Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

«САРАТОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра информатики и программирования

**Отчет по программированию**

Отчет по программированию выполнил:

студент 1 курса 141 группы

по направлению «Математическое обеспечение и администрирование информационных систем»

факультета «Компьютерных наук и информационных технологий»

Черногоров Владислав Максимович

Проверил(а):

**СОДЕРЖАНИЕ**

1. Базовые элементы языка С++ 3
2. Функции в С++ 6
3. Операторы С++ 8
4. Рекуррентные соотношения 19
5. Вычисление конечных и бесконечных сумм и произведений 20
6. Массивы 26
7. Строки 38
8. Организация файлового ввода/вывода 44
9. Структуры 47

**Базовые элементы языка C++**

**Упражнение I**

Написать программу, которая вычисляет значение выражения:

**Задача 20**

**Код программы**

#include <iostream>

// Подключение заголовочного файла для поддержки операторов ввода/вывода

#include <cmath>

// Заголовочный файл, включающий в себя математические функции

// Объявление пространства имен std

using namespace std;

int main()

{

// Объявление переменных

double x, y;

// Вывод строки в консоль с помощью оператора cout

cout << "Type X and Y: ";

// Считывание элементов из консоли с помощью оператора cin

cin >> x >> y;

//log, abs и pow - математические функции

cout << "Solution: " << log(abs((y - pow(abs(x), 0.5))\*(x - y / (x + (x\*x) / 4)))) << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Type X and Y: 1 0 | Solution: 0 |
| Type X and Y: 2 5 | Solution: 0.178366 |

**Базовые элементы языка C++**

**Упражнение II**

Написать программу, которая подсчитывает:

**Задача 20**

Сумму членов геометрической прогрессии, если известен ее первый член, знаменатель и число членов прогрессии.

**Принцип решения**

Нам известны все параметры, чтобы подставить их в формулу для нахождения суммы геометрической прогрессии.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

double b1, q, n;

cout << "Type b1, devisor and count: ";

cin >> b1 >> q >> n;

cout << "Solution: " << (b1\*(pow(q, n) - 1) / (q - 1)) << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Type b1, devisor and count: 1 4 3 | Solution: 21 |

**Базовые элементы языка C++**

**Упражнение III**

Написать программу, которая определяет:

**Задача 20**

Является ли треугольник с длинами сторон a, b, c равносторонним.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

int a, b, c;

cout << "Type a, b, c: ";

cin >> a >> b >> c;

// Применение тернарной операции

cout << (a == b && b == c ? "YES" : "NO") << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Type a, b, c: 1 1 1 | YES |
| Type a, b, c: 3 4 2 | NO |

**Функции в С++**

**Задача 20**

Разработать функцию f(, которая вычисляет длину отрезка по координатам вершин ( и , и функцию t(a, b, c), которая проверяет, существует ли треугольник с длинами сторон a, b, c. С помощью данных функций проверить, сколько различных треугольников можно построить по четырем заданным точкам на плоскости.

**Принцип решения**

В функции f находим длину отрезка между двумя координатами, а в функции t проверяем могут ли три отрезка соединенных тремя вершинами образовывать треугольник.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

// Объявление функции

double f(int x1, int y1, int x2, int y2)

{

return sqrt((x1 - x2)\*(x1 - x2)\*1.0 + (y1 - y2)\*(y1 - y2)\*1.0);

}

bool t(double a, double b, double c)

{

if (a + b <= c || a + c <= b || b + c <= a)

return false;

else return true;

}

int main()

{

int x1, x2, y1, y2, x3, y3, x4, y4, sum = 0;

cout << "Type 1 coordinates: ";

cin >> x1 >> y1;

cout << "Type 2 coordinates: ";

cin >> x2 >> y2;

cout << "Type 3 coordinates: ";

cin >> x3 >> y3;

cout << "Type 4 coordinates: ";

cin >> x4 >> y4;

// Вызов функций

if (t(f(x1, y1, x2, y2), f(x1, y1, x3, y3), f(x2, y2, x3, y3))) ++sum;

if (t(f(x1, y1, x3, y3), f(x1, y1, x4, y4), f(x4, y4, x3, y3))) ++sum;

if (t(f(x2, y2, x3, y3), f(x2, y2, x4, y4), f(x3, y3, x4, y4))) ++sum;

if (t(f(x4, y4, x1, y1), f(x4, y4, x2, y2), f(x2, y2, x1, y1))) ++sum;

cout << sum << endl;

}

**Функции в С++**

++sum;

if (t(f(x1, y1, x3, y3), f(x1, y1, x4, y4), f(x4, y4, x3, y3)))

++sum;

if (t(f(x2, y2, x3, y3), f(x2, y2, x4, y4), f(x3, y3, x4, y4)))

++sum;

if (t(f(x4, y4, x1, y1), f(x4, y4, x2, y2), f(x2, y2, x1, y1)))

++sum;

cout << sum << endl;

}

**Пример(ы)**

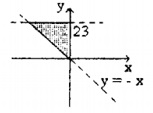
|  |  |
| --- | --- |
| Ввод | Вывод |
| Type 1 coordinates: 3 4  Type 1 coordinates: 2 1  Type 1 coordinates: 3 2  Type 1 coordinates: 1 4 | 4 |
| Type 1 coordinates: 1 1  Type 1 coordinates: -1 1  Type 1 coordinates: -2 1  Type 1 coordinates: 0 -2 | 3 |

**Операторы C++**

**Упражнение I**

Дана точка на плоскости с координатами (x, y). Составить программу, которая выдает одно из сообщений «Да», «Нет», «На границе» в зависимости от того, лежит ли точка внутри заштрихованной области, вне заштрихованной области или на ее границе. Области задаются графически следующим образом.

**Задача 20**



**Принцип решения**

Проверяем, удовлетворяет ли точка с указанными координатами всем трем условиям нахождения в этой области. Если точка удовлетворяет только одному или двум условиям, значит она лежит на границе. В противном случае точка не принадлежит области.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

double x1, y1;

cout << "Type coordinates: ";

cin >> x1 >> y1;

// Применение логических операций

if (-23 <= x1 && x1 <= 0 && 0 <= y1 && y1 <= 23 && (y1\*1.0 / x1 <= -1.0 || x1 == 0))

if (x1 == 0 || y1 == 23 || y1\*1.0 / x1 == -1.0)

cout << "On the border" << endl;

else cout << "Yes" << endl;

else cout << "No" << endl;

}

**Операторы C++**

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Type a, b, c: -6 8 | Yes |
| Type a, b, c: -5 5 | On the border |

**Операторы C++**

**Упражнение II**

Составить программу.

