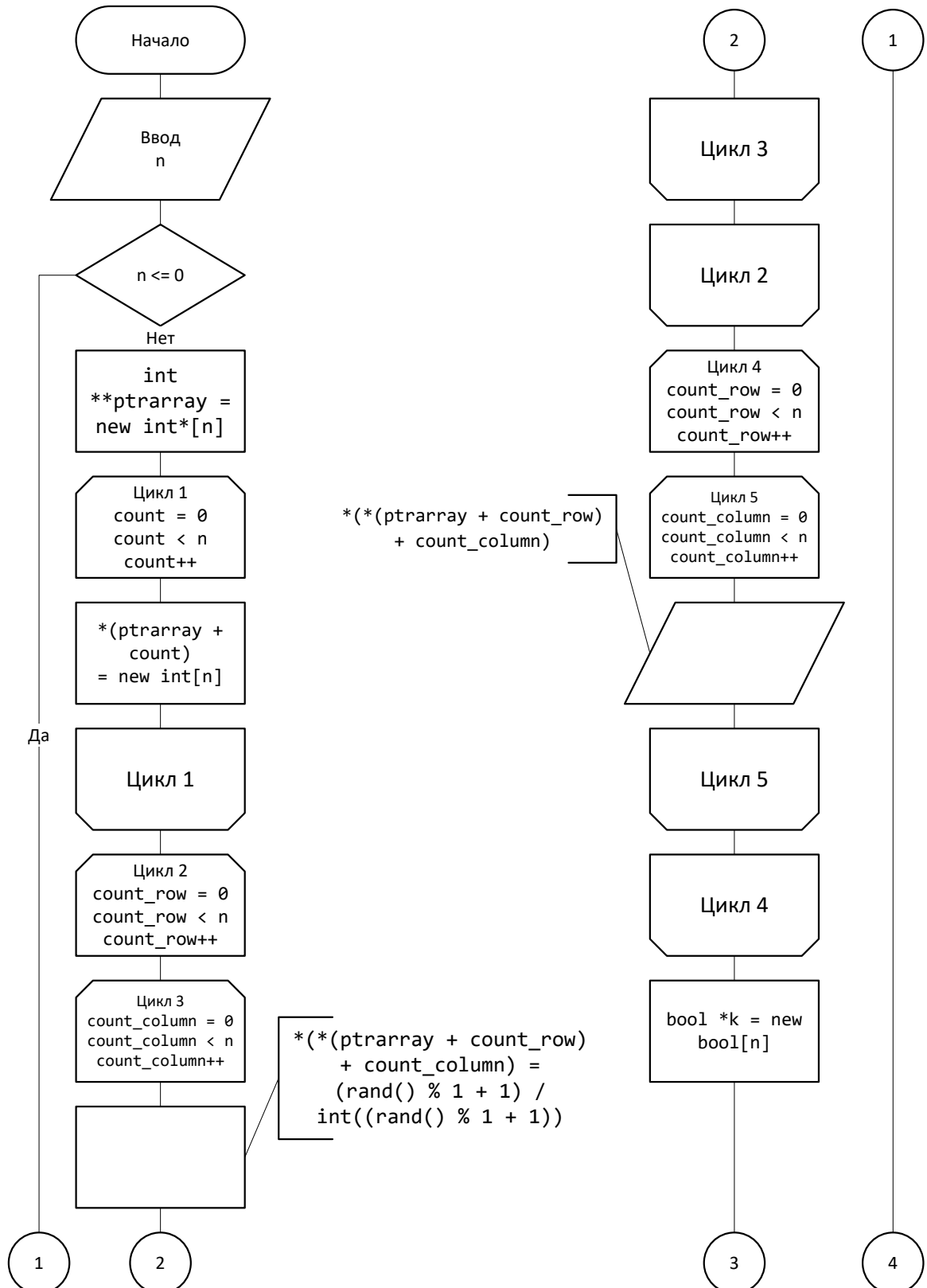
$i + 1$ – такие номера k , что k -ая строка матрицы совпадает с k -ым столбцом
сдвиг позиций;

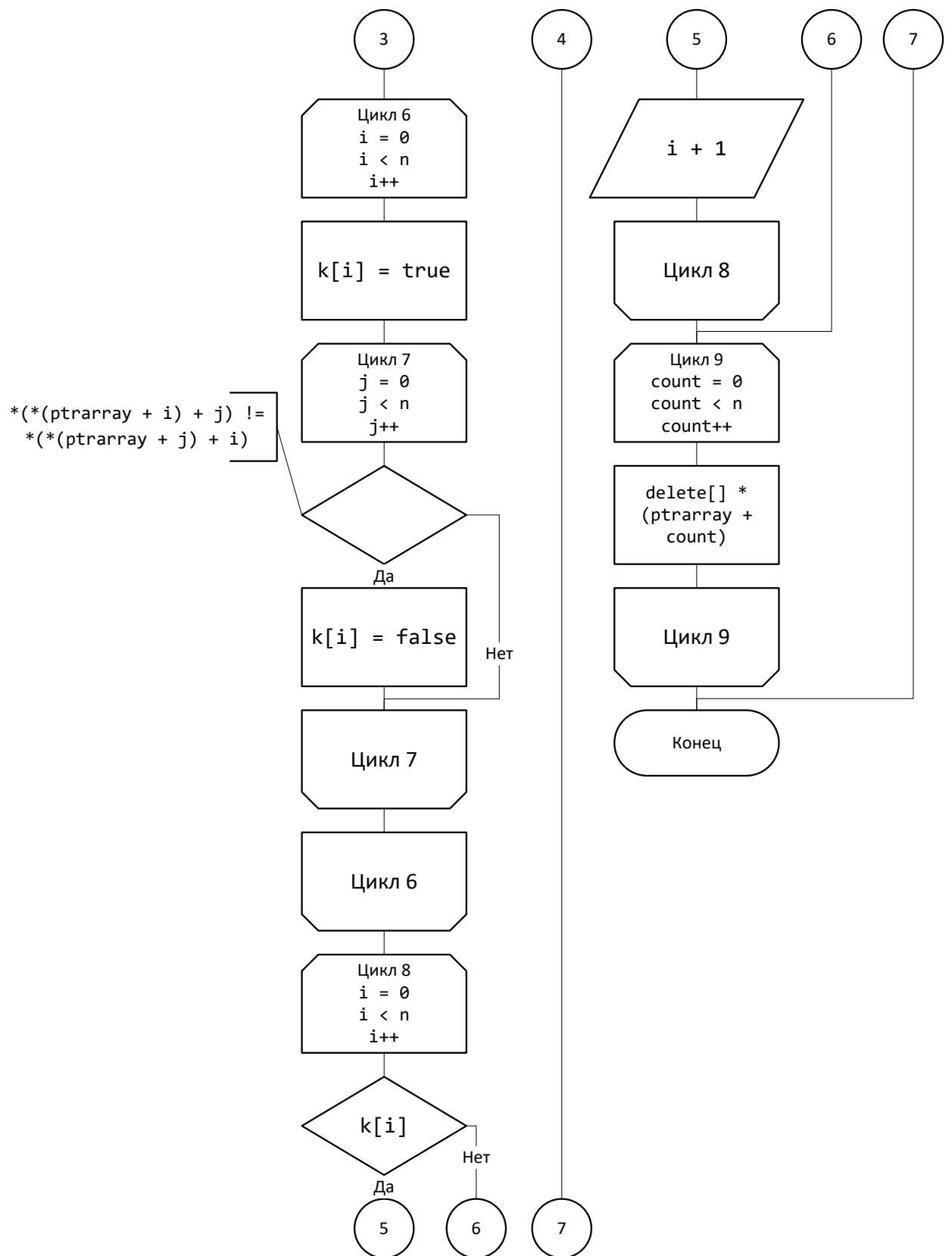
```
#include <iostream>
#include <stdio.h>
#include <ctime>
#include <iomanip> // в заголовочном файле <iomanip> содержится прототип функции
setprecision() Задаёт точность для плавающей запятой.
using namespace std;
int main(int argc, char* argv[])
{
    setlocale(LC_ALL, "Russian");
    srand(time(0)); // генерация случайных чисел
    int count, n; // count - индекс массива, j - индекс для сдвинутого массива, n -
    кол-во элементов, k - сдвиг позиций
    cout << "Введите кол-во строк и столбцов: "; cin >> n;
    if (n <= 0)
        { cout << "Ошибка!"; return 0; }
    int **ptrarray = new int*[n]; // создание динамического массива вещественных
    чисел на n элементов
    for (int count = 0; count < n; count++)
        *(ptrarray+count) = new int[n];
    for (int count_row = 0; count_row < n; count_row++)
        for (int count_column = 0; count_column < n; count_column++)
            (*(ptrarray + count_row) + count_column) = (rand() % 1 + 1) /
int((rand() % 1 + 1)); //заполнение массива случайными числами с масштабированием от 1
до 10
    cout << "/ array / \n";
    for (int count_row = 0; count_row < n; count_row++)
    {
        for (int count_column = 0; count_column < n; count_column++)
            cout << setw(4) << (*(ptrarray + count_row) + count_column) << "    ";
        cout << endl;
    }
    bool *k = new bool[n];
    for (int i = 0; i < n; i++)
    {
        k[i] = true;
        for (int j = 0; j < n; j++)
            if (*(ptrarray + i) + j) != (*(ptrarray + j) + i))
                { k[i] = false; break; }
    }
}
```

```

cout << "\nНомера k, что k -ая строка матрицы совпадает с k -ым столбцом: ";
for (int i = 0; i < n; i++)
    if (k[i])
        cout << i + 1 << " ";
for (int count = 0; count < n; count++)
    delete[] * (ptrarray + count);
cout << endl; system("pause"); return 0;
}

