Министерство науки и высшего образования РФ ФГБОУ ВПО

«Пермский государственный национальный исследовательский университет»

Кафедра прикладной математики и информатики

**Индивидуальное задание №1**

**«Длинный корень»**

***по дисциплине «Алгоритмизация и программирование»***

|  |  |
| --- | --- |
|  | **Выполнил:** |
|  | Студент 1-го курса механико-математического факультета  \_\_\_\_\_\_\_\_\_\_\_ Черткова С.М.,  группа ПМИ-1,2-2021 |
|  | **Проверил:** |
|  | Старший преподаватель кафедры прикладной математики и информатики ПГНИУ,  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Шеина Т. Ю. |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_ 2021 г. |

Пермь 2021

# Постановка задачи

Написать программу, которая по заданному натуральному числу А находит наибольшее число В такое, что B^2 <= A. Во входном файле INPUT.TXT записано натуральное число A (A <= 10^3000). В выходной файл OUTPUT.TXT вывести максимальное натуральное число B, квадрат которого не превосходит A. Число B следует выводить без лидирующих нулей.

# Определение идеи алгоритма,

# выбор методов решения и структур данных

Для решения задачи использовалась среда разработки Microsoft Visual Studio и язык программирования C++.

По условию задачи, во входном файле может содержаться очень длинное число, поэтому удобнее вводить число в формате строки, для операций преобразовывать строку в массив, каждый элемент которого – цифра этого числа.

После того, как входные данные прошли проверку на корректный ввод, можно приступать к выполнению поставленной задачи. Существует способ извлечения корня из целого числа, который описал А.П. Киселёв в книге «Элементы алгебры и анализа». Согласно ему, длина извлеченного корня является результатом целочисленного половинного деления увеличенной на единицу длины исходного числа. После этого выполняем следующий алгоритм:

1. Разбиваем число на пары цифр (считая с конца), после чего каждую из пар обозначаем, как элемент нового массива (размерность – длина извлеченного корня), с которым мы будем работать. Если получилось так, что первая цифра числа осталась без пары, ее обозначаем за отдельный элемент (нулевой). Т.к. извлекать корень удобнее сначала, то число мы не переворачиваем.
2. Чтобы найти первую цифру результата, извлекаем корень из первой пары (нулевой элемент массива) методом перебора цифр от 0 до 9. В том случае, когда квадрат текущей цифры будет превышать значение элемента (первой пары), мы выйдем из цикла и в результирующую строку запишем цифру, полученную на прошлом шаге, либо 9, если дошли до конца.
3. Находим разность значения первой пары и квадрата цифры, записанной в результирующей строке, после чего приписываем полученный остаток ко 2 паре, если такова имеется.
4. Далее мы проходимся по циклу, который будет находить оставшиеся цифры:
   1. Удваиваем значение результирующей строки, считаем его за количество десятков промежуточного числа. В качестве единиц данного числа мы подбираем цифру от 0 до 9 с помощью бинарного поиска, которая будет удовлетворять требованию: произведение промежуточного числа и подобранной цифры меньше или равно значению элемента, состоящего из пары и остатка, полученного на прошлом шаге.
   2. Когда подобрана очередная цифра, к следующей паре приписываем разность значения текущей пары и промежуточного числа, умноженного на подобранную цифру. Если разность или значение следующей пары равны нулю, то могут возникнуть ошибки, поэтому следует дописать упущенные на этапе формирования пар нули или удалить незначащие, если это необходимо.
   3. К результирующей строке приписываем подобранную цифру.
   4. Работа цикла остановится после того, как мы обработаем последнюю пару.

Замечание: числа, над которыми выполняются операции умножения, сравнения, вычитания, могут увеличиваться в длине при каждом последующем вхождении в цикл, поэтому эти операции следует рассмотреть в виде отдельных функций над длинными числами, которые представлены в виде строк:

1. Сравнение проходит на основе сравнения соответствующих элементов (цифр числа) с одинаковыми индексами, если длины чисел равны. Иначе результат сравнения будет найдет, исходя из сравнения длин заданных чисел. Так как в данной задаче нужно лишь убедиться в том, что первое число меньше или равно второму, то в качестве результата функция будет возвращать переменную логического типа.
2. С вычитанием длинных чисел мы познакомились на дисциплине «АиП», а так как умножение в имеющейся программе выполняется только на однозначное число, то эта операция похожа на операцию сложения, с которой мы также познакомились в ходе изучения данной дисциплины. В качестве описания этих операций можно рассмотреть комментарии, указанные к функциям.