**Задача 5**

Дан номер карты k (6 ≤ k ≤ 14), определить достоинство карты. Определить полное название соответствующей карты в виде «дама пик», «шестерка бубен» и т.д.

*Замечание.* Названия всех нечисленных карт заменять на «картинка».

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

// Включение распознавания русских символов

setlocale(LC\_ALL, "Russian");

int n, m;

cin >> n >> m;

// Вывод зависит от переменной m

switch (m)

{

case 6: cout << "Шестерка "; break;

case 7: cout << "Семерка "; break;

case 8: cout << "Восьмерка "; break;

case 9: cout << "Девятка "; break;

case 10: cout << "Десятка "; break;

case 11: case 12: case 13:

case 14: cout << "Картинка "; break;

default: cout << "ERROR "; break;

}

// Вывод зависит от переменной n

switch (n)

{

case 1: cout << "пик" << endl; break;

case 2: cout << "бубен" << endl; break;

case 3: cout << "треф" << endl; break;

case 4: cout << "червей" << endl; break;

default: cout << "ERROR" << endl; break;

}

}

**Операторы C++**

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| 3 12 | Картинка треф |

**Операторы C++**

**Упражнение III**

Вывести на экран:

*Замечание.* Решите каждую задачу тремя способами: использую операторы цикла while, do while и for.

**Задача 20**

Все трехзначные числа, в которых хотя бы две цифры повторяются.

*Замечание.* В циклах while решить задачу в определенных границах, заданных пользователем.

**Принцип решения**

Проходимся по циклу от 100 до 1000 и проверяем, удовлетворяет ли текущее значения нашему условию. В while эти границы обозначаются вводимыми значениями a и b.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

for (int i = 100; i<1000; i++)

{

if ((i % 10 == i % 100 / 10 || i % 10 == i / 100 || i % 100 / 10 == i / 100) && (i / 100>0 && i / 100<10))

cout << i << ' ';

}

cout << endl << "----------------" << endl;

{

int a, b;

cin >> a >> b;

int i = a;

while (i <= b)

{

if ((i % 10 == i % 100 / 10 || i % 10 == i / 100 || i % 100 / 10 == i / 100) && (i / 100>0 && i / 100<10))

cout << i << ' ';

i++;

}

}

cout << endl << "----------------" << endl;

{

int a, b;

cin >> a >> b;

int i = a;

do

{

if ((i % 10 == i % 100 / 10 || i % 10 == i / 100 || i % 100 / 10 == i / 100) && (i / 100>0 && i / 100<10)) cout << i << ' ';

i++;

} while (i <= b);

}

}

**Операторы C++**

cout << endl << "----------------" << endl;

{

int a, b;

cin >> a >> b;

int i = a;

do

{

if ((i % 10 == i % 100 / 10 || i % 10 == i / 100 || i % 100 / 10 == i / 100) && (i / 100>0 && i / 100<10))

cout << i << ' ';

i++;

} while (i <= b);

}

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| 200 200 | 200 |
| -50 110 | 100 101 110 |

**Операторы C++**

**Упражнение IV**

Вывести на экран числа в виду следующей таблицы:

**Задача 20**

2 2 2 2 2

3 4 5 6 7

2 2 2 2

2 3 4 5

2 2 2

1 2 3

2 2

0 1

2

-1

**Принцип решения**

Создаем в цикле длинною 5 два других цикла, которые будут выполнятся до значения, зависимого от шага основого цикла.

**Код программы**

#include <iostream>

#include <cmath>

using namespace std;

int main()

{

for (int i = 5; i >= 1; i--)

{

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

cout << 2 << ' ';

cout << endl;

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

cout << j + i - 2 << ' ';

cout << endl;

}

}

**Операторы C++**

**Упражнение V**

Постройте таблицу значений функции y = f(x) для x [a, b] с шагом h. Если в некоторой точке x функция не определена, то выведите на экран сообщение об этом.

*Замечание.* Для решения задачи использовать вспомогательную функцию.

**Задача 20**

**Принцип решения**

Создаем функцию f, вычисляющую значение функции в точке. Затем в main создаем цикл, длинную от a до b с шагом h, который будет применять эту функцию к каждому x.

**Код программы**

#include <iostream>

// Этот заголовочный файл включает в себя функции форматирования вывода

#include <iomanip>

#include <cmath>

using namespace std;

// Объявление глобальной константы

const double PO = 1e-12;

// Передача параметра y по ссылке

void f(double x, double &y)

{

if (abs(x)<1 - PO && abs(x)>-1 + PO)

{

y = 1 / (x + 7) + log(1 - abs(x));

// Выделение знаков после запятой

cout << setprecision(3) << fixed << x << ": " << setprecision(20) << fixed << y << endl;

}

else

cout << setprecision(3) << fixed << x << ": " << "NONE" << endl;

}

int main()

{

double y;

double a, b, h;

cout << "Type a,b,h ([a,b]; h- step): ";

cin >> a >> b >> h;

for (double x = a; x <= b + PO; x += h)

{

f(x, y);

}

cout << endl;

}

**Операторы C++**

double y;

double a, b, h;

cout << "Type a,b,h ([a,b]; h- step): ";

cin >> a >> b >> h;

for (double x = a; x <= b + PO; x += h)

{

f(x, y);

}

cout << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Type a,b,h ([a,b]; h- step): -0.5 0.1 0.05 | -0.500: -0.53930102671379143000  -0.450: -0.44516524503042953000  -0.400: -0.35931047225083923000  -0.350: -0.28040697624283034000  -0.300: -0.20742121259544888000  -0.250: -0.13953392430363276000  -0.200: -0.07608472778479807900  -0.150: -0.01653352803792104700  -0.100: 0.03956702057405765700  -0.050: 0.09259159769878035100  -0.000: 0.14285714285714274000  0.050: 0.09055067724365523100  0.100: 0.03548455476470893700 |
| Type a,b,h ([a,b]; h- step): 0.99 1.01 0.01 | 0.990: -4.48001374043114440000  1.000: NONE  1.010: NONE |

**Операторы C++**

**Упражнение VI**

Постройте таблицу значений функции y = f(x) для x [a, b] с шагом h.

*Замечание.* Для решения задачи использовать вспомогательную функцию.