```





План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in [0; 2147483647], n \in \mathbb{N}$	$n = 3$ Ожидаемые ответы: вывод массива из 5 строк и столбцов, вывод k, если такие имеются <i>//номера k – индексы совпадающих строк и столбцов.</i>
2	Класс недопустимых значений $n \in (-\infty; 0], n \leq 0$ $n \in (-\infty; 0], n \leq 0$	$n = 0$ $n = -1$ Ожидаемые ответы: сообщение об ошибке

Результаты тестирования:

```
C:\Windows\system32\cmd.exe
Введите кол-во строк и столбцов: 3
/ array /
0      0      0
1      1      0
0      1      2
Номера k, что k -ая строка матрицы совпадает с k -ым столбцом:
Для продолжения нажмите любую клавишу . . .
```

```
C:\Windows\system32\cmd.exe
Введите кол-во строк и столбцов: 3
/ array /
1      1      0
2      1      0
0      0      0
Номера k, что k -ая строка матрицы совпадает с k -ым столбцом: 3
Для продолжения нажмите любую клавишу . . .
```

```
C:\Windows\system32\cmd.exe
Введите кол-во строк и столбцов: -1
Ошибка! Для продолжения нажмите любую клавишу . . .
```

Тема 4: «Обработка строчных величин»

Цель: получить практические навыки по обработке строчных величин.

Задача 4:

- Разработать функцию, которая возвращает строку символов, которая получена из строки S1 путём удаления символов с позиции N1 до позиции N2.
- Разработать функцию, которая возвращает строку символов, которая получена из строки S1 путём удаления символов с позиции N1 до позиции N2 и вставки на это место строки S2.

Все функции поместить в отдельном файле (библиотеке). Разработать проект, подключающий собственную библиотеку, для проверки результатов вызова функций.

Входные данные:

char str[100], substr[100] – строка и доп. строка;

int n1, n2 – позиции от и до, в пределах которых нужно удалить элементы и внедрить туда доп. строку;

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (обеих строк и диапазона);
- 2 – вывод начальной строки на экран;
- 3 – вывод строки с удалением на экран;
- 4 – вывод строки с добавлением доп. строки;

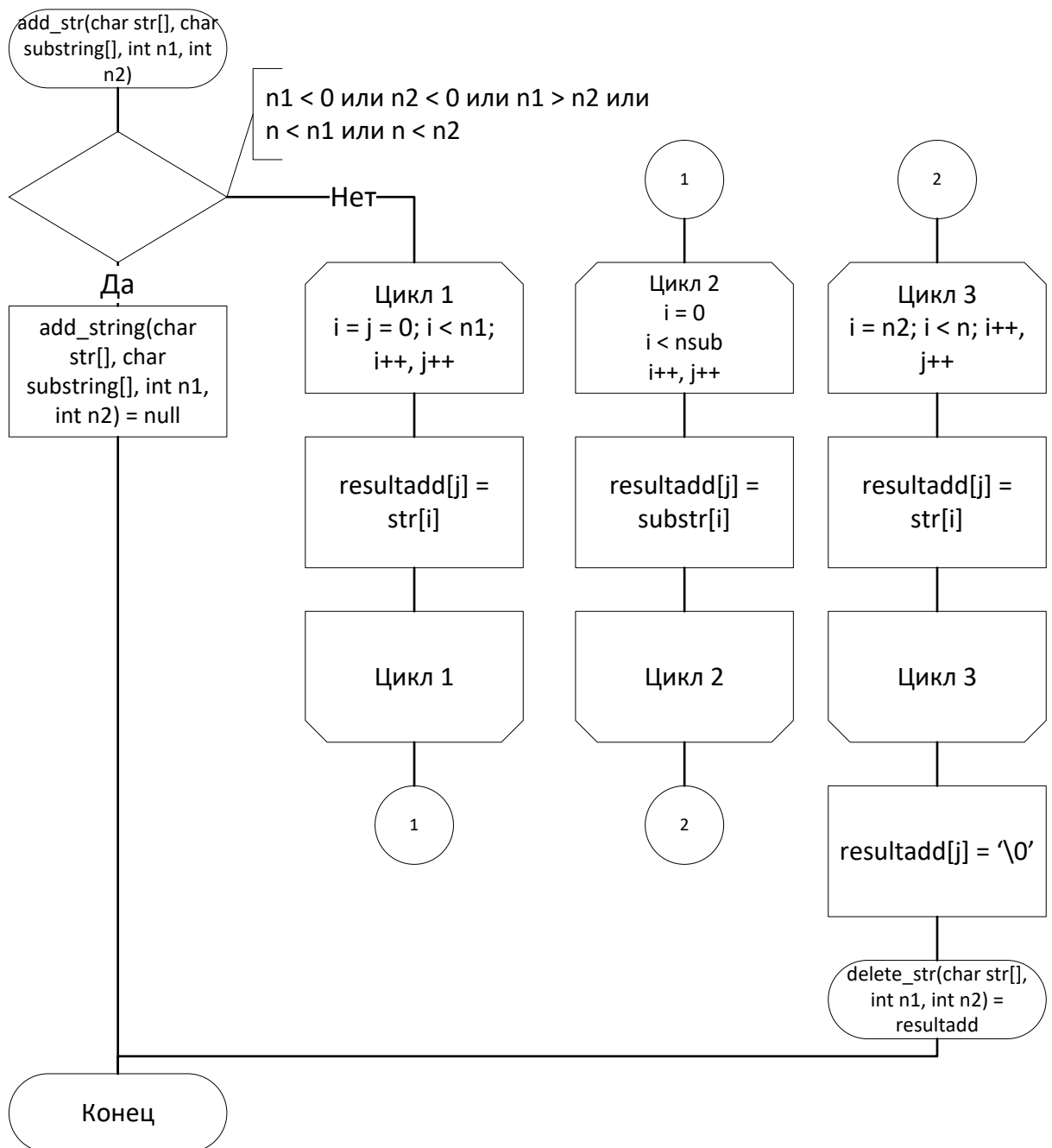
Выходные данные:

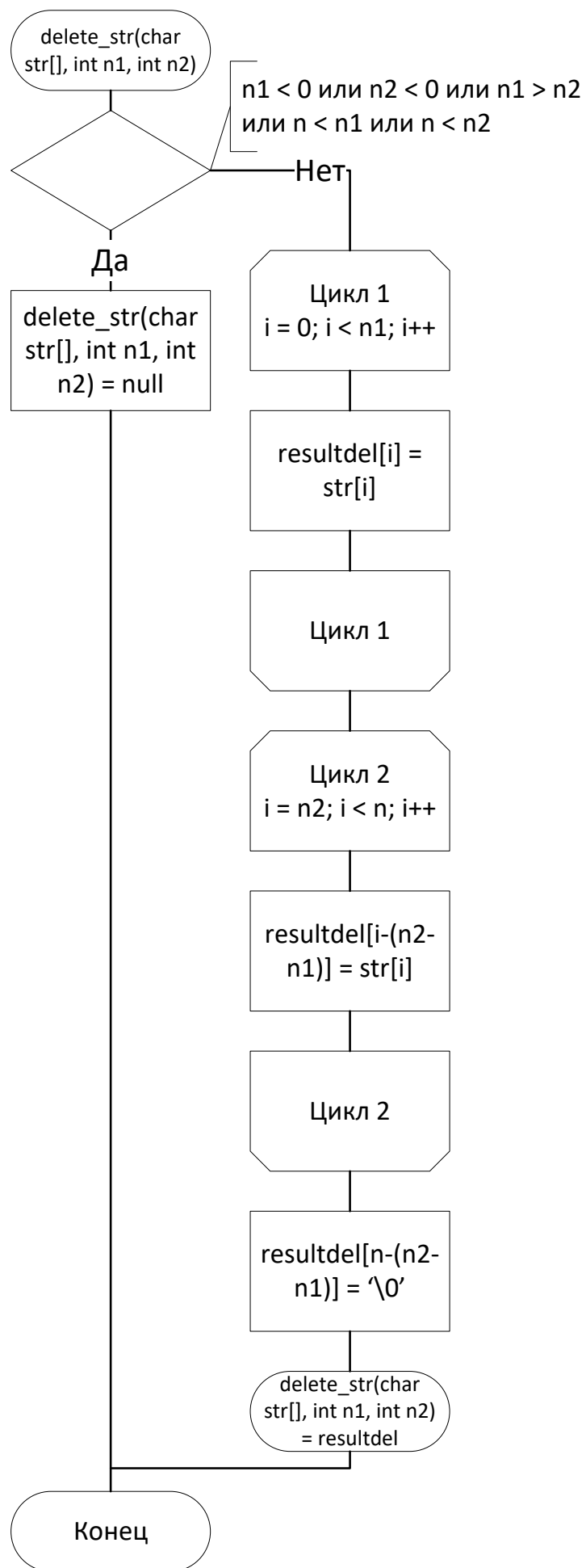
char str – первоначальная строка;