# Тестирование программы

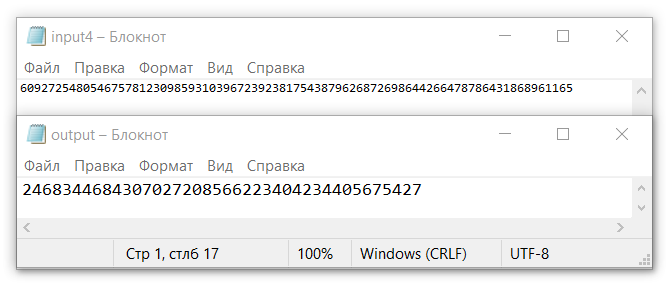
Программа была протестирована для 6-ти видов входных данных, для каждого из них ниже дано описание, представлен тест и результат прохождения данного теста программой. Все ожидаемые результаты были получены при помощи сайта <https://web2.0calc.com>.

1. Исходные данные не имеют ошибок; на входе – число с большим количеством символов.

*ВХОДНОЙ ФАЙЛ:* 6092725480546757812309859310396723923817543879626872698644266478786431868961165

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* 2468344684307027208566223404234405675427

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

**

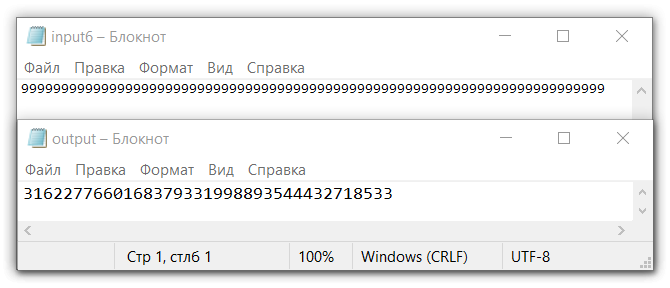
**

1. Исходные данные не имеют ошибок; на входе – другое число с большим количеством символов.

*ВХОДНОЙ ФАЙЛ:* 9999999999999999999999999999999999999999999999999999999999999999999999999

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* 3162277660168379331998893544432718533

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

**

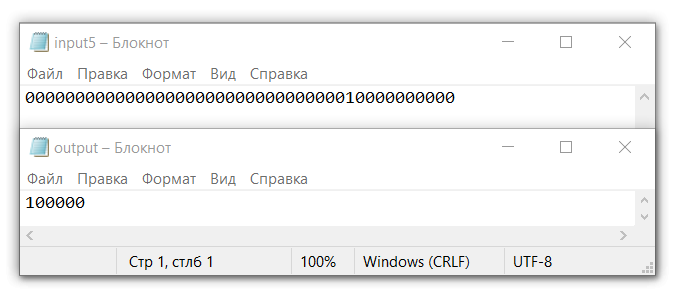
**

1. Исходные данные не имеют ошибок; на входе – число с незначащими нулями.

*ВХОДНОЙ ФАЙЛ:* 0000000000000000000000000000000010000000000

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* 100000

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

**

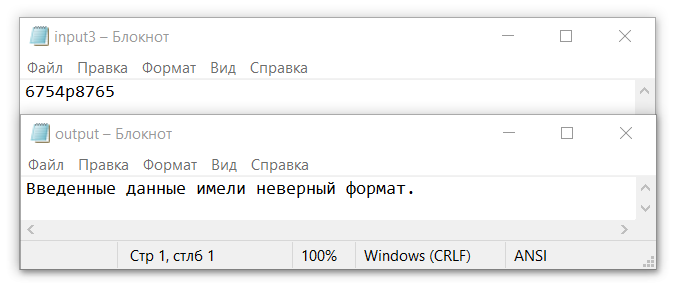
**

1. Исходные данные содержат ошибку: строка содержит символ, не являющийся цифрой.

*ВХОДНОЙ ФАЙЛ:* 6754p8765

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* Введённые данные имели неверный формат.

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

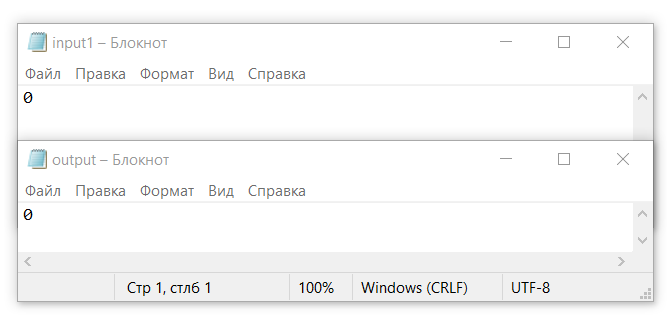
**

1. Исходные данные не имеют ошибок; на входе – нулевое значение.