**Задача 20**

**Код программы**

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

const double PI = 3.14159265359;

const double PO = 1e-12;

void f(double i)

{

cout << setprecision(3) << fixed << i << ": ";

if (PI / 2 - PO <= abs(i) && abs(i) <= PI + PO)

cout << setprecision(15) << fixed << cos(i) << endl;

else if (abs(i)>PI + PO)

cout << 0 << endl;

else if (abs(i) < PI / 2 - PO)

cout << setprecision(15) << fixed << sin(i) << endl;

else

cout << "ERROR" << endl;

}

int main()

{

double a, b, h;

cout << "Enter range and step: ";

cin >> a >> b >> h;

for (double i = a; i <= b + PO; i += h)

{

f(i);

}

}

**Операторы C++**

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Enter range and step: 0 3 0.2 | 0.000: 0.000000000000000  0.200: 0.198669330795061  0.400: 0.389418342308651  0.600: 0.564642473395035  0.800: 0.717356090899523  1.000: 0.841470984807897  1.200: 0.932039085967226  1.400: 0.985449729988460  1.600: -0.029199522301289  1.800: -0.227202094693087  2.000: -0.416146836547142  2.200: -0.588501117255345  2.400: -0.737393715541245  2.600: -0.856888753368947  2.800: -0.942222340668658  3.000: -0.989992496600446 |
| Enter range and step: -0.1 0.1 0.1 | -0.100: -0.099833416646828  0.000: 0.000000000000000  0.100: 0.099833416646828 |

**Рекуррентные соотношения**

**Упражнение**

Написать программу, вычисляющую первые n элементов заданной последовательности:

**Задача 20**

, , ,

**Принцип решения**

В начале сразу выводим и , так как они нам известны, а затем заходим в цикл и после каждого вывода проверяем, сколько чисел мы уже вывели. Для этого создаем условие выхода внутри цикла.

**Код программы**

#include <iostream>

#include <iomanip>

#include <cmath>

using namespace std;

const double PI = 3.14159265359;

const double PO = 1e-12;

int main()

{

int b1 = 1, b2 = 5;

int n;

cout << "Enter n: ";

cin >> n;

if (n >= 1)

{

cout << b1 << ' ';

if (n >= 2)

cout << b2 << ' ';

}

for (int i = 2; i<n; i++)

{

b1 = b2 - b1;

cout << b1 << ' ';

i++;

if (i == n) break;

b2 = b1 + b2;

cout << b2 << ' ';

}

}

**Вычисление конечных и бесконечных сумм и произведений**

**Упражнение I**

Для данного натурального n и действительного x подсчитать следующую сумму:

**Задача 20**

S = sin x + sin sin x + sin sin sin x + … + sin sin sin … sin x;

**Код программы**

// Убраны лишние заголовочные файлы и константы

#include <iostream>

using namespace std;

const double PI = 3.14159265359;

const double PO = 1e-12;

int main()

{

double x, sum = 0;

int n;

cout << "Enter n and x: ";

cin >> n >> x;

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

{

x = sin(x);

sum += x;

}

cout << "Sum = " << sum << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Enter n and x: 3 6.1 | Sum = -0.543487 |

**Вычисление конечных и бесконечных сумм и произведений**

**Упражнение II**

Для заданного натурального k и действительного x подсчитать следующее выражение:

**Задача 20**

**Код программы**

#include <iostream>

using namespace std;

int main()

{

double x, op = 1;

int k;

cout << "Enter k and x: ";

cin >> k >> x;

double xp = x;

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

{

xp = 1 + (pow(x, 2 \* n)\*pow(-1.0, n - 1)) / ((double)(n + 2)\*(n + 1));

op \*= xp;

}

cout << "Op = " << op << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Enter k and x: 5 1 | Op = 0.555666 |
| Enter k and x: 0 246261 | Op = 0.5 |

**Вычисление конечных и бесконечных сумм и произведений**

**Упражнение III**

Вычислить бесконечную сумму ряда с заданной точностью e (e>0)

**Задача 20**

*Замечение. Вывести перед суммой все слагаемые.*

**Код программы**

#include <iostream>

#include <iomanip>

using namespace std;

int main()

{

double e;

double result = 0;

double sqrt3 = 1 / sqrt(3.0);

cout << setprecision(10);

cout << "Enter the precision: ";

cin >> e;

while (sqrt3 >= e)

{

result += sqrt3;

sqrt3 \*= 1 / sqrt(3.0);

cout << sqrt3 << endl;

}

cout << "Result with the precision " << e << ": " << result << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Enter the precision: 0.1 | 0.3333333333  0.1924500897  0.1111111111  0.06415002991  Result with the precision 0.1: 1.214244803 |

**Вычисление конечных и бесконечных сумм и произведений**

**Упражнение IV**

Вычислить и вывести на экран значение функции F(x) на отрезке [a,b] с шагом h = 0.1 и точностью e. Результат работы программы представить в виде следующей таблицы:

|  |  |  |  |
| --- | --- | --- | --- |
| № | Значение x | Значение функции F(x) | Количество слагаемых n |
| 1 |  |  |  |
| 2 |  |  |  |
| … |  |  |  |

*Замечание.* При решении задачи использовать вспомогательную функцию.

**Задача 20**

**Принцип решения**

Создаем функцию func1ex1 с передачей значения opCnt по ссылке, где opCnt – количество слагаемых. Переменная last является следующим слагаемым нашей суммы, result – это сама сумма. В цикле функции происходит рекуррентное изменение нашего следующего слагаемого и суммы, а также подсчет шагов цикла, пока не достигнется нужная точность.

В main мы сначала запрашиваем пользователя указать точность e, затем выводим шапку таблицы (первую строку). В цикле выполняем нашу функцию, после чего выводим нужные результаты.

**Код программы**

#include <iostream>

#include <iomanip>

using namespace std;

// Заменение строки pi на константу 3.14159265359

const double lu = 1e-12;

const double pi = 3.14159265359;

// Если вещественное число близко к нулю на 12 и больше знаков, то приравнивает его к 0

#define if\_lu(n) (n <= lu && n > 0.0 || n >= lu\*(-1.0) && n < 0.0) ? 0.0 : n

double func1ex1(double i, double e, int &opCnt)

{

double last = i;

double result = i;

for (int j = 1; abs(last) > e; j++)

{

last \*= (i \* i \* (j \* 2 - 1)) / ((j \* 2 + 1) \* (j \* 2));

result += last;

opCnt++;

}

return result;

}

int main()

{

setlocale(LC\_ALL, "Russian");

double a = -0.9, b = 0.9;

double h = 0.1;

double e;

int j = 1;

cout << "Enter the precision: ";

cin >> e;

// setiosglags(ios::left) - выравнивание по левому краю

// setw(n) - создание поля, в которой будет производиться вывод

cout << setprecision(10) << setiosflags(ios::left) << setw(6) << "№№: " << setw(12) << "Значение x" << setw(24) << "Значение функции F(x)" << setw(21) << "Кол-во слагаемых n" << endl;

for (double i = a; i <= b; i += h, j++)