del – строка с удалёнными элементами;

add – строка с добавлением доп. строки;

```
#pragma ones
char* delete_str(char str[], int n1, int n2); //удаляет из строки символы в диапазоне;
Принимает строку и диапазон значений; Выводит либо ошибку, либо изменённую строку
char* add_str(char str[], char substr[], int n1, int n2); //Добавляет дополнительную
строку; Принимает строку и диапазон значений; Выводит либо ошибку, либо изменённую
строку
```





```

#include <iostream>
using namespace std;
char* delete_str(char str[], int n1, int n2)
{
    int n = strlen(str);
    if (n1 < 0 || n2 < 0 || n1 > n2 || n < n1 || n < n2)
        return NULL;
    else
    {
        char *resultdel = new char[n];
        for (int i = 0; i <= n1; i++)
            resultdel[i] = str[i]; //Записываем символы исходной строки до начала
индекса диапазона
        for (int i = n2; i < n; i++)
            resultdel[i - (n2 - n1)] = str[i]; //сдвигаем
        resultdel[n - (n2 - n1)] = '\0';
        return resultdel;
    }
}

char* add_str(char str[], char substr[], int n1, int n2)
{
    int n = strlen(str);
    int nsub = strlen(substr);
    if (n1 < 0 || n2 < 0 || n1 > n2 || n < n1 || n < n2)
        return NULL;
    else
    {
        char *resultadd = new char[n + nsub];
        int i; //индексы исходной строки
        int j; //индекс дополнительной строки
        for (i = j = 0; i < n1; i++, j++)
            resultadd[j] = str[i]; //Записываем исходную строку до начала диапазона
        for (i = 0; i < nsub; i++, j++)
            resultadd[j] = substr[i]; //Записываем подстроку в результирующую строку
        for (i = n2; i < n; i++, j++)
            resultadd[j] = str[i]; //В конец результирующей строки(после индекса
конца диапазона) дописываем оставшиеся символы исходной строки
        resultadd[j] = '\0';
        return resultadd;
    }
}

```

```

#include <iostream>
#include <conio.h>
#include <locale>
#include <Windows.h>
#include <lib.h>
using namespace std;
void main()
{
    setlocale(LC_ALL, "Rus");
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    char *del, *add;
    int n1, n2;
    cout << "Введите строку текста: ";
    char str[100]; gets_s(str);
    cout << "Введите доп. строку: ";
    char substr[100]; gets_s(substr);
    cout << endl << "Введите позиции от и до, в пределах которых нужно удалить
символы: ";
    cin >> n1 >> n2;
    cout << endl << "Начальная строка: " << str << endl << endl;
}

```

```

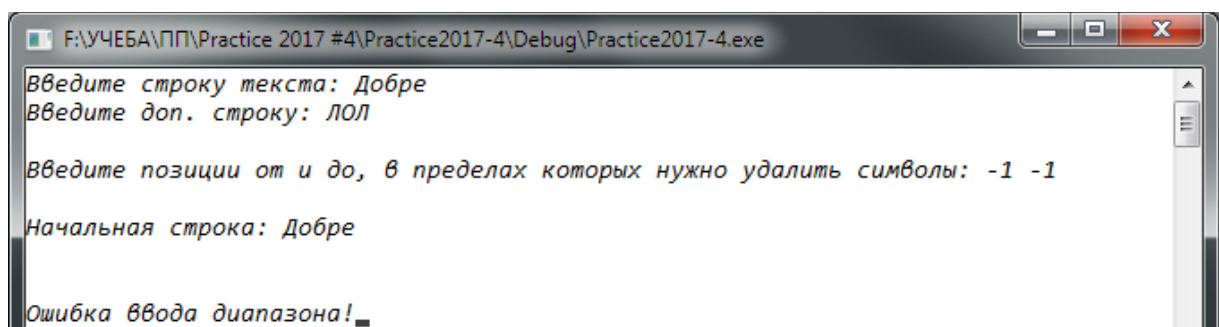
del = delete_str(str, n1, n2); cout << endl;
add = add_str(str, substr, n1, n2);
if (del == NULL || add == NULL)
    cout << "Ошибка ввода диапазона!";
else
{
    cout << "Строка с удалением: " << endl << del << endl << endl;
    cout << "Строка с добавлением: " << endl << add;
}
_getch();
}

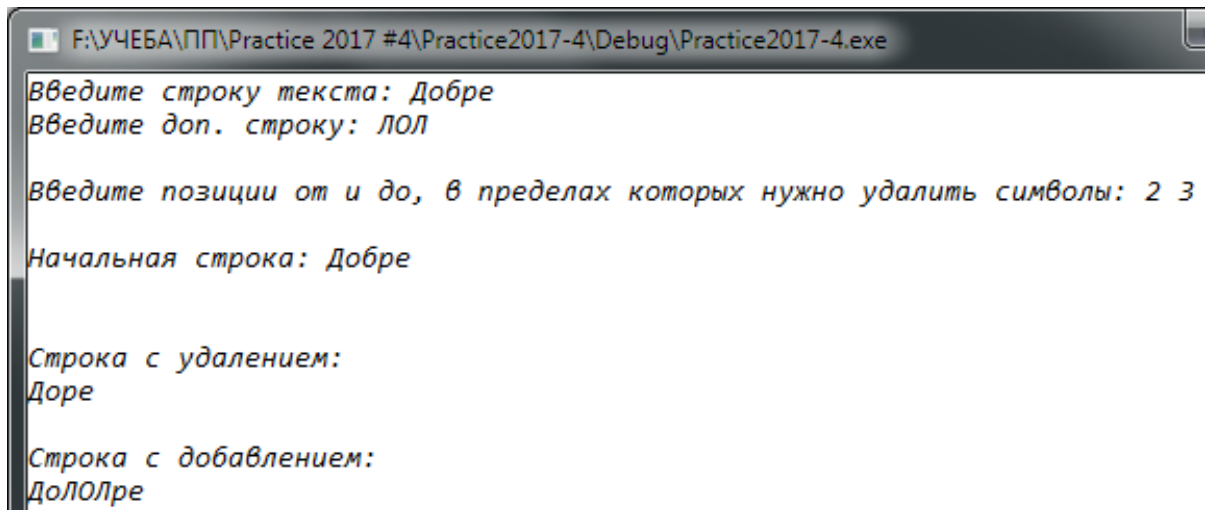
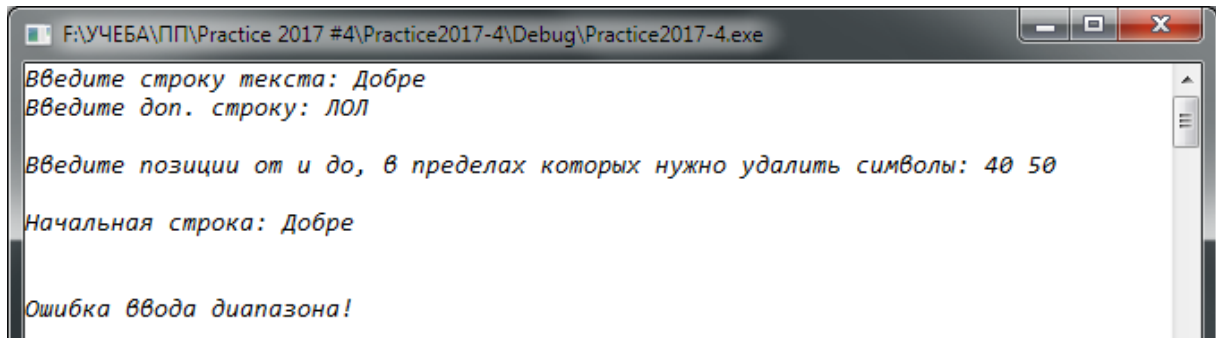
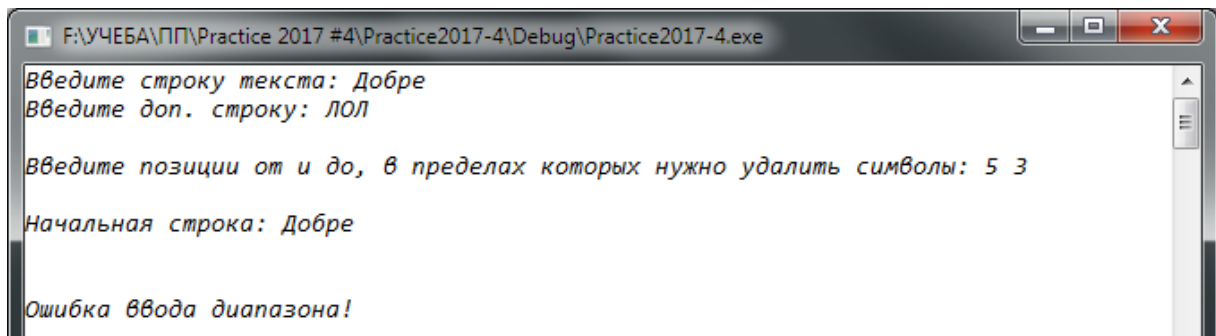
```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $str \in (0;100], str > 0$ $substr \in (0;100], substr > 0$ $n1 > 0, n1 < n2, n1 < n$ $n2 > 0, n2 > n1, n2 < n$	str = Добре substr = ЛОЛ Ожидаемые ответы: ...
2	Класс допустимых значений $n1 \in (0;+\infty), n1 > 0$ $n2 \in (0;+\infty), n1 > 0$	n1 = 2 n2 = 3 Ожидаемые ответы: ДоЛОЛре <i>//сначала из исходной строки удаляются символы в определённом диапазоне, при этом первый символ строки имеет нулевой индекс</i>
3	Класс недопустимых значений $n1 < 0, n2 < 0$	n1=-1 n2=-1 Ожидаемые ответы: ошибка ввода диапазона
4	Класс недопустимых значений $n1 > n2$	n1 = 5 n2 = 3 Ожидаемые ответы: ошибка ввода диапазона
5	Класс недопустимых значений $n < n1, n < n2$	n = 10 n1 = 40 n2 = 50 Ожидаемые ответы: ошибка ввода диапазона

Результаты тестирования:





Тема 5: «Работа с файлами»

Цель: получить практические навыки по работе с файлами.

