Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

ПНИПУ

**Лабораторная работа**

Перегруженные функции

Выполнил:

студент группы РИС-23-1б

А. В. Головизнин

Проверила:

доцент кафедры ИТАС

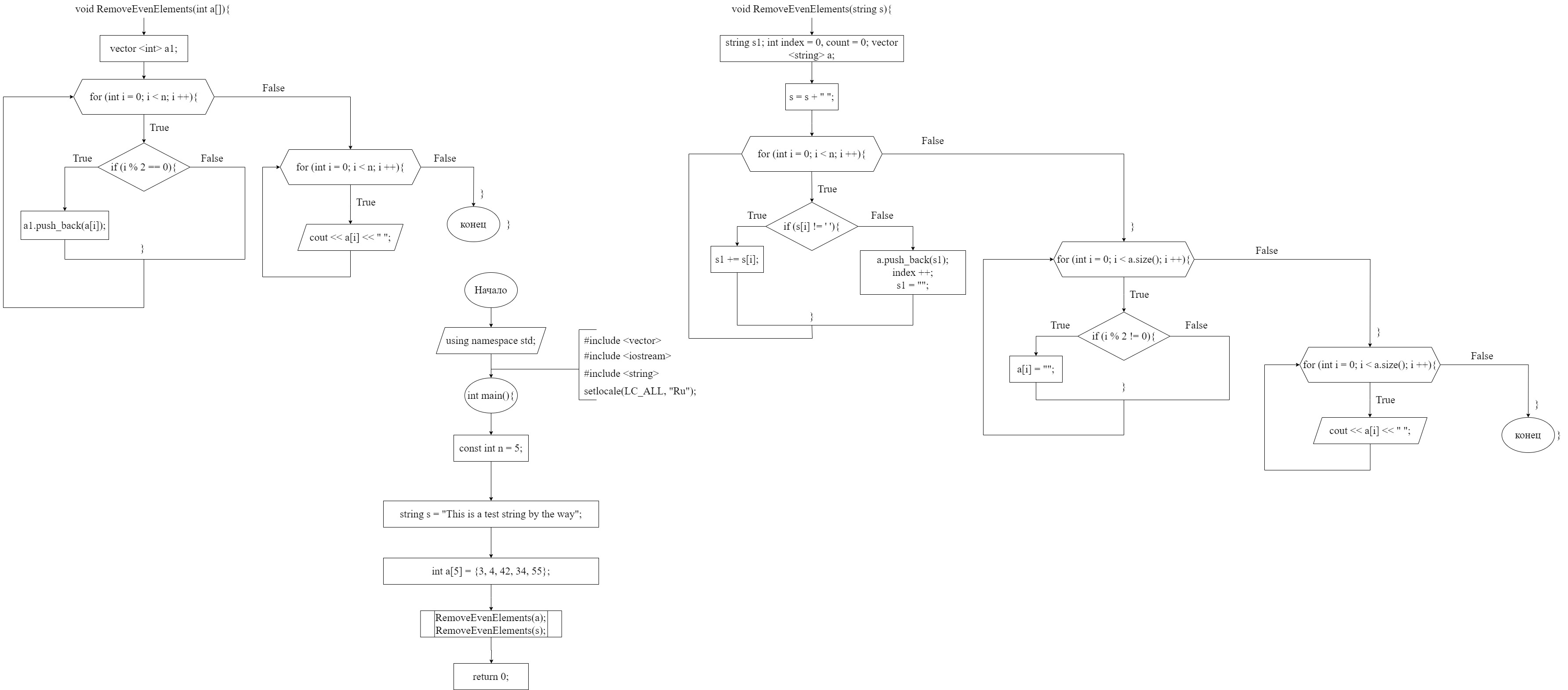
О. А. Полякова

Пермь, 2024

1) Задача: написать перегруженную функцию, которая либо удаляет чётные элементы (по нумерации) из массива, либо удаляет чётные слова из строки

2) Анализ задачи:

* Для решения задача необходимо создать перегруженную функцию **RemoveEvenElements**, если необходимо удалить элементы с чётными номерами из массива — создаём динамический массив
* Циклически проходим данный массив и, встретив элемент с чётным номером, добавляем его в динамический массив
* Если необходимо удалить чётные слова из строки — создаём динамический массив
* Проходясь по **символам** данной строки, мы добавляем каждый символ в вспомогательную строку
* Если цикл дошёл до пробела — сбрасываем вспомогательную строку до пустой, предварительно добавив эту строку в динамический массив

3) Блок — схема

4) Код на C++

#include <iostream>

#include <string>

#include <vector>

const int n = 5;

using namespace std;

void RemoveEvenElements(string s) {

string s1; int index = 0, count = 0; s = s + " "; vector <string> a;

for (int i = 0; i < s.size(); i++) {

if (s[i] != ' ') { s1 += s[i]; }

else { a.push\_back(s1); index++; s1 = ""; }

}

for (int i = 0; i < a.size(); i++) { if (i % 2 == 0) a[i] = ""; }

for (int i = 0; i < a.size(); i++) { cout << a[i] << " "; }

}

void RemoveEvenElements(int a[]) {

vector <int> a1;

for (int i = 0; i < n; i++) { if (i % 2 == 0) a1.push\_back(a[i]); }

for (int i = 0; i < a1.size(); i++) { cout << a1[i] << " "; }

}

int main() {

setlocale(LC\_ALL, "Ru");

string s = "This is a test string by the way";

int a[5] = { 3,4,42,34,55 };

RemoveEvenElements(a); cout << endl;

RemoveEvenElements(s);

return 0;

}

5) Пример работы программы:

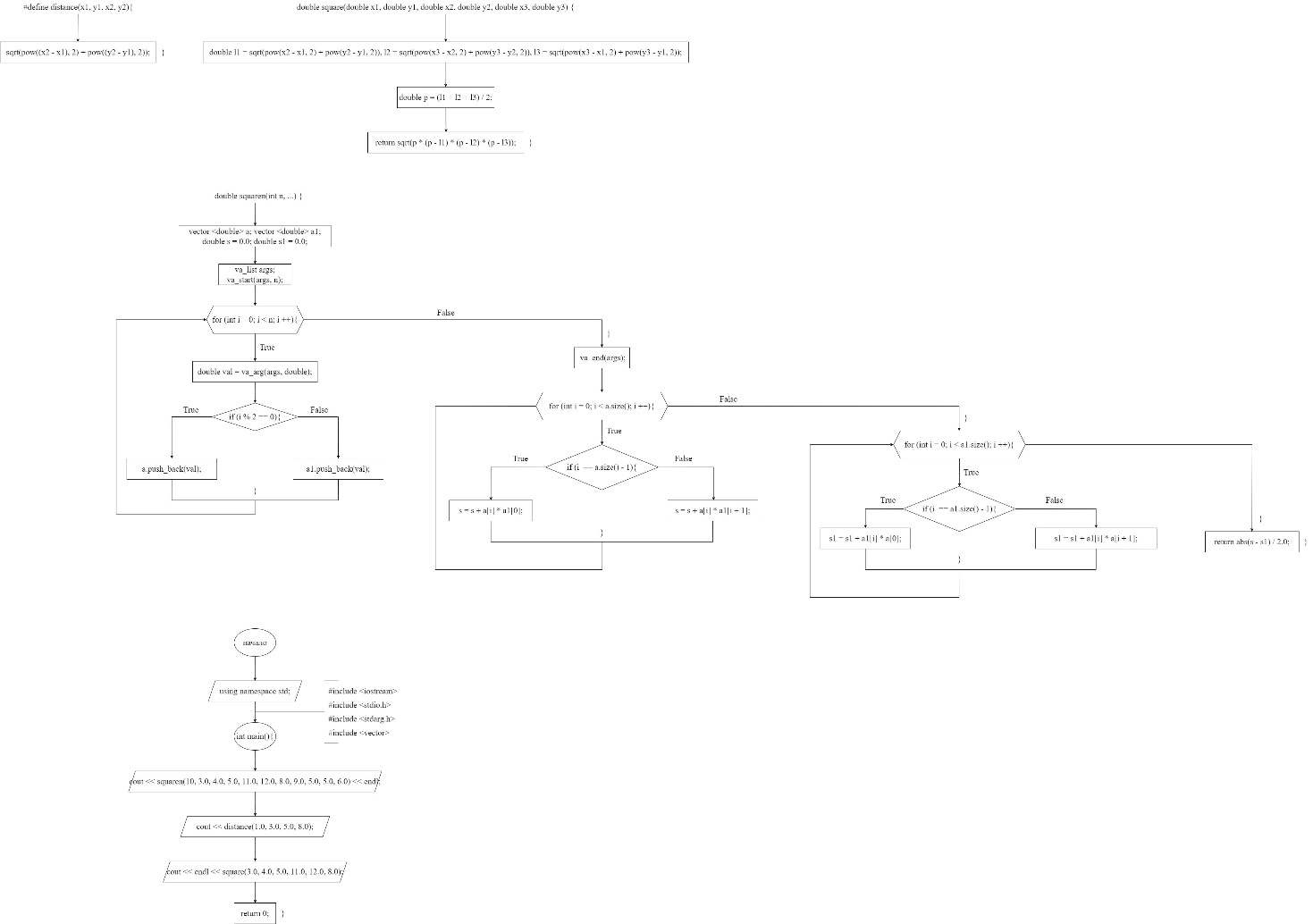


1) Задача: написать функцию (или макроопределение), которая находит длину стороны по координатам его точек. Написать функцию square, которая вычисляет площадь треугольника, заданного координатами вершин. Написать функцию squaren c переменным числом параметров, которая определяет площадь выпуклого многоугольника, заданного координатами своих вершин.

2) Анализ задачи:

* Для первой подзадачи достаточно будет макроопределения, потому что для получения ответа достаточно одного действия, использования одной формулы
* Вторая функцию реализовать также достаточно просто, для получения ответа достаточно использовать формулу Герона , где p – полупериметр треугольника, a, b, c – стороны, которые находятся по формуле из предыдущей подзадачи xi
* Третья функция – функция с переменным числом параметров, для решения распределим **x** и **y** в разные массивы, чтобы удобно применить формулу Гаусса S =

3)

****

4) Код на C ++

#include <iostream>

#include <stdio.h>

#include <stdarg.h>

#include <vector>

using namespace std;

#define distance(x1, y1, x2, y2) (sqrt(pow((x2 - x1), 2) + pow((y2 - y1), 2)));

double square(double x1, double y1, double x2, double y2, double x3, double y3) {

double l1 = sqrt(pow(x2 - x1, 2) + pow(y2 - y1, 2)), l2 = sqrt(pow(x3 - x2, 2) + pow(y3 - y2, 2)), l3 = sqrt(pow(x3 - x1, 2) + pow(y3 - y1, 2));

double p = (l1 + l2 + l3) / 2;

return sqrt(p \* (p - l1) \* (p - l2) \* (p - l3));

}

double squaren(int n, ...) {

vector <double> a; vector <double> a1; double s = 0.0; double s1 = 0.0;

va\_list args;

va\_start(args, n);

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

double val = va\_arg(args, double);

if (i % 2 == 0) a.push\_back(val);

else a1.push\_back(val);

}

va\_end(args);

for (int i = 0; i < a.size(); i++) {

if (i == a.size() - 1) s = s + a[i] \* a1[0];

else s = s + a[i] \* a1[i + 1];

}

for (int i = 0; i < a1.size(); i++) {

if (i == a1.size() - 1) s1 = s1 + a1[i] \* a[0];

else s1 = s1 + a1[i] \* a[i + 1];

}

return abs(s - s1) / 2.0;

}

int main() {

cout << squaren(10, 3.0, 4.0, 5.0, 11.0, 12.0, 8.0, 9.0, 5.0, 5.0, 6.0) << endl;

cout << distance(1.0, 3.0, 5.0, 8.0);

cout << endl << square(3.0, 4.0, 5.0, 11.0, 12.0, 8.0);

return 0;

}

5) Пример работы программы:

Изображение выглядит как Шрифт, текст, снимок экрана, белый

Автоматически созданное описание