{

int opCnt = 1;

double func\_res = func1ex1(i, e, opCnt);

cout << setw(2) << j << setw(4) << ": " << setw(12) << (if\_lu(i)) << setw(24) << pi / 2 - (if\_lu(func\_res)) << setw(21) << opCnt << endl;

}

}

**Вычисление конечных и бесконечных сумм и произведений**

double func1ex1(double i, double e, int &opCnt)

{

double last = i;

double result = i;

for (int j = 1; abs(last) > e; j++)

{

last \*= (i \* i \* (j \* 2 - 1)) / ((j \* 2 + 1) \* (j \* 2));

result += last;

opCnt++;

}

return result;

}

int main()

{

setlocale(LC\_ALL, "Russian");

double a = -0.9, b = 0.9;

double h = 0.1;

double e;

int j = 1;

cout << "Enter the precision: ";

cin >> e;

// setiosglags(ios::left) - выравнивание по левому краю

// setw(n) - создание поля, в которой будет производиться вывод

cout << setprecision(10) << setiosflags(ios::left) << setw(6) << "№№: " << setw(12) << "Значение x" << setw(24) << "Значение функции F(x)" << setw(21) << "Кол-во слагаемых n" << endl;

for (double i = a; i <= b; i += h, j++)

{

int opCnt = 1;

double func\_res = func1ex1(i, e, opCnt);

cout << setw(2) << j << setw(4) << ": " << setw(12) << (if\_lu(i)) << setw(24) << pi / 2 - (if\_lu(func\_res)) << setw(21) << opCnt << endl;

}

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Enter the precision: 0.001 |
| Вывод | |
| №№: Значение x Значение функции F(x) Кол-во слагаемых n  1 : -0.9 2.60859418 5  2 : -0.8 2.464945813 4  3 : -0.7 2.332409846 4  4 : -0.6 2.208823641 4  5 : -0.5 2.09241091 3  6 : -0.4 1.981718993 3  7 : -0.3 1.875357077 3  8 : -0.2 1.77213766 3  9 : -0.1 1.670962993 2  10: 0 1.570796327 1  11: 0.1 1.47062966 2  12: 0.2 1.369454993 3  13: 0.3 1.266235577 3  14: 0.4 1.15987366 3  15: 0.5 1.049181743 3  16: 0.6 0.9327690125 4  17: 0.7 0.809182808 4  18: 0.8 0.6766468411 4  19: 0.9 0.5329984733 5 | |

**Массивы**

**Упражнение I**

Дана последовательность целых чисел.

*Замечание.* Задачи из данного пункта решить двумя способами: используя одномерный массив, а затем двумерный.

**Задача 20**

Вывести на экран положительные элементы с нечетными индексами (для двумерного массива – первый индекс должен быть нечетным).

*Замечание*. Индексация начинается с 0.

**Принцип решения**

Если у нас одномерный массив, то создаем цикл от 1 до размера массива с шагом 2, чтобы вывести все элементы на нечетных позициях. В двумерном массиве делаем тоже самое, но при этом пропускаем четные строки с помощью такого-же цикла.

**Код программы**

#include <iostream>

using namespace std;

// Создание и считывание переменной в укороченной записи

#define ini(type, n) \

type n; \

cin >> n;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

int main()

{

ini(int, n);

ini(int, m);

// Инициализация массива

int mas1[100];

// Инициализация двумерного массива

int mas2[100][100];

if (m == 1)

{

forn(i, n) cin >> mas1[i];

for (int i = 1; i < n; i += 2)

cout << mas1[i] << ' ';

cout << endl;

}

else

{

forn(i, n)

forn(j, m)

cin >> mas2[i][j];

for (int i = 1; i < n; i += 2)

{

for (int j = 1; j < m; j += 2)

{

cout << mas2[i][j] << ' ';

}

cout << endl;

}

}

}

**Массивы**

else

{

forn(i, n)

forn(j, m)

cin >> mas2[i][j];

for (int i = 1; i < n; i += 2)

{

for (int j = 1; j < m; j += 2)

{

cout << mas2[i][j] << ' ';

}

cout << endl;

}

}

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| 6 1  1 2 3 4 5 6 | 2 4 6 |
| Enter k and x: 0 246261 | Op = 0.5 |

**Массивы**

**Упражнение II**

Дана последовательность из n действительных чисел.

*Замечание.* Задачи из данного пункта решить, используя одномерный массив.

**Задача 20**

Найти количество пар соседних элементов, в которых предыдущий элемент меньше последующего.

*Замечание*. Использовать считывание из файлового потока.

**Принцип решения**

Проверяем каждую пару соседних элементов, и если она удовлетворяет нашим условиям, то прибавляем счетчик result и в конце выводим его в файл.

**Код программы**

#include <iostream>

// Заголовочный файл fstream позволяет производить действия над файлами

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

// Определение потока для входных данных

ifstream in("input.txt");

// Определение потока для выходных данных

ofstream out("output.txt");

int main()

{

double mas2[1000];

int a;

int result = 0;

// Считывание с файла

in >> a;

forn(i, a)

in >> mas2[i];

forn(i, a - 1)

if (mas2[i] < mas2[i + 1])

result++;

// Вывод в файл

out << result;

if (result <= a && result >= 0)

cout << "Success!";

// Закрываем файлы

in.close();

out.close();

}

**Массивы**

if (result <= a && result >= 0)

cout << "Success!";

// Закрываем файлы

in.close();

out.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| input.txt | output.txt |
| 5  2 5 2 3 0 | 2 |
| 1  0 | 0 |

**Массивы**

**Упражнение III**

Дан массив размером n×n (если не оговорено иначе), элементы которого целые числа.

**Задача 20**

В каждом столбце найти минимум и заменить его нулем.

*Замечание*. Использовать динамический массив и считывание из файлового потока.

**Принцип решения**

Проходимся по каждому столбцу массива и находим в нем минимальный элемент, после чего еще раз проходимся по нему и заменяем все элементы равные минимуму на 0.