Задача 5.1:

Создать в редакторе текстовый файл, внести в него 10 строк произвольного текста. Разработать программу, которая:

- выводит на экран строки файла, в которых встречается заданный символ
- формирует другой файл, в который переписывает строки исходного файла, содержащие латинские буквы, и указывает после каждой строки количество латинских букв в ней.

Входные данные:

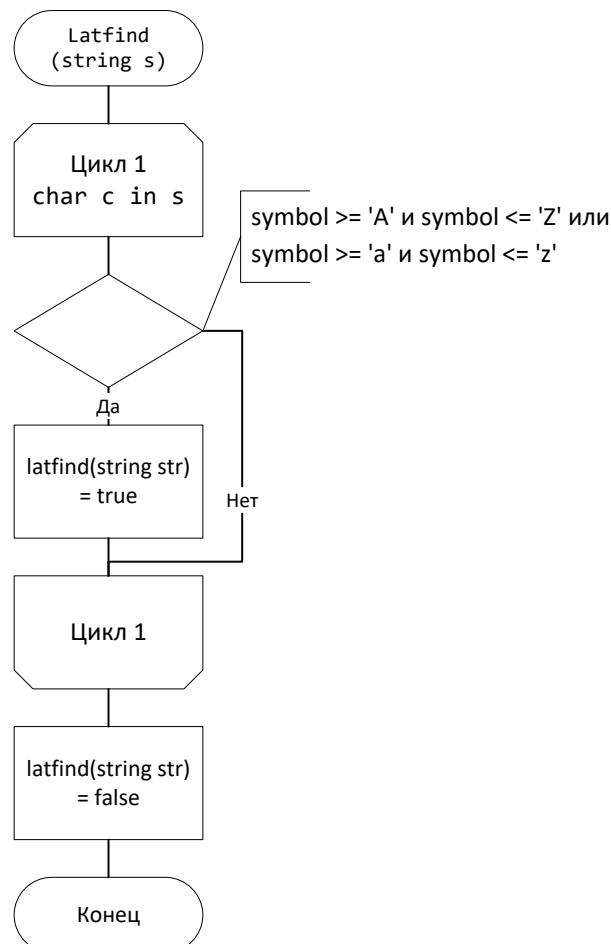
string bukva – ввод буквы для поиска;

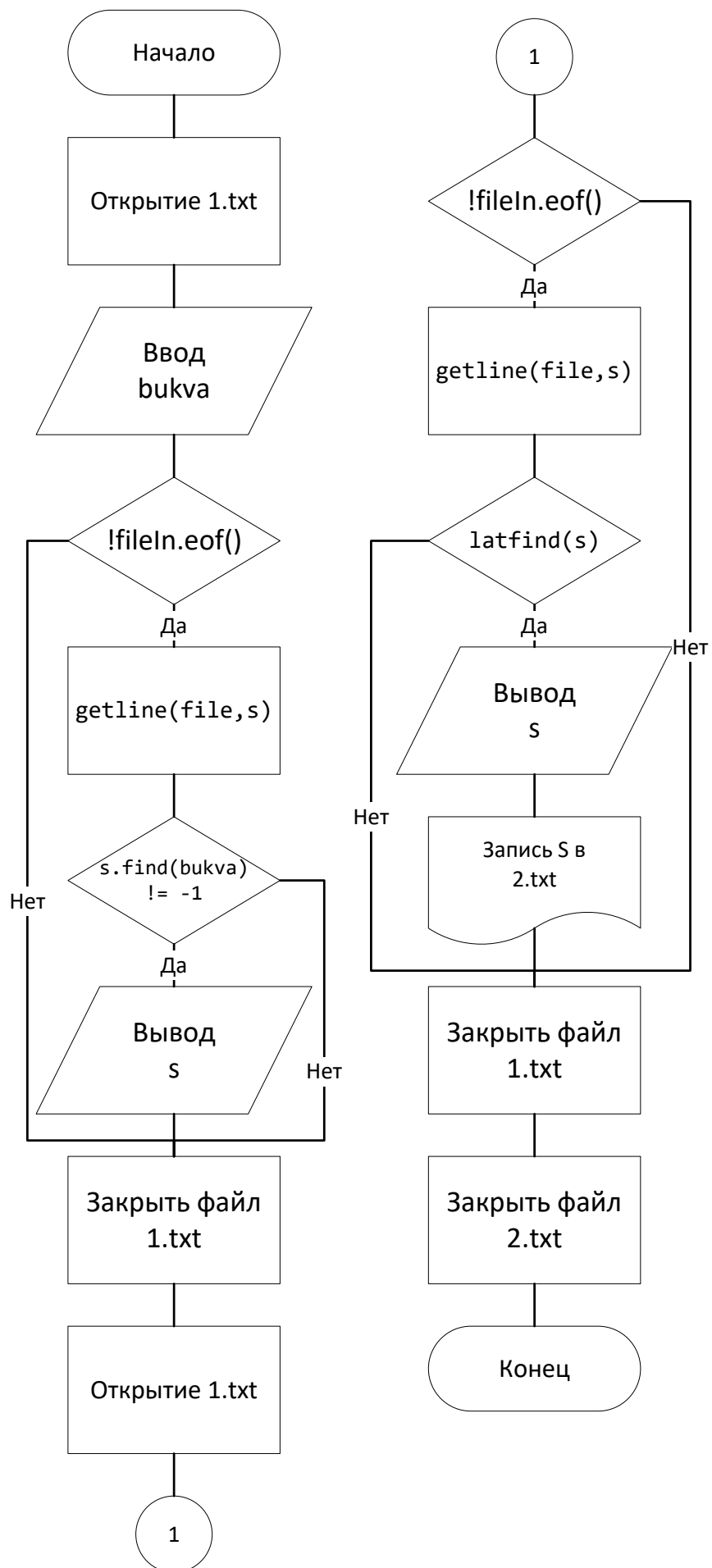
Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (буквы для поиска);
- 2 – вывод строк, содержащих искомую букву;
- 3 – запись строк в файл, содержащих латиницу;

Выходные данные:

string s – вывод строки с найденной буквой, записывается в файл, если найдена латиница;





```

#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <windows.h>
using namespace std;
bool latfind(string s)
{
    for each (char c in s)
    {
        if (c >= 'A' && c <= 'Z' || c >= 'a' && c <= 'z')
            return true;
    }
    return false;
}
int main()
{
    SetConsoleCP(1251); // Ввод с консоли в кодировке 1251
    SetConsoleOutputCP(1251); // Вывод на консоль в кодировке 1251. Нужно только
будет изменить шрифт консоли на Lucida Console или Consolas
    ifstream file("1.txt"); // открыли файл с текстом
    string s, буква;
    char c;
    int pos;
    cout << "Тыкни букву: ";
    cin >> буква;
    cout << "\n";
    while (!file.eof()) // поиск
    {
        getline(file, s);
        if (s.find(буква) != -1)
            cout << s << endl;
    }
    file.close();
    file.open("1.txt");
    cout << "\n\nПредложения с ин.язом (также запишутся в файл):\n\n";
    ofstream zap("2.txt");
    while (!file.eof()) // поиск
    {
        getline(file, s);
        if (latfind(s))
        {
            cout << s << endl;
            zap << s << endl;
        }
    }
    zap.close();
    file.close(); // обязательно закрыли
    system("Pause");
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений <i>буква – любой символ</i>	буква = о Ожидаемые ответы: вывод строк с буквой «о» и запись предложений с кириллицей в другой файл
2	Класс недопустимых значений если не удалось открыть файл (path)	Ожидаемые ответы: Ошибка открытия файла!

Результаты тестирования:

```
1.txt Исходный код.cpp
Новый дизайн VK = -13/10
Наша система образования просто "топ"
Lmao this is char of fresh memes
Кто вообще это читает?
Google: Как придумать 10 строк случайного текста?
Люблю ПП
Красный Кардинал Красного цвета
This is 8th string
NEIN!!!
Конечно строка, вы!!!
```