*ВХОДНОЙ ФАЙЛ:* 0

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* 0

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

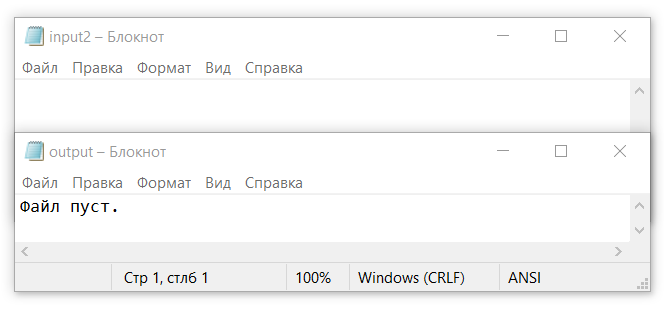
**

1. Исходные данные содержат ошибку: на входе – пустая строка.

*ВХОДНОЙ ФАЙЛ ПУСТ*

*ОЖИДАЕМЫЙ РЕЗУЛЬТАТ -* Фаил пуст.

*РЕЗУЛЬТАТ, ПОЛУЧЕННЫЙ ПРИ ТЕСТИРОВАНИИ ПРОГРАММЫ:*

**

# Текст программы

Исходный файл программы *ind\_Chertkova\_Sofya.cpp* прилагается к работе в электронном виде, ниже приводим его полностью.

#include <iostream>

#include <string>

#include <windows.h>

#include <fstream>

using namespace std;

string razn(string sa, string sb); // вычитание длинных чисел

bool comparison(string x, string y); //сравнение длинных чисел "<="

string mult(string sx, int m); //умножение длинного числа на цифру

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int size\_a, size\_b, i = 0,j,num; //size\_a - длина заданного числа, size\_b - длинна результата, i,j - счетчики, num - подобранная цифра

string a,c,x,t,k,d;//a - заданное число, c - результативное число, x,t,k,d - промежуточные переменные

bool check, check\_empty;

ifstream f1("input1.txt");

getline(f1, a); // чтение строки из файла

ofstream f2("output.txt", ios::out);

size\_a = a.size();

check = check\_empty = false;

if (size\_a > 0 and !(a == "0")) //удаление незначащих нулей

{

while (a[i] == '0')

{

a.erase(i, 1);

size\_a--;

}

}

if (size\_a == 0) //проверка на "пустой фаил"

{

check = true;

check\_empty = true;

}

for (i = 0; i < size\_a; i++) //проверка на правильность введённых данных

{

if (a[i] < '0' or a[i] > '9')

{

check = true;

}

}

if (!(check))

{

size\_b = (size\_a + 1) / 2; //длина результата после удаления незначащих (при их существовании)

int\* b = new int[size\_b];

for (i = 0; i < size\_b; i++) //заполняем нулями массив, где будет храниться ответ

{

b[i] = 0;

}

if (size\_a % 2 != 0) //заполняем массив парами цифр числа, если 1-я цифра без пары

{

b[0] = a[0] - '0';

j = 1;

for (i = 1; i < size\_b; i++)

{

b[i] = (a[j]-'0') \* 10 + (a[j + 1]-'0');

j += 2;

}

}

else //заполняем массив парами цифр числа, если количество цифр четное

{

j=0;

for (i = 0; i < size\_b; i++)

{

b[i] = (a[j]-'0') \* 10 + (a[j + 1]-'0');

j += 2;

}

}

for (i = 0; i <= 9; i++) // находим из 1-ой пары первую цифру результата

{

if (i \* i <= b[0])

{

num = i;

}

else

{

break;

}

}

if (size\_b > 1) b[1] = (b[0] - num \* num) \* 100 + b[1]; //остаток переносим к 2-ой паре

b[0] = num; // записываем 1-ую цифру результата

c = ""; k = ""; t = "";

c = to\_string(b[0]); //c - 1-я цифра

if (size\_b > 1) k = to\_string(b[1]); //k - 2-я пара

for (i = 1; i < size\_b; i++)

{

x = mult(c, 2); // удваиваем результат, полученный на прошлом шаге, x - количество десятков

int l = 0, r = 9,m;

while (l <= r) // с помощью бин.поиска подбираем нужную цифру

{

m = (l + r) / 2; // подбираемая цифра

t = x + to\_string(m); // составляем число t, где x - десятки, m - единицы

if (comparison(mult(t,m),k) == 1) // подбираемся цифра пойдет в результат, если t\*m <= k, проверяем это условие

{

num = m;

l = m + 1;

}

else

{

r = m - 1;

}

}

if (i != size\_b - 1) // если не конец

{

d = "";

if (b[i + 1] < 10) //когда пары формировались, пары вида "00" -> "0", "01" -> "1", поэтому к d добавляем "0"

{

d = razn(k, mult(x + to\_string(num), num)) + "0" + to\_string(b[i + 1]); //остаток переносим в следующую пару

for (j = 0; j < d.size() - 1; j++) //удаляем незначащие нули, чтобы не возникло проблем с произведением

{

if (d[j] == '0')

{

d.erase(j, 1);

}

else

{

break;