**Код программы**

#include <iostream>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

ofstream out("outfile.txt");

int main()

{

int a;

in >> a;

// Объявление динамического массива и выделение памяти для первого столбца в нем

int \*\*mas3 = new int\*[a];

forn(i, a)

{

// Выделение памяти под новую строку в массиве

mas3[i] = new int[a];

forn(j, a) in >> mas3[i][j];

}

bool sthHappened = false;

forn(j, a)

{

int minj = mas3[0][j];

int countMin = 0;

forn(i, a)

{

if (minj > mas3[i][j])

{

minj = mas3[i][j];

countMin = 1;

}

else if (minj == mas3[i][j])

countMin++;

}

if (countMin != a)

{

sthHappened = true;

forn(i, a) if (minj == mas3[i][j]) mas3[i][j] = 0;

}

}

if (sthHappened) cout << "Success!";

else cout << "Nothing changed.";

forn(i, a)

{

forn(j, a) out << mas3[i][j] << ' ';

out << endl;

}

// Освобождение памяти, занимаемой массивом

delete[] mas3;

in.close();

out.close();

}

**Массивы**

{

minj = mas3[i][j];

countMin = 1;

}

else if (minj == mas3[i][j])

countMin++;

}

if (countMin != a)

{

sthHappened = true;

forn(i, a) if (minj == mas3[i][j]) mas3[i][j] = 0;

}

}

if (sthHappened) cout << "Success!";

else cout << "Nothing changed.";

forn(i, a)

{

forn(j, a) out << mas3[i][j] << ' ';

out << endl;

}

// Освобождение памяти, занимаемой массивом

delete[] mas3;

in.close();

out.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | outfile.txt |
| 4  4 2 3 1  0 2 -3 5  9 9 9 9  1 4 0 2 | 4 0 3 0  0 0 0 5  9 9 9 9  1 4 0 2 |

**Массивы**

**Упражнение IV**

Дан массив размером n×n (если не оговорено иначе), элементы которого целые числа.

**Задача 20**

Для каждого столбца найти номер первой пары одинаковых элементов. Данные записать в новый массив.

*Замечание*. Использовать динамический массив и считывание из файлового потока.

**Принцип решения**

Проверяем каждый элемент в столбце на наличие похожей пары. Если пара найдена, то выходим из циклов и переходим к следующему столбцу. Таким образом мы находим первую пару похожих элементов и выводим их индексы в массиве. Если в столце пар не найдено, то выводим X.

**Код программы**

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

ofstream out("outfile.txt");

int main()

{

int a;

in >> a;

int \*\*mas3 = new int\*[a];

int \*\*mas3\_1 = new int\*[a];

forn(i, a)

{

mas3\_1[i] = new int[2];

mas3[i] = new int[a];

forn(j, a) in >> mas3[i][j];

}

bool sthHappened = false;

forn(j, a)

{

forn(i, a)

{

bool check = false;

for (int k = i + 1; k < a; k++)

{

if (mas3[k][j] == mas3[i][j])

{

sthHappened = true;

mas3\_1[j][0] = i;

mas3\_1[j][1] = k;

check = true;

break;

}

}

if (check)

break;

}

}

out << setiosflags(ios::left) << setw(16) << "First element: ";

forn(i, a)

{

if (mas3\_1[i][0] >= 0) out << mas3\_1[i][0] << ' ';

else out << "X ";

}

out << endl << setw(16) << "Second element: ";

forn(i, a)

{

if (mas3\_1[i][1] >= 0) out << mas3\_1[i][1] << ' ';

else out << "X ";

}

if (sthHappened) cout << "We found something.";

else cout << "Pairs not found.";

delete[] mas3\_1;

delete[] mas3;

in.close();

out.close();

}

**Массивы**

for (int k = i + 1; k < a; k++)

{

if (mas3[k][j] == mas3[i][j])

{

sthHappened = true;

mas3\_1[j][0] = i;

mas3\_1[j][1] = k;

check = true;

break;

}

}

if (check)

break;

}

}

out << setiosflags(ios::left) << setw(16) << "First element: ";

forn(i, a)

{

if (mas3\_1[i][0] >= 0) out << mas3\_1[i][0] << ' ';

else out << "X ";

}

out << endl << setw(16) << "Second element: ";

forn(i, a)

{

if (mas3\_1[i][1] >= 0) out << mas3\_1[i][1] << ' ';

else out << "X ";

}

if (sthHappened) cout << "We found something.";

else cout << "Pairs not found.";

delete[] mas3\_1;

delete[] mas3;

in.close();

out.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | outfile.txt |
| 4  1 1 3 9  0 2 -3 9  9 1 9 9  1 2 0 9 | First element: 0 0 X 0  Second element: 3 2 X 1 |

**Массивы**

**Упражнение V**

В одномерном массиве, элементы которого – целые числа, произвести следующие действия:

**Задача 5**

Удалить из массива элементы с номера k1 по номер k2.

*Замечание*. Использовать динамический массив и считывание из файлового потока.

**Принцип решения**

Все элементы массива, которые находятся слева от отрезка удаления оставляем, а те, что справа записываем в начало этого отрезка, перекрывая удаляемые элементы. Затем выводим элементов получившегося массива.

**Код программы**

#include <iostream>

#include <fstream>

// Эта библиотека содержит в себе функции для работы с несколькими переменными

#include <algorithm>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

ofstream out("outfile.txt");

int main()

{

int a, k1, k2;

in >> a >> k1 >> k2;

int \*mas1 = new int[a];

if (k1 < 0 || k2 < 0 || k2 >= a || k1 >= a)

{

cout << "Error";

return 0;

}

forn(i, a)

in >> mas1[i];

int g = 0;

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

{

// Функции min и max возвращают минимальный и максимальный элемент соответственно

if ((i < min(k1, k2) && i >= 0) || (i>max(k1, k2) && i < a))

{

mas1[g] = mas1[i];

g++;

}

}

a -= abs(k2 - k1) + 1;

forn(i, a)

out << mas1[i] << ' ';

delete[]mas1;

return 0;

}

**Массивы**

// Функции min и max возвращают минимальный и максимальный элемент соответственно

if ((i < min(k1, k2) && i >= 0) || (i>max(k1, k2) && i < a))

{

mas1[g] = mas1[i];

g++;

}

}

a -= abs(k2 - k1) + 1;

forn(i, a)

out << mas1[i] << ' ';

delete[]mas1;

return 0;

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | outfile.txt |
| 7 2 5  1 2 3 4 5 6 7 | 1 2 7 |

**Массивы**

**Упражнение VI**

В двумерном массиве, элементы которого – целые числа, произвести следующие действия:

**Задача 5**

Вставить строку из нулей после всех строк, в которых нет ни одного нуля.

**Принцип решения**

Проверяем строку на наличе нуля. Если 0 не найден, то сдвигаем все строчки ниже этой и вставляем новую, заполненную нулями.