```
F:\УЧЕБА\ПП\Practice 2017 #5,1\Practice2017-5.1\Debug\Practice2017-
Тыкни букву: o
Новый дизайн VK = -13/10
Наша система образования просто "топ"
Кто вообще это читает?
Google: Как придумать 10 строк случайного текста?
Красный Кардинал Красного цвета
Конечно строка, вы!!!
Предложения с ин.язом (также запишутся в файл):
Новый дизайн VK = -13/10
Lmao this is char of fresh memes
Google: Как придумать 10 строк случайного текста?
This is 8th string
NEIN!!!
Для продолжения нажмите любую клавишу . . .
```

```
2.txt 1.txt Исходный код.cpp
Новый дизайн VK = -13/10
Lmao this is char of fresh memes
Google: Как придумать 10 строк случайного текста?
This is 8th string
NEIN!!!
```

Задача 5.2:

Разработать программу для заполнения двоичного файла целыми числами в интервале $[1; N]$. Получить новый файл из компонент исходного файла, являющимися полными квадратами.

Входные данные:

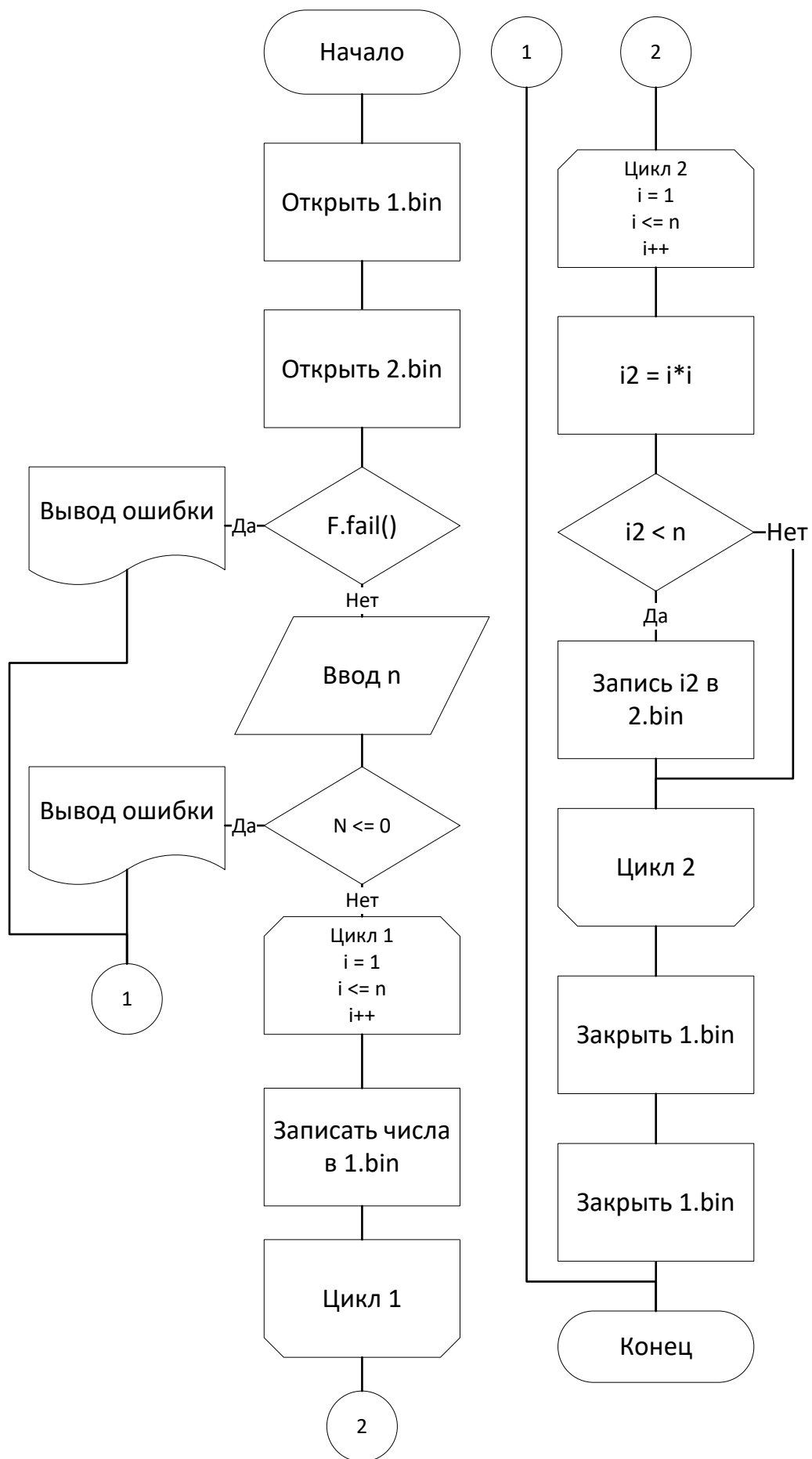
int n – предел для целых чисел;

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (предела для целых чисел);
- 2 – заполнение двоичного файла целыми числами в заданном диапазоне;
- 3 – вывод чисел в заданном диапазоне на экран;
- 4 – запись во 2ой компонент исходного файла, являющимися полными квадратами.
- 5 – вывод полных квадратов на экран;

Выходные данные:

- int i* – вывод чисел;
- int i2* – вывод полных квадратов;



```

#include <iostream>
#include <string>
#include <fstream>
#include <sstream>
#include <windows.h>
using namespace std;
bool IsSquare(int n,)//используется тот факт, что любой квадрат это сумма
последовательных нечетных чисел
{
    int i = 1;
    while (n>0)
    {
        n -= i;
        i += 2;
    };
    if (n == 0)return true;
    return false;
};
void main()
{
    setlocale(0, ""); // локализация
    int n = 0, x = 1; // i - счетчик, n - количество чисел
    ofstream F("1.bin", std::ios::binary); // создание файловой переменной, поток
для записи в файл
    ofstream F2("2.bin", std::ios::binary);
    if (F.fail()) // если открытие файла прошло некорректно, то
    {
        cerr << "Ошибка открытия файла. Проверьте местоположение и имя файла! \n"; //
вывод сообщения об ошибке
    }
    else
    {
        cout << "Введите n = ";
        if (n <= 0)
        {
            cout << "Ошибка!";
            return 0;
        }
        cin >> n; cout << endl;
        for (int i = 1; i <= n; i++)// цикл для ввода целых чисел и записи их в
файл
        {
            F.write((char*)&i, sizeof(int));
            cout << i << ' ';
        }
        cout << "\n";
        for (int i = 1; i <= n; i++)// проверка числа на полный квадрат
        {
            int i2 = i*i;
            if (i2 < n)
            {
                F2.write((char*)&i2, sizeof(int));
                cout << i2 << ' ';
            }
        }
        cout << "\n";
        F.close(); // закрытие потока
        F2.close();
        system("pause");
    }
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in [1; 2147483647), n \in \mathbb{N}$	$n = 10$ Ожидаемые ответы: 1,2,3,4,5,6,7,8,9,10 и полные квадраты: 1,4,9
2	Класс недопустимых значений $n \in (-\infty; 0], n \leq 0$	$n = 0$ $n = -1$ Ожидаемые ответы: сообщение об ошибке

Результаты тестирования:

```

F:\УЧЕБА\ПП\Practice 2017
Введите n = 10
1 2 3 4 5 6 7 8 9 10
1 4 9
Для продолжения нажми

```

```

F:\УЧЕБА\ПП\Pra
Введите n = 0
Ошибка!Для про

```

```

F:\УЧЕБА\ПП\Pra
Введите n = -1
Ошибка!Для про

```

```

2.bin 1.bin Исходный код.cpp
00000000 01 00 00 00 02 00 00 00 03 00 00 00 04 00 00 00
00000010 05 00 00 00 06 00 00 00 07 00 00 00 08 00 00 00
00000020 09 00 00 00 0A 00 00 00

2.bin 1.bin Исходный код.cpp
00000000 01 00 00 00 04 00 00 00 09 00 00 00

```


Тема 6: «Динамические структуры данных»

Цель: получить практические навыки по работе с динамическими структурами данных;

Задача 6:

Составить программу обработки динамической структуры данных: поменять местами первый и последний элементы односвязной очереди Q.