}

}

k = "";

k = d; //в следующую пару записываем остаток от разности + пара

}

else

{

d = razn(k, mult(x + to\_string(num), num)) + to\_string(b[i + 1]); // остаток переносим в следующую пару

for (j = 0; j < d.size() - 1; j++) //удаляем незначащие нули, чтобы не возникло проблем с произведением

{

if (d[j] == '0')

{

d.erase(j, 1);

}

else

{

break;

}

}

k = "";

k = d; // в следующую пару записываем остаток от разности + пара

}

}

b[i] = num; //записываем уже подобранную цифру

c += to\_string(b[i]); //в результирующую строку записываем подобранную цифру

}

f2 << c;//записываем результат в файл

f1.close();

f2.close();

delete[]b;

}

else

{

if (check\_empty)

f2 << "Файл пуст."; //если файл пуст, то выводится сообщение

else

f2 << "Введенные данные имели неверный формат."; //если введённые данные неверны, то выводится сообщение

}

return 0;

}

string mult(string sx, int m) //умножение длинного числа на цифру

{

int length = sx.size() + 1,z=0; //length - длина результата с местом для переноса, z - разряд для переноса

int\* x = new int[length];

for (int i = 0; i < length; i++) //заполняем нулями массив, где будет храниться ответ

{

x[i] = 0;

}

for (int i = sx.size() - 1; i >= 0; i--) //заполняем массив цифрами числа с конца

{

x[sx.size() - 1 - i] = sx[i] - '0';

}

for (int i = 0; i < length; i++) //вычисления

{

if (i < sx.size())

{

x[i] = x[i] \* m + z; //умножается последний разряд числа на цифру и прибавляется разряд для переноса с прошлого шага

z = x[i] / 10; //записывается разряд для переноса

x[i] %= 10; //если есть разряд для переноса, он отсекается

}

}

x[length - 1] = z; //на последнее место записывается разряд для переноса

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

{

if (x[length - 1] == 0) //удаляются незначащие нули

{

length--;

}

else

{

break;

}

}

sx = ""; //результирующая строка

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

{

sx += to\_string(x[i]); //добавляем по 1 цифре к результату с конца

}

return sx;

}

bool comparison(string x, string y) //сравнение длинных чисел "<="

{

bool t = true; //true - если 1-е <= 2-ого, false - если 1-е > 2-го

if (x.size() < y.size()) //если длина 1-го меньше длины 2-го, то 1-е число меньше

{

t = true;

}

if (x.size() > y.size()) //если длина 1-го больше длины 2-го, то 1-е число больше

{

t = false;

}

if (x.size() == y.size()) //если числа одинаковой длины, то сравниваем по разрядам, начиная с 1

{

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

{

if (x[i] == y[i]) //пока разряды равны, пропускаем

{

continue;

}

else

{

if (x[i] > y[i]) //если разряд 1-го числа > разряда 2-го, то 1-е числа > 2-го числа

{

t = false;

break;

}

else //если разряд 1-го числа < разряда 2-го, то 1-е числа < 2-го числа

{

t = true;

break;

}

}

}

}

return t;

}

string razn(string sa,string sb) // вычитание длинных чисел

{

int size\_a, size\_b, length;

size\_a = sa.length(); // длина 1-ого числа

size\_b = sb.length(); // длина 2-ого числа

string d = "0";

int k = 2;

length = size\_a; //1-е число >= 2-е числ0, это проверено на предыдущем шаге

if (size\_a > size\_b) //k=1, если 1-е числа > 2-го числа

{

k = 1;

}

else

{

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

{

if (sa[i] > sb[i])

{

k = 1;

break;

}

}

}

if (k == 2) return d; //k=2, если числа равны, то их разность равна 0

else

{

int\* a = new int[size\_a];

int\* b = new int[size\_b];

int\* c = new int[length];

for (int i = 0; i < length; i++) //заполняем нулями массив, где будет храниться ответ

{

c[i] = 0;

}

for (int i = size\_a - 1; i >= 0; i--) //заполняем массив цифрами 1-го числа с конца

{

a[size\_a - 1 - i] = sa[i] - '0';

}

for (int i = size\_b - 1; i >= 0; i--) //заполняем массив цифрами 2-го числа с конца

{

b[size\_b - 1 - i] = sb[i] - '0';

}

for (int i = 0; i < length; i++) //вычитание

{

if (i < size\_b)

{

c[i] = a[i] - b[i];

if (c[i] < 0)

{

c[i] += 10;

a[i + 1]--;

}

}

else

c[i] = a[i];

}

sb = "";

for (int i = 0; length > 0; i--) //удаляем незначащие нули

{

if (c[length - 1] == 0)

{

length--;

}

else

{

break;

}

}

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

{

sb += to\_string(c[i]); //добавляем по 1 цифре к результату с конца

}

return sb;

}

}