**Код программы**

#include <iostream>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

ofstream out("outfile.txt");

int main()

{

int a, b;

bool nonull = true;

bool sthHappened = false;

in >> a >> b;

int \*\*mas3 = new int \*[2 \* a];

forn(i, a)

{

mas3[i] = new int[b];

forn(j, b)

{

in >> mas3[i][j];

}

}

int \*masSPEC = new int[b];

forn(i, b)

masSPEC[i] = 0;

int linesAdded = 0;

for (int i = 0; i < a + linesAdded; i++)

{

forn(j, b)

{

if (mas3[i][j] == 0)

nonull = false;

}

if (nonull)

{

sthHappened = true;

linesAdded++;

i++;

for (int k = a + linesAdded - 1; k > i; k--)

mas3[k] = mas3[k - 1];

mas3[i] = masSPEC;

}

nonull = true;

}

if (sthHappened)

cout << "Something happened...";

else

cout << "Nothing happened.";

forn(i, a + linesAdded)

{

forn(j, b)

{

out << mas3[i][j] << ' ';

}

out << endl;

}

delete[]mas3;

in.close();

out.close();

}

**Массивы**

nonull = false;

}

if (nonull)

{

sthHappened = true;

linesAdded++;

i++;

for (int k = a + linesAdded - 1; k > i; k--)

mas3[k] = mas3[k - 1];

mas3[i] = masSPEC;

}

nonull = true;

}

if (sthHappened)

cout << "Something happened...";

else

cout << "Nothing happened.";

forn(i, a + linesAdded)

{

forn(j, b)

{

out << mas3[i][j] << ' ';

}

out << endl;

}

delete[]mas3;

in.close();

out.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | outfile.txt |
| 3 4  1 3 0 1  0 0 0 0  2 3 5 2 | 1 3 0 1  0 0 0 0  2 3 5 2  0 0 0 0 |

**Строки**

**Упражнение II**

Простые действия со строками.

**Задача 1**

В данной строке вставить символ c1 после каждого вхождения символа с2.

**Код программы**

#include <iostream>

// Подключение заголовочного файла, включающего в себя функции для работы со строками

#include <string>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

#define ini(type, n) \

type n; \

cin >> n;

int main() {

cout << "Enter the string: ";

string s;

// Считывание строки до знака переноса

getline(cin, s);

cout << "Enter the sensor: ";

ini(char, c1);

cout << "Enter the insert: ";

ini(char, c2);

forn(i, s.size())

if (s[i] == c1) {

// Вставка элемента в строку

s.insert(s.begin() + i + 1, c2);

i++;

}

cout << "Result: " << s << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Enter the string: 142df. dvsd  Enter the sensor: d  Enter the insert: 0 | Result: 142d0f. d0vsd0 |

**Строки**

**Упражнение III**

Сложные действия со строками.

*Замечание*. Дано осмысленное текстовое сообщение (т.е. алфавитно-цифровая информация, разделенная проблеами и знаками препинания, в конце которого ставится точка.

**Задача 20**

Удалить из сообщения все повторяющиеся слова.

**Принцип решения**

При считывании заносим каждое слово, разделяющееся пробелами или знаками препинания, в массив и копию этого слова, но со всеми строчными символами в другой массив. Таким образом при поиске совпадений будут найдены одинаковые слова с разным регистром и удалены из строки. После чего выполняется удаление лишних пробелов в тексте.

**Код программы**

#include <iostream>

#include <string>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

#define ini(type, n) \

type n; \

cin >> n;

string clear\_spaces(string s)

{

if (s[0] == ' ' && s.find\_first\_not\_of(' ') < s.size())

// Удаление части строки

s.erase(s.begin(), s.begin() + s.find\_first\_not\_of(' '));

if (s[s.size() - 1] == ' ' && s.find\_last\_not\_of(' ') < s.size())

s.erase(s.begin() + s.find\_last\_not\_of(' ') + 1, s.end());

if (s.find\_first\_not\_of(' ') >= s.size() && s.find\_last\_not\_of(' ') >= s.size())

return "";

forn(i, s.size() - 1)

{

if (s[i] == s[i + 1] && s[i] == ' ')

{

s.erase(s.begin() + i);

i--;

}

}

forn(i, s.size() - 1)

{

if (s[i] == ' ' && ispunct(s[i + 1]))

s.erase(s.begin() + i);

}

return s;

}

int main()

{

string mas1[100];

string real\_mas[100];

string s;

getline(cin, s, '.');

int cini = 0;

int am = 0;

while (cini < s.size())

{

// Провреяем, является ли символ пробелом или знаком-разделителем

if (ispunct(s[cini]) || (isspace(s[cini])))

{

if (cini > 0 && !(ispunct(s[cini - 1]) || (isspace(s[cini - 1]))))

am++;

}

else

{

real\_mas[am] += s[cini];

// Меняем регистр алфавитного символа на строчный

mas1[am] += tolower(s[cini]);

}

cini++;

}

forn(i, am)

{

for (int j = i + 1; j <= am; j++)

{

if (real\_mas[j] == "")

break;

if (mas1[i] == mas1[j])

{

// Поиск совпадений подстроки в строке s

int index = s.find(" " + real\_mas[i]);

s.erase(s.begin() + index, s.begin() + index + real\_mas[i].size()+1);

real\_mas[i] = "";

}

}

}

cout << clear\_spaces(s) << endl;

}

**Строки**

i--;

}

}

forn(i, s.size() - 1)

{

if (s[i] == ' ' && ispunct(s[i + 1]))

s.erase(s.begin() + i);

}

return s;

}

int main()

{

string mas1[100];

string real\_mas[100];

string s;

getline(cin, s, '.');

int cini = 0;

int am = 0;

while (cini < s.size())

{

// Проверяем, является ли символ пробелом или знаком-разделителем

if (ispunct(s[cini]) || (isspace(s[cini])))

{

if (cini > 0 && !(ispunct(s[cini - 1]) || (isspace(s[cini - 1]))))

am++;

}

else

{

real\_mas[am] += s[cini];

// Меняем регистр алфавитного символа на строчный

mas1[am] += tolower(s[cini]);

}

cini++;

}

forn(i, am)

{

for (int j = i + 1; j <= am; j++)

{

if (real\_mas[j] == "")

break;

if (mas1[i] == mas1[j])

{

// Поиск совпадений подстроки в строке s

int index = s.find(" " + real\_mas[i]);

s.erase(s.begin() + index, s.begin() + index + real\_mas[i].size()+1);

real\_mas[i] = "";

}

}

}

cout << clear\_spaces(s) << endl;

}

**Строки**

real\_mas[i].size()+1);

real\_mas[i] = "";

}

}

}

cout << clear\_spaces(s) << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| Leaves from the vine  Falling so slow  Like fragile, tiny shells  Drifting in the foam  Little soldier boy  Come marching home  Brave soldier boy  Comes marching home  . | Leaves from vine  Falling so slow  Like fragile, tiny shells  Drifting in the foam  Little  Come  Brave soldier boy  Comes marching home |

**Строки**

**Упражнение IV**

Преобразование символов в числа.