Входные данные:

int n – ввод длины очереди;

queue<int> queue – ввод очереди;

Функциональные характеристики:

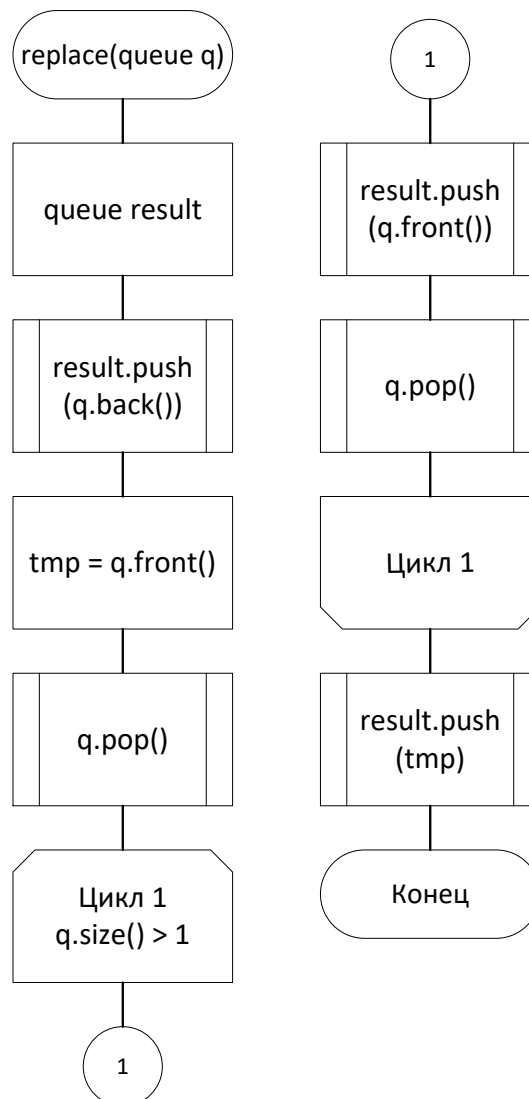
1 - ввод данных пользователем с клавиатуры (длины очереди, затем и самой очереди);

2 – замена в очереди местами первого и последнего элементов;

3 – вывод измененной очереди на экран;

Выходные данные:

replace(queue) – вывод измененной очереди;



```

#include <iostream>
#include <string>
#include <queue>
#include <conio.h>
#include <Windows.h>
using namespace std;
    //Функция смены местами первого и последнего элемента очереди работает следующим
    //образом:
    //Создаём результирующую очередь, в которую отправляем последний элемент
    //основной очереди, т.е. получается, что последний элемент станет первым.
    //Затем из основной мы сохраняем первый элемент во временную переменную. После
    //сохранения этого элемента удаляем его.
    //Циклически проверяем всю основную очередь до тех пор, пока её размер больше
    //одного.
    //В этом цикле мы переписываем первый элемент основной очереди в результирующую
    //и удаляем его из основной очереди. (очередь будет опустошаться, поэтому каждый
    //раз передаём первый элемент)
    //После прохода цикла - возвращаем в конец результирующей очереди первый
    //элемент, который мы запомнили.
template <typename T> //параметр шаблона, принимающий любой встроенный тип данных
queue<T> replace(queue<T> q)
{ //Функция для замены, q - очередь
    queue<T> result; //делаем результирующую очередь
    result.push(q.back()); //выводим последний элемент в начало
    T tmp = q.front(); //сохраняем во временную переменную первый элемент
    q.pop(); //удаляем первый элемент
    while (q.size() > 1)
    { //очередь должна содержать больше одного элемента
        result.push(q.front()); //передать в результат очереди первый элемент
        q.pop(); //удаляет первый элемент
    }
    result.push(tmp); //добавить в конец сохранённый элемент
    return result;
}

    //Функция вывода очереди на экран:
    //Выводим описание очереди.
    //Проходимся в цикле по элементам очереди, выводим и удаляем первый элемент
    //очереди. (Так как мы выводим первый элемент, нам нужно его удалить, чтобы
    //вывести следующий)
template <typename T> //параметр шаблона, принимающий любой встроенный тип данных
void print(queue<T> q, string name)
{ //передаётся очередь и строка с описанием
    cout << name;
    while (!q.empty())
    { //пока очередь не пуста
        cout << q.front() << " "; //вывод первого элемента
        q.pop(); //удаляет первый элемент
    }
    cout << endl;
}

    //Функция чтения очереди:
    //Выводим описание функции.
    //T x - x может принимать любой тип данных, в зависимости от указанного к
    //очереди.
    //Ручной ввод элементов очереди в зависимости от количества элементов
template <typename T> //параметр шаблона, принимающий любой встроенный тип данных
queue<T> read(int n, string invite)
{ //чтение очереди по количеству элементов, передаётся количество элементов и строка с
    //описанием
    queue<T> result;
    cout << invite;
    T x;
    for (int i = 0; i < n; i++)
    {
        cin >> x;
    }
}

```

```

        result.push(x); //Передать в конец элементы
    }
    if (n == 0)
        cout << endl;
    return result;
}

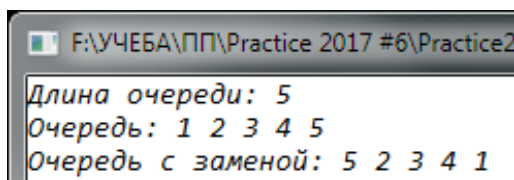
int main()
{
    setlocale(LC_ALL, "Rus");
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    cout << "Длина очереди: ";
    int n;
    cin >> n; //Ввод длины очереди
    if (n <= 0 || n == 1)
    {
        cout << "Неверный ввод!" << endl;
        return -1;
    }
    queue<int> queue = read<int>(n, "Очередь: "); //очередь работает с целыми
    числами (чтение очереди)
    print(replace(queue), "Очередь с заменой: "); //вывести очередь с изменёнными
    элементами
    _getch();
    return 0;
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений $n \in (0; +\infty), n > 0$	$n = 5$ Ожидаемые ответы: 5 2 3 4 1
2	Класс недопустимых значений $n \in (-\infty; 0], n \leq 0$	$n = -1$ $n = 0$ Ожидаемые ответы: сообщение об ошибке
4	Класс недопустимых значений $n = 1$	$n = 1$ Ожидаемые ответы: сообщение об ошибке

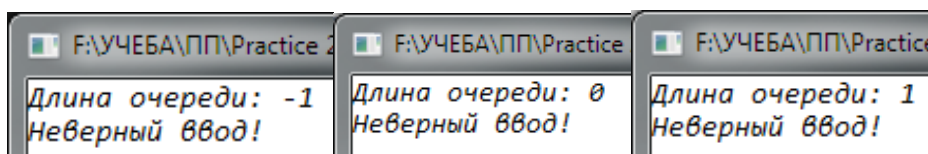
Результаты тестирования:



```

F:\УЧЕБА\ПП\Practice 2017 #6\Practice2
Длина очереди: 5
Очередь: 1 2 3 4 5
Очередь с заменой: 5 2 3 4 1

```



```

F:\УЧЕБА\ПП\Practice 2    F:\УЧЕБА\ПП\Practice    F:\УЧЕБА\ПП\Practice
Длина очереди: -1        Длина очереди: 0        Длина очереди: 1
Неверный ввод!          Неверный ввод!          Неверный ввод!

```

Тема 7: «Работа с файловой системой»

Цель: получить практические навыки по работе с файловой системой.

Задача 7:

Составить программу по работе с файловой системой, которая позволяет узнать системное время, и если время от 845 до 1115 – выводит сообщение «Доброе утро!» и даёт звуковой сигнал, а также отыскивает в указанном каталоге файлы, созданные в это время.

Входные данные:

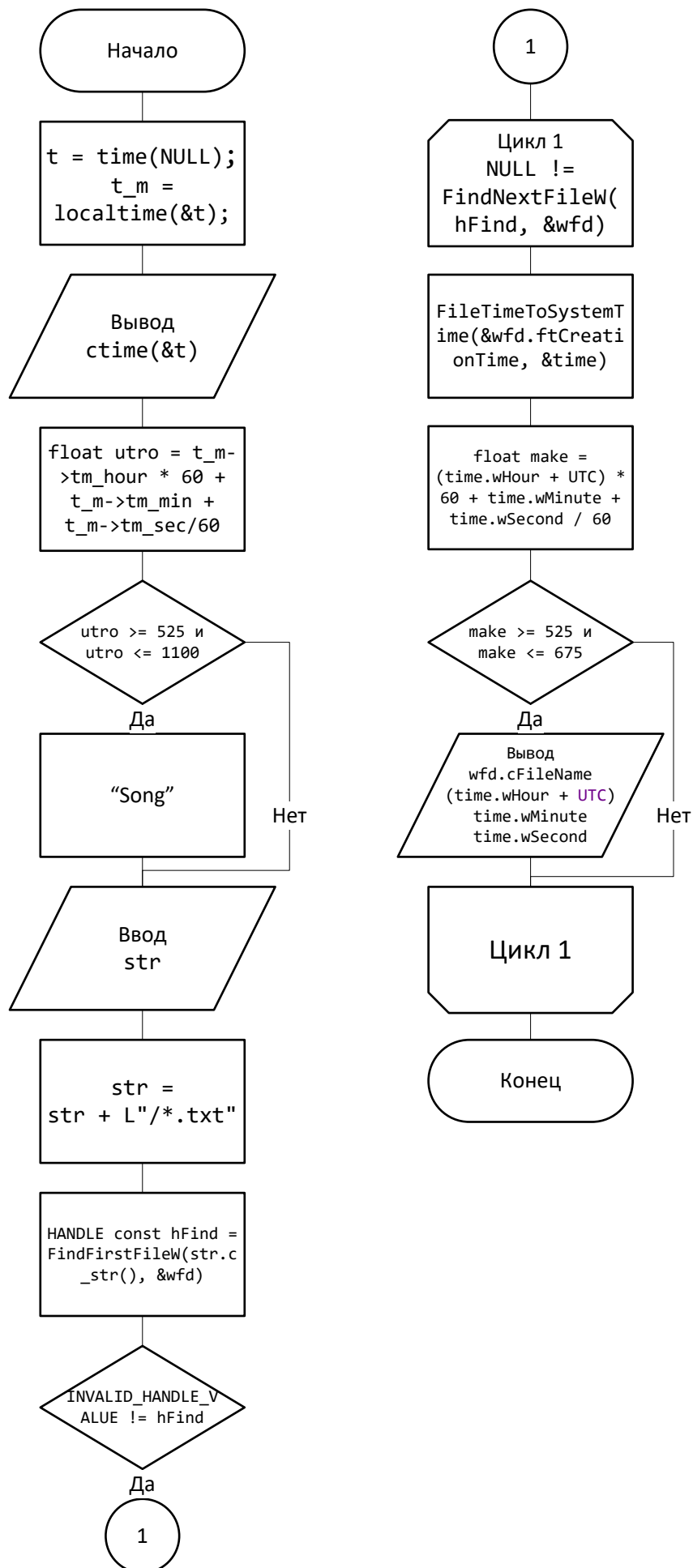
string str – ввод директории;

Функциональные характеристики:

- 1 - ввод данных пользователем с клавиатуры (директории);
- 2 – проверка времени на «утро»;
- 3 – получение информации о времени создания файлов;
- 4 – сравнение времени создания файла и времени «утра»;
- 5 – вывод файлов, созданных « утром»;

Выходные данные:

wfd.cFileName – имя файла;
(*time.wHour* + *UTC*) – часы создания;
time.wMinute – минуты создания;
time.wSecond – секунды создания;



