МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2**

**по дисциплине  
 «АЛГОРИТМЫ И АНАЛИЗ СЛОЖНОСТИ»**

Выполнил студент группы 24/1                                       А. А. Иванов

Направление подготовки 02.03.03 Математическое обеспечение и администрирование информационных систем

Курс    2

Отчет принял доктор физико-математических наук, профессор                                                                                       А.И. Миков

Краснодар

2022 г.

**Задание**: создать функцию поиска буквы «А» в матрице и найти её сложность.

**Решение**: при поиске будем двигаться как программа - так как программа просматривает матрицу слева направо, сверху вниз, то и мы будем искать первый символ нашей буквы так-же. А последний символ нашей буквы будем искать так-же, но наоборот - будем идти с конца (справа налево, снизу вверх). После того, как мы нашли наши символы, смотрим их расположение в матрице и записываем его как координаты по осям Х и У. Если буква состоит из 1 символа - пишем, что буква отсутствует.

**Код программы:**

#include <iostream>

#include <ctime>

#include <cstdlib>

using namespace std;

struct tochka

{

int x;

int y;

};

void matrica(int\*\* matr, int n)

{

for (int i = 0; i < n; i++)

for (int j = 0; j < n; j++)

matr[i][j] = 0;

}

void image(int\*\* matr, int n)

{

tochka p1;

p1.x = rand() % (n - 6) + 3;

p1.y = rand() % (n - 3);

matr[p1.y][p1.x] = 1;

int lev = 0;

for (int i = p1.x; i >= 0; i--)

if (matr[i][p1.y] != 1)

lev++;

int prav = 0;

for (int i = p1.x; i < n; i++)

if (matr[i][p1.y] != 1)

prav++;

if (lev <= prav)

{

int p2 = rand() % lev;

int i1 = p1.y;

int j1 = p1.x;

for (int k = p2; k >= 0; k--)

{

if (i1 < n && j1 < n)

matr[i1][j1] = 1;

i1++;

j1++;

}

int i2 = p1.y;

int j2 = p1.x;

for (int k = p2; k >= 0; k--)

{

if (i2 < n && j2 < n)

matr[i2][j2] = 1;

i2++;

j2--;

}

int sered = p1.y + (p2 / 2) + 1;

int j\_perv = 0;

for (int j = 0; j < n; j++)

if (matr[sered][j] == 1)

j\_perv = j;

int j\_vtor = 0;

for (int j = (n-1); j >= 0; j--)

if (matr[sered][j] == 1)

j\_vtor = j;

for (int j = 0; j < n; j++)

if (j < j\_perv && j > j\_vtor)

matr[sered][j] = 1;

}

else

{

int p2 = rand() % prav;

int i1 = p1.y;

int j1 = p1.x;

for (int k = p2; k >= 0; k--)

{

if (i1 < n && j1 < n)

matr[i1][j1] = 1;

i1++;

j1++;

}

int i2 = p1.y;

int j2 = p1.x;

for (int k = p2; k >= 0; k--)

{

if (i2 < n && j2 < n)

matr[i2][j2] = 1;

i2++;

j2--;

}

int sered = p1.y + (p2 / 2) + 1;

int j\_perv = 0;

for (int j = 0; j < n; j++)

if (matr[sered][j] == 1)

j\_perv = j;

int j\_vtor = 0;

for (int j = (n - 1); j >= 0; j--)

if (matr[sered][j] == 1)

j\_vtor = j;

for (int j = 0; j < n; j++)

if (j < j\_perv && j > j\_vtor)

matr[sered][j] = 1;

}

}

void print(int\*\* matr, int n)

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

cout << matr[i][j] << "";

cout << endl;

}

cout << endl; cout << endl; cout << endl;

}

void poisk(int\*\* matr, int n)

{

tochka point1, point2;

int kolop = 0;

point1.x = 0;

point1.y = 0;

point2.x = 0;

point2.y = 0;

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

if (matr[i][j] == 1)

{

point1.x = j;

point1.y = i;

kolop += 3;

}

}

}

for (int i = (n - 1); i >= 0; i--)

{

for (int j = (n - 1); j >= 0; j--)

{

if (matr[i][j] == 1)

{

point2.x = j;

point2.y = i;

kolop += 3;

}

}

}

if (point1.x == point2.x && point1.y == point2.y)

{

kolop += 2;

cout << endl;

cout << "Буквы нет." << endl;

cout << "Колличество операций при поиске - " << kolop << endl;

cout << endl << endl << endl;

}

else

{

cout << endl;

cout << "Буква есть. Координаты: точка1 - " << point2.y << " " << point2.x << "; точка2 - " << point1.y << " " << point1.x << endl;

cout << "Колличество операций при поиске - " << kolop << endl;

cout << endl << endl << endl;

}

}

int main()

{

setlocale(LC\_ALL, "rus");

srand(time(NULL));

for (int n = 10; n < 101; n += 10)

{

int\*\* matr = new int\* [n];

for (int i = 0; i < n; i++)

matr[i] = new int[n];

matrica(matr, n);

image(matr, n);

print(matr, n);

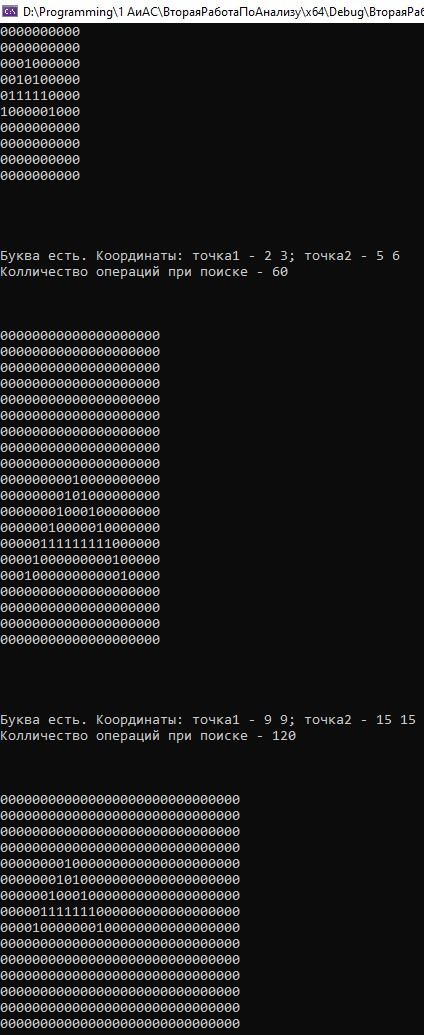
poisk(matr, n);

}

system("pause");

}

**Вывод в консоль:**

****

**Таблица со значениями:**

|  |  |
| --- | --- |
| Кол-во операций при поиске буквы (10 зыпусков) | |
| Размер матрицы | Кол-во операций (сред.) |
| 10 | 60 |
| 20 | 128 |
| 30 | 112 |
| 40 | 66,(6) |
| 50 | 136 |
| 60 | 312 |
| 70 | 252 |
| 80 | 120 |
| 90 | 268 |
| 100 | 244 |
| Вывод : зависимость от размера буквы | |

**График:**

**Вывод:** чем меньше буква - тем меньше операций требуется программе для её поиска (не зависит от размера матрицы).

**Система уравнений и коэффиценты:**

а = ; в = ; с =

**Средняя сложность:**

, где *n* > 0.