**Задача 20**

Дан текст, содержащий вещественные числа. Вывести на экран все вещественные числа, содержащиеся в нем.

**Принцип решения**

Записываем строки, разделенные пробелами в массив, после чего каждую проверяем на следующие условия: в массиве должна быть одна точка и хотябы одна цифра после точки.

**Код программы**

#include <iostream>

#include <string>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

#define ini(type, n) \

type n; \

cin >> n;

int main()

{

string s[100];

int cini = 0;

while (cin)

{

cin >> s[cini++];

// Проверка следующего символа

if (cin.peek() == '\n')

break;

}

double digits[100];

int num\_digit = 0;

forn(i, cini)

{

bool onepoint = false;

bool canbe = true;

forn(j, s[i].size())

{

if (isdigit(s[i][j]) || s[i][j] == '.')

{

if (s[i][j] == '.' && (onepoint || j == s[i].size() - 1))

{

canbe = false;

break;

}

else if (s[i][j] == '.')

onepoint = true;

}

else

{

canbe = false;

break;

}

}

if (canbe)

{

digits[num\_digit] = stod(s[i]);

num\_digit++;

}

}

cout << "Doubles' found: ";

forn(i, num\_digit)

cout << digits[i] << ' ';

cout << endl;

}

**Строки**

if (s[i][j] == '.' && (onepoint || j == s[i].size() - 1))

{

canbe = false;

break;

}

else if (s[i][j] == '.')

onepoint = true;

}

else

{

canbe = false;

break;

}

}

if (canbe)

{

digits[num\_digit] = stod(s[i]);

num\_digit++;

}

}

cout << "Doubles' found: ";

forn(i, num\_digit)

cout << digits[i] << ' ';

cout << endl;

}

**Пример(ы)**

|  |  |
| --- | --- |
| Ввод | Вывод |
| 12. .34 5.6 78 9abc d | Doubles' found: 0.34 5.6 78 |

**Организация файлового ввода/вывода**

**Упражнение I**

Работа с текстовыми файлами.

**Задача 20**

Дан текстовый файл. Переписать в новый файл все его строки, удалив их них символы, стоящие на четных местах.

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

ofstream out("outfile.txt");

int main() {

string s;

while (in.peek() != EOF) {

string sNew = "";

getline(in, s);

forn(i, s.length())

if (i % 2 == 0) sNew += (char)s[i];

out << sNew << endl;

}

in.close();

out.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | outfile.txt |
| Leaves from the vine  Falling so slow  Like fragile, tiny shells  Drifting in the foam  Little soldier boy  Come marching home  Brave soldier boy  Comes marching home | Lae rmtevn  Fligs lw  Lk rgl,tn hls  Ditn ntefa  Ltl ode o  Cm acighm  Baeslirby  Cmsmrhn oe |

**Организация файлового ввода/вывода**

**Упражнение II**

Работа с текстовыми файлами.

**Задача 1**

Дан файл f, компонентами которого являются целые числа. Переписать все четные числа в файл g, нечетные – в файл h.

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("f.txt");

ofstream out("g.txt");

ofstream out2("h.txt");

int main(){

int cini;

while (in.peek() != EOF) {

in >> cini;

if (cini % 2 == 0)

out << cini << ' ';

else

out2 << cini << ' ';

}

in.close();

out.close();

out2.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| f.txt | g.txt |
| 6 4 3 -2 100  0 4 -5 -5  -34 5 2 1 1  4 2 0 5 2  0 0 3  3 6 | 6 4 -2 100 0 4 -34 2 4 2 0 2 0 0 6 |
| h.txt |
| 3 -5 -5 5 1 1 5 3 3 |

**Организация файлового ввода/вывода**

**Упражнение III**

Работа с двоичными файлами.

**Задача 5**

Создать файл из n целых чисел. Вывести на экран все четные числа данного файла.

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream intxt("infile.txt");

int main(){

int n;

intxt >> n;

int a, b;

//Создание бинарного файла

ofstream out("datfile.dat", ios::binary);

forn(i, n){

intxt >> a;

//Запись в бинарный файл

out.write((char\*)&a, sizeof(int));

}

out.close();

ifstream in("datfile.dat", ios::binary);

forn(i, n){

// Считывание с бинарного файла

in.read((char\*)&b, sizeof(int));

if (b % 2 == 0) cout << b << ' ';

}

cout << endl;

out.close();

intxt.close();

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | Вывод |
| 7  9 8 72 4 0 -2 1 | 8 72 4 0 -2 |

**Структуры**

**Упражнение I**

Решить задачу, используя структуру point для хранения координат точки:

*Замечание*. В задачах с четными номерами множество точек задано на плоскости, в задачах с нечетными номерами множество точек задано в пространстве.

**Задача 20**

Найти три различные точки из заданного множества точек, образующих треугольник наименьшего периметра.

**Принцип решения**

Создаем функцию, высчитывающую расстояние между двумя точками и структуру, содержащую три координаты. Перебираем все возможные тройки точек и высчитываем периметр треугольника, который они образуют (при этом проверяем, что они не лежат на одной прямой). Таким образом мы найдем минимальный периметр и координаты точек.

**Код программы**

#include <iostream>

#include <string>

#include <fstream>

#include <algorithm>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

ifstream in("infile.txt");

// Создание структуры

struct point {

double x;

double y;

double z;

};

double funcDistance(int i, int j, point mas1[100]){

double returned = sqrt((mas1[j].x - mas1[i].x)\*(mas1[j].x - mas1[i].x) + (mas1[j].y - mas1[i].y)\*(mas1[j].y - mas1[i].y) + mas1[j].z - mas1[i].z)\*(mas1[j].z - mas1[i].z));

return returned;

}

bool checkPoints(double x, double y, double z){

if ((x == y + z) || (y == x + z) || (z == x + y))

return true;

return false;

}

int main(){

point mas1[100];

point thisPoints[3];

double thisDistances[3];

int cini = 0;

while (in.peek() != EOF){

in >> mas1[cini].x >> mas1[cini].y >> mas1[cini].z;

cini++;