```

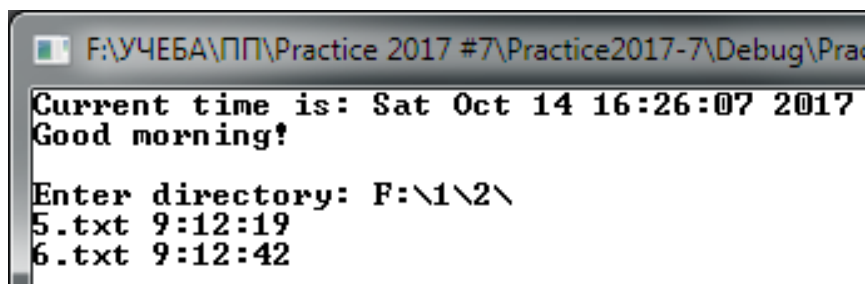
#include <iostream>
#include <conio.h>
#include <windows.h>
#include <time.h>
#include <ctime>
#include <string>
#define UTC (+3)
using namespace std;
int main()
{
    time_t t;
    struct tm *t_m;
    t = time(NULL);
    t_m = localtime(&t);
    wstring str;
    cout << "Current time is: " << ctime(&t); //вывод текущего времени
    float utro = t_m->tm_hour * 60 + t_m->tm_min + t_m->tm_sec/60;
    if (utro >= 525 && utro <= 675) //675
    {
        //вывод сообщения с последующей звуковой темой из фильма «Миссия невыполнима»
        cout << "Good morning!\n";
        Beep(784, 150);Sleep(300);Beep(784, 150);Sleep(300);
        Beep(932, 150);Sleep(150);Beep(1047, 150);Sleep(150);
        Beep(784, 150);Sleep(300);Beep(784, 150);Sleep(300);
        Beep(699, 150);Sleep(150);Beep(740, 150);Sleep(150);
        Beep(784, 150);Sleep(300);Beep(784, 150);Sleep(300);
        Beep(932, 150);Sleep(150);Beep(1047, 150);Sleep(150);
        Beep(784, 150);Sleep(300);Beep(784, 150);Sleep(300);
        Beep(699, 150);Sleep(150);Beep(740, 150);Sleep(150);
        Beep(932, 150);Beep(784, 150);Beep(587, 1200);Sleep(75);
        Beep(932, 150);Beep(784, 150);Beep(554, 1200);Sleep(75);
        Beep(932, 150);Beep(784, 150);Beep(523, 1200);Sleep(150);
        Beep(466, 150);Beep(523, 150);Beep(784, 150);Sleep(300);
        Beep(784, 150);Sleep(300);Beep(932, 150);Sleep(150);
        Beep(1047, 150);Sleep(150);Beep(784, 150);Sleep(300);
        Beep(784, 150);Sleep(300);Beep(699, 150);Sleep(150);
        Beep(740, 150);Sleep(150);Beep(784, 150);
    }
    cout << "\nEnter directory: "; //ввод директории
    wcin >> str;
    str = str + L"/*.txt";
    SYSTEMTIME time;
    WIN32_FIND_DATA wfd;
    HANDLE const hFind = FindFirstFileW(str.c_str(), &wfd); //поиск файлов
    if (INVALID_HANDLE_VALUE != hFind)
    {
        do
        {
            FileTimeToSystemTime(&wfd.ftCreationTime, &time);
            float make = (time.wHour + UTC) * 60 + time.wMinute + time.wSecond
/ 60; //задание формулы для времени создания файла
            if (make >= 525 && make <= 675) //вывод информации о времени
создания файлов
                wcout << wfd.cFileName << ' ' << (time.wHour + UTC) << ':' <<
time.wMinute << ':' << time.wSecond << endl;
        } while (NULL != FindNextFileW(hFind, &wfd));
        FindClose(hFind);
    }
    _getch();
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений адрес существующего каталога	Ожидаемые ответы: вывод имен файлов и времени их создания. <i>//выводятся только те файлы, которые были созданы с 8⁴⁵ до 11¹⁵</i>
2	Класс недопустимых значений адрес несуществующего каталога	Ожидаемые ответы: завершение программы

Результаты тестирования:



```

F:\УЧЕБА\ПП\Practice 2017 #7\Practice2017-7\Debug\Prac
Current time is: Sat Oct 14 16:26:07 2017
Good morning!

Enter directory: F:\1\2\
5.txt 9:12:19
6.txt 9:12:42

```

Тема 8: «Организация многопоточной обработки данных»

Цель: получить практические навыки по организации многопоточной обработки данных;

Задача 8:

Напишите программу, которая создает процесс. Используйте атрибуты по умолчанию. Родительский и вновь созданный процесс должны распечатать строки текста так, чтобы вывод родительского и дочернего процесса был синхронизован: сначала родительский процесс выводил бы три строки, затем дочерний пять строк, затем родительский следующие три строки и т.д. Используйте мьютексы.

Входные данные:

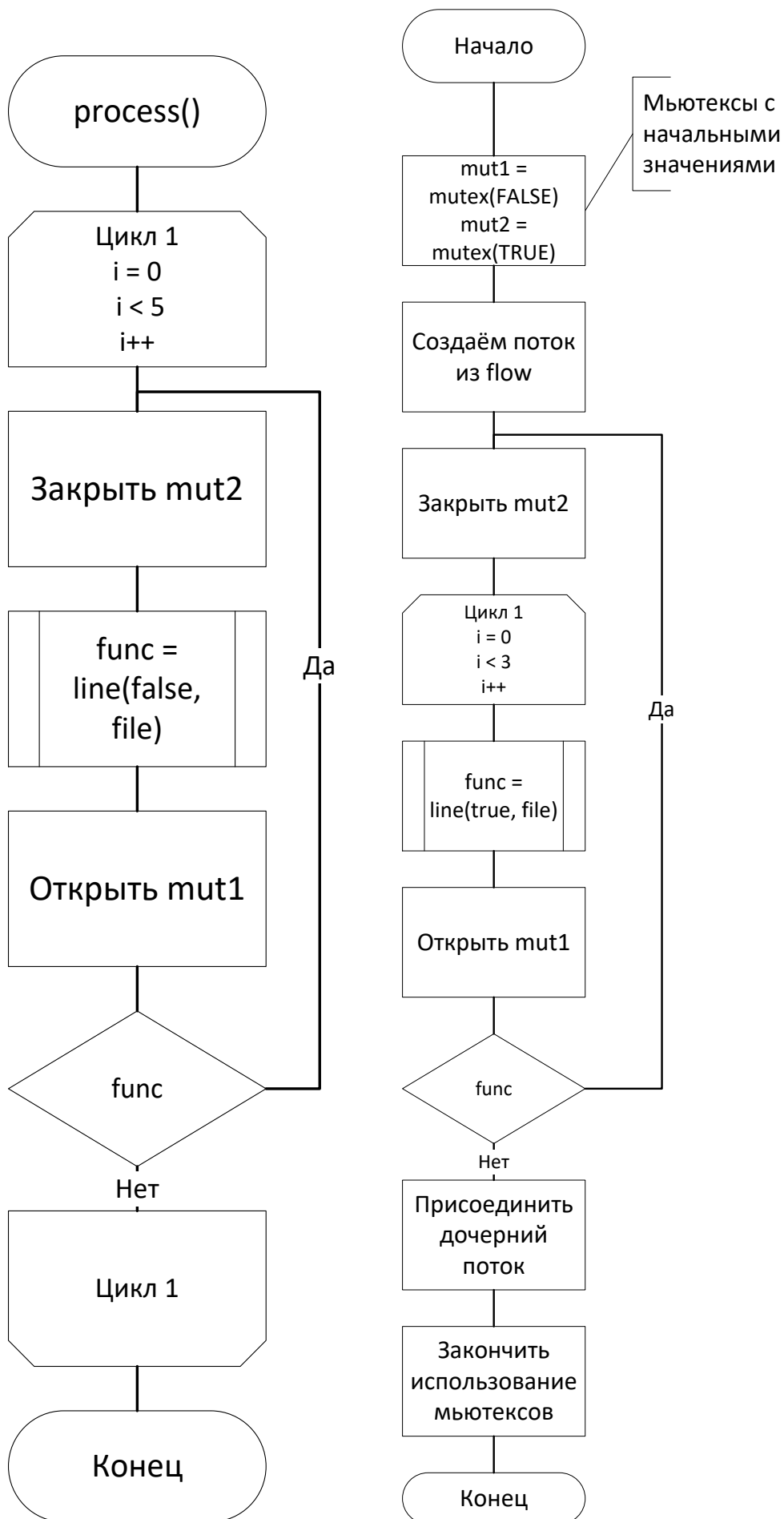
file – исходный файл строк для вывода их на экран;

