МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

Отчет по практике

Реализовать программу для работы с длинными числами (теми, которые не помещаются в тип long).

2 курс, группа 2УТС

Выполнил:
_____ А. Е. Тарасьян
«___» _____ 2024 г.

Руководитель:
_____ С. В. Теплоухов
«___» _____ 2024 г.

Майкоп, 2024 г.

1. Введение

1) Задание

2) Код прилагающий к заданию

3) Скриншот программы

Содержание

2. Теория

2.1. Техническое задание

Задание: Реализовать программу для работы с длинными числами (теми, которые не помещаются в тип long).

2.2. Теоретическая часть

-

Для описания алгоритмов обычно используется следующее представление целых чисел. Выбирается основание $\beta > 1$. Тогда целое число длины n можно представить в виде:

$$A = a_{n-1} \cdot \beta^{n-1} + \ldots + a_1 \cdot \beta + a_0,$$

где $0 \le a_i \le \beta - 1$.

Умножение Карацубы

Метод умножения Карацубы относится к парадигме «разделяй и властвуй». Вычислительная сложность алгоритма:

$M(n) = O(n^{\log_2 3}), \quad \log_2 3 = 1{,}5849\ldots.$

Данный алгоритм представляет собой простую реализацию идеи разделения входных данных, которая стала базисной для нижеперечисленных алгоритмов. Идея заключается в разделении одной операции умножения над n-значными числами на три операции умножения над числами длины n/2 плюс O(n).

Для перемножения двух чисел, превышающих длину машинного слова, алгоритм Карацубы вызывается рекурсивно до тех пор, пока эти числа не станут достаточно маленькими, чтобы их можно было перемножить непосредственно. Пример такого алгоритма:

Рис. 1.

3. Ход работы

3.1. Код прилагающий к заданию

```
#Программа, реализующая умножение Карацубы для работы с длинными числами
   #include <iostream>
#include <string>
#include <algorithm>

std::string add(const std::string& a, const std::string& b) {
    std::string result;
    int carry = 0;
    int length = std::max(a.length(), b.length());
    for (int i = 0; i < length || carry; ++i) {
        int sum = carry;
        if (i < a.length()) {
            sum += a[a.length() - 1 - i] - '0';
        }
        if (i < b.length()) {
            sum += b[b.length() - 1 - i] - '0';
        }
        carry = sum / 10;
        result += (sum % 10) + '0';
    }
    std::reverse(result.begin(), result.end());
    return result;
}

std::string subtract(const std::string& a, const std::string& b) {
    std::string result;
    int length = std::max(a.length(), b.length());
    int borrow = 0;
    for (int i = 0; i < length; ++i) {
        int x = (i < a.length()) ? (a[a.length() - 1 - i] - '0') : 0;
        int y = (i < b.length()) ? (b[b.length() - 1 - i] - '0') : 0;
        int diff = x - y - borrow;
```

```
        if (diff < 0) {
            diff += 10;
            borrow = 1;
        }
        else {
            borrow = 0;
        }
        result += diff + '0';
    }
    while (result.length() > 1 && result.back() == '0') {
        result.pop_back();
    }
    std::reverse(result.begin(), result.end());
    return result;
}

std::string karatsuba(const std::string& x, const std::string& y) {
    if (x.length() < 2 || y.length() < 2) {
        return std::to_string(std::stol(x) * std::stol(y));
    }

    int n = std::max(x.length(), y.length());
    int m = n / 2;

    std::string a = x.substr(0, x.length() - m);
    std::string b = x.substr(x.length() - m);
    std::string c = y.substr(0, y.length() - m);
    std::string d = y.substr(y.length() - m);

    std::string ac = karatsuba(a, c);
    std::string bd = karatsuba(b, d);
    std::string abcd = karatsuba(add(a, b), add(c, d));
    std::string adbc = subtract(subtract(abcd, ac), bd);

    for (int i = 0; i < 2 * m; ++i) {
```

```
        ac += "0";
    }
    for (int i = 0; i < m; ++i) {
        adbc += "0";
    }

    return add(add(ac, adbc), bd);
}

int main() {
    std::string num1 = "3141592653589793238462643383279502884197169399375
    std::string num2 = "2718281828459045235360287471352662497757247093699

    std::cout << "Result: " << karatsuba(num1, num2) << std::endl;

    return 0;
}
```

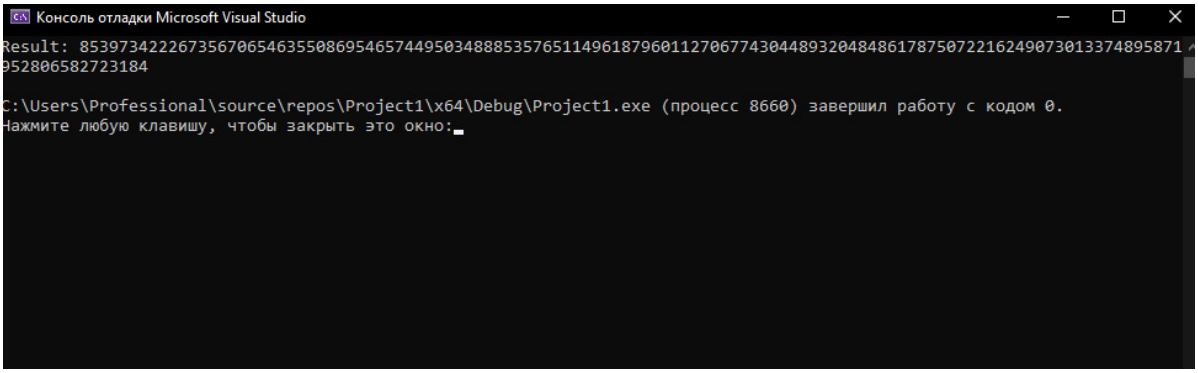

Рис. 2. Enter Caption