}

int P = 1;

double minDistance = DBL\_MAX;

for (int i = 0 + 0; i < cini; i++){

for (int j = i + 1; j < cini; j++){

for (int l = j + 1; l < cini; l++){

thisDistances[0] = funcDistance(i, j, mas1);

thisDistances[1] = funcDistance(j, l, mas1);

thisDistances[2] = funcDistance(l, i, mas1);

if (checkPoints(thisDistances[0], thisDistances[1], thisDistances[2])) continue;

if (minDistance >= thisDistances[0] + thisDistances[1] + thisDistances[2]){

thisPoints[0] = mas1[i];

thisPoints[1] = mas1[j];

thisPoints[2] = mas1[l];

minDistance = thisDistances[0] + thisDistances[1] + thisDistances[2];

cout << "P[" << P << "] = " << minDistance << endl;

P++;

}

}

}

}

cout << endl;

cout << "A(" << thisPoints[0].x << ", " << thisPoints[0].y << ", " << thisPoints[0].z << ")\n";

cout << "B(" << thisPoints[1].x << ", " << thisPoints[1].y << ", " << thisPoints[1].z << ")\n";

cout << "C(" << thisPoints[2].x << ", " << thisPoints[2].y << ", " << thisPoints[2].z << ")\n";

cout << "\nP[FINAL] = " << minDistance << endl;

}

**Структуры**

return false;

}

int main(){

point mas1[100];

point thisPoints[3];

double thisDistances[3];

int cini = 0;

while (in.peek() != EOF){

in >> mas1[cini].x >> mas1[cini].y >> mas1[cini].z;

cini++;

}

int P = 1;

double minDistance = DBL\_MAX;

for (int i = 0 + 0; i < cini; i++){

for (int j = i + 1; j < cini; j++){

for (int l = j + 1; l < cini; l++){

thisDistances[0] = funcDistance(i, j, mas1);

thisDistances[1] = funcDistance(j, l, mas1);

thisDistances[2] = funcDistance(l, i, mas1);

if (checkPoints(thisDistances[0], thisDistances[1], thisDistances[2])) continue;

if (minDistance >= thisDistances[0] + thisDistances[1] + thisDistances[2]){

thisPoints[0] = mas1[i];

thisPoints[1] = mas1[j];

thisPoints[2] = mas1[l];

minDistance = thisDistances[0] + thisDistances[1] + thisDistances[2];

cout << "P[" << P << "] = " << minDistance << endl;

P++;

}

}

}

}

cout << endl;

cout << "A(" << thisPoints[0].x << ", " << thisPoints[0].y << ", " << thisPoints[0].z << ")\n";

cout << "B(" << thisPoints[1].x << ", " << thisPoints[1].y << ", " << thisPoints[1].z << ")\n";

cout << "C(" << thisPoints[2].x << ", " << thisPoints[2].y << ", " << thisPoints[2].z << ")\n";

cout << "\nP[FINAL] = " << minDistance << endl;

}

**Структуры**

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | Вывод |
| 5 5 5  6 6 6  7 7 7  -20 30 10  20 14 17  13 0 -5 | P[1] = 73.0482  P[2] = 42.4645  P[3] = 29.8325  A(5, 5, 5)  B(6, 6, 6)  C(13, 0, -5)  P[FINAL] = 29.8325 |
| 5 2 9  0 -4 2  0 4 10  3 -2 4  5 3 2  4 5 1  2 0 0  9 9 9  2 5 3 | P[1] = 27.279  P[2] = 21.3194  P[3] = 21.1854  P[4] = 20.1742  P[5] = 19.5238  P[6] = 18.1229  P[7] = 13.6047  P[8] = 12.6171  P[9] = 9.01957  A(5, 3, 2)  B(4, 5, 1)  C(2, 5, 3)  P[FINAL] = 9.01957 |

**Структуры**

**Упражнение II**

Решить задачу, используя структуру, содержащую члены-данные и члены-функции.

*Замечание*. Во всех задачах данного раздела подразумевается, что исходная информация хранится в текстовом файле input.txt, каждая строка которого содержит полную информацию о некотором объекте, результирующая информация должна быть записана в файл output.txt.

**Задача 20**

На основе данных входного файла составить список студентов, включающий ФИО, курс, группу, результат забега. Вывести в новый файл список студентов, удалив из него информацию о тех студентах, которые не выполнили норматив по бегу.

**Принцип решения**

Создаем член-функцию структуры passed, в которой проверяем, является ли результат студента меньше введенного пользователем и только тогда выводим его данные. С помощью цикла считываем информацию о каждом студенте и заносим ее в эту функцию. В результате в выходной файл выводится нужная информация.

**Код программы**

#include <iostream>

#include <iomanip>

#include <string>

#include <fstream>

using namespace std;

#define forn(i,n) for(unsigned int i = 0; i < n; i++)

#define ini(type, n) \

type n; \

cin >> n;

ifstream in("infile.txt");

ofstream out("outfile.txt");

struct passed {

string name[3];

int course;

int group;

double result;

void print(double);

};

// Член-функция для вывода строки с уловием

void passed::print(double n){

if (result <= n)

out << setiosflags(ios::left) << setw(14) << name[0] + " " + name[1] + " " + name[2] << setw(7) << course << setw(9) << group << setw(9) << result << endl;

}

int main(){

string s;

passed stru;

ini(double, n);

getline(in, s, '\n');

out << s << endl;

while(in.peek() != EOF){

in >> stru.name[0] >> stru.name[1] >> stru.name[2] >> stru.course >> stru.group >> stru.result;

stru.print(n);

}

}

**Структуры**

// Член-функция для вывода строки с уловием

void passed::print(double n){

if (result <= n)

out << setiosflags(ios::left) << setw(14) << name[0] + " " + name[1] + " " + name[2] << setw(7) << course << setw(9) << group << setw(9) << result << endl;

}

int main(){

string s;

passed stru;

ini(double, n);

getline(in, s, '\n');

out << s << endl;

while(in.peek() != EOF){

in >> stru.name[0] >> stru.name[1] >> stru.name[2] >> stru.course >> stru.group >> stru.result;

stru.print(n);

}

}

**Пример(ы)**

|  |  |
| --- | --- |
| infile.txt | Ввод |
| student course group result  Aba Caba A 5 105 5.421  Huhy Jo Di 2 122 2.13  Ki Noi U 2 132 6.122  Moi Ga J 1 101 7.1  Lana Sara Mi 1 201 0.111  Kio Rio Sio 1 201 9.231  Los Re Tano 4 444 4 | 4 |
| outfile.txt |
| student course group result  Huhy Jo Di 2 122 2.13  Lana Sara Mi 1 201 0.111  Los Re Tano 4 444 4 |