Функциональные характеристики:

- 1 – открываем файл на чтение;
- 2 – создаём два процесса
- 3 – берём строки из файла;
- 4 – вызываем функцию печати и раскраски строк из файла;
- 5 – печатаем строки файла по очереди;

Выходные данные:

line – строки исходного файла;



```

#pragma once
#include <windows.h>
typedef HANDLE Mutex;

//mutex is locking mechanism
//Semaphore is signaling mechanism

DWORD WINAPI lock(Mutex sem)
{
    //DWORD WaitForSingleObject(
    // sem - идентификатор объекта
    //Идентификатор объекта идентифицирует процесс который нужно задержать...
    //Второй параметр означает насколько времени нужно задержать
    процесс(в миллисекундах) с момента выполнения данной функции... Либо вечно : INFINITE
    //Данная функция возвращает значение :
    //WAIT_OBJECT_0 в случае перехода "объекта" в сигнальное
    состояние(SetEvent(THandle)), при этом оставшийся интервал времени соответственно
    игнорируется;
    //WAIT_TIMEOUT в случае истечения заданного интервала времени, т.е.
    "объект" сигнал не получил...
    return WaitForSingleObject(sem, INFINITE);
}
DWORD WINAPI unlock(Mutex sem) //Семафор является одним из объектов синхронизации и
содержит счетчик, учитывающий количество потоков, обратившихся к данному ресурсу.
{
    return ReleaseMutex(sem); //После завершения работы с ресурсом поток
увеличивает значение счетчика
}
BOOL WINAPI join(HANDLE pThreadHandle)
{
    WaitForSingleObject(pThreadHandle, INFINITE);
    return CloseHandle(pThreadHandle); //закрывает дескриптор pThreadHandle открытого
объекта.
}
Mutex mutex(int c)
{
    return CreateMutex(NULL, c, (LPCWSTR)"Test");
}

```

```

#include <iostream>
#include <fstream>
#include <string>
#include <windows.h>
#include "lib.h"
using namespace std;
bool func;
Mutex mut1, mut2;
HANDLE console = GetStdHandle(STD_OUTPUT_HANDLE);
inline ostream& red(ostream &s)
{SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), FOREGROUND_INTENSITY |
FOREGROUND_RED | BACKGROUND_BLUE | BACKGROUND_GREEN | BACKGROUND_RED); return s;}
inline ostream& white(ostream &s)
{SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), BACKGROUND_BLUE |
BACKGROUND_GREEN | BACKGROUND_RED); return s;}
bool line(bool isred, ifstream &in) //выводит строки в опр цвете
{
    string line;
    if ( !getline(in, line)) //проверка на возможность считывания строки
        return false;
    if (isred)
    {SetConsoleTextAttribute(console, (WORD)(6)); cout << line << endl;} //зеленый
    else
    {SetConsoleTextAttribute(console, (WORD)(7)); cout << line << endl;} //серый
    return true;
}

```

```

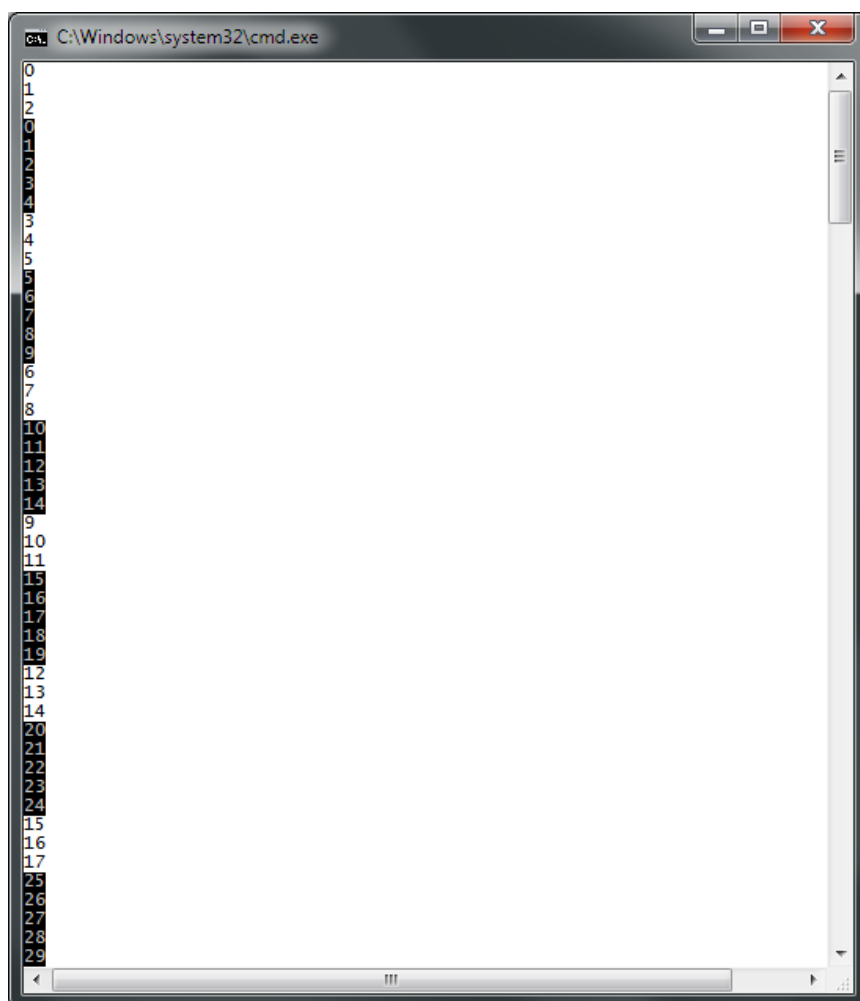
}
DWORD WINAPI process(LPVOID)
{
    ifstream in("1.txt");// разделяемый ресурс
    do
    {
        lock(mut1);//лок 2 мьютекса
        for (int i = 0; i < 5; i++)
            func = line(0, in);//вывод дочерних строк
        unlock(mut2);//разблок мьютекса 1
    }
    while (func);
    return 0;
}
int main()
{
    ifstream in("1.txt");// разделяемый ресурс
    if (!in.is_open())
    { cout << "Файл не найден!!!" << endl; }
    DWORD dwThreadId;
    mut1 = mutex(0);
    mut2 = mutex(1);
    HANDLE pThreadHandle = CreateThread(NULL, 0, process, NULL, 0,
&dwThreadId);//создание процесса
    do
    {
        lock(mut2); //лок 2 мьютекса
        for (int i = 0; i < 3; i++)
            func = line(1, in); //вывод родительского класса
        Sleep(300);
        unlock(mut1); //разблок мьютекса 1
    }
    while (func);
    join(pThreadHandle); //закрытие процесса
    CloseHandle(mut1);
    CloseHandle(mut2);
    return 0;
}

```

План тестирования:

№	Классы эквивалентности	Тестовый набор
1	Класс допустимых значений 29 цифр в файле	Ожидаемые ответы: 0120123434556789 и так далее до 29 <i>//программа заканчивает работу родительским процессом</i>
2	Количество значений в файле значений 29 цифр в файле	Ожидаемые ответы: 0120123434556789 и так далее до 29 <i>//программа заканчивает работу дочерним процессом</i>
3	Класс недопустимых значений: файл не существует	Ожидаемые ответы: файл не найден
4	Класс недопустимых значений: пустой файл	Ожидаемые ответы: файл пуст

Результаты тестирования:



A screenshot of a Windows command prompt window. The title bar shows the path "C:\Windows\system32\cmd.exe". The window contains a list of numbers from 0 to 29, arranged in a single column. The numbers are: 0, 1, 2, 0, 1, 2, 3, 4, 3, 4, 5, 5, 6, 7, 8, 9, 6, 7, 8, 10, 11, 12, 13, 14, 9, 10, 11, 15, 16, 17, 18, 19, 12, 13, 14, 20, 21, 22, 23, 24, 15, 16, 17, 25, 26, 27, 28, 29. The numbers are displayed in a monospaced font, and the window has a standard Windows XP-style interface with a scrollbar on the right.