

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ КОСМИЧЕСКИХ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

ОТЧЕТ
по дисциплине Алгоритмы и структуры данных
Практическая работа №1 — Оценка сложности алгоритма

Преподаватель

подпись, дата

Матковский И. В.

инициалы, фамилия

Студент

КИ19-07Б, 031941597

номер группы, зачётной книжки

подпись, дата

Горбацевич А. А.

инициалы, фамилия

Красноярск 2020

Содержание

1. Задание на работу.....	3
1.1 Разработать для решения поставленной задачи два алгоритма с разными уровнями сложности.....	3
2. Задание на вариант.....	4
2.1 Вычислить суммы факториалов К первых целых чисел для K=1..N.....	4
3. Исходный код программы.....	5
4. Теоретические оценки временной и пространственной сложности алгоритмов.....	7
4.1 Неэффективный алгоритм.....	7
4.2 Эффективный алгоритма.....	7
5. Экспериментальные оценки временной и пространственной сложности алгоритмов.....	8
Приложение А Результаты работы программы.....	9

1. Задание на работу

1.1 Разработать для решения поставленной задачи два алгоритма с разными уровнями сложности.

2. Задание на вариант

2.1 Вычислить суммы факториалов К первых целых чисел для K=1..N.

3. Исходный код программы

```
// dsaa_01.cpp
// Горбацевич Андрей
#include <iostream>
#include <cmath>
#include <chrono>

typedef double long_type;

void time_passed(std::chrono::system_clock::time_point start, long_type& holder);

void test(int N);
void algo_01(int N, long_type& out_val); // Naive way
void algo_02(int N, long_type& out_val); // Somewhat efficient way

int main() {
    int i;
    std::cout << "N=";
    std::cin >> i;

    test(i);

    return 0;
}

inline void time_passed(std::chrono::system_clock::time_point start, long_type& holder) {
    auto stop = std::chrono::high_resolution_clock::now();
    auto duration = std::chrono::duration_cast<std::chrono::microseconds>(stop - start);
    holder = duration.count();
}

void test(int N) {
    long_type fft, sft = 0;
    long_type ffv, sfv = 0;

    {
        auto start = std::chrono::high_resolution_clock::now();
        algo_01(N, ffv);
        time_passed(start, fft);
    }
    {
        auto start = std::chrono::high_resolution_clock::now();
        algo_02(N, sfv);
        time_passed(start, sft);
    }

    printf(
        "algo_01 is %s than algo_02 by %.0f microseconds (%.0f against %.0f; N=%d; %.0f; %.0f)\n"
        "Sum of i! from 1 to %d = %.0f",
        (fft > sft? "slower" : "faster"), fabs(fft - sft), fft, sft, N, ffv, sfv, N, ffv
    );
}
```

```

void algo_01(int N, long_type& out_val) { // O(N^2)
    long_type sum = 0;

    for (int k = 1; k <= N; k++) { // O(N)
        long_type acc = 1;
        for (int i = 1; i <= k; i++) { // O(N)
            acc *= i;
        }
        sum += acc;
    }

    out_val = sum;
}

void algo_02(int N, long_type& out_val) { // O(N)
    long_type sum = 0;
    long_type acc = 1;

    for (int k = 1; k <= N; k++) { // O(N)
        sum += (acc *= k);
    }

    out_val = sum;
}
// Вычислить суммы факториалов K первых целых чисел для K=1..N

```

4. Теоретические оценки временной и пространственной сложности алгоритмов

4.1 Неэффективный алгоритм

Данный алгоритм описан в функции algo_01. Временная сложность данного алгоритма: $O(n^2)$, за счёт использования вложенных циклов, просчитывающих промежуточное значение для каждого К. Пространственная сложность: $O(1)$, за счёт использования статических типов данных.

4.2 Эффективный алгоритма

Данный алгоритм описан в функции algo_02. Временная сложность данного алгоритма: $O(n)$, за счёт повторного использования предыдущих значений К. Пространственная сложность: $O(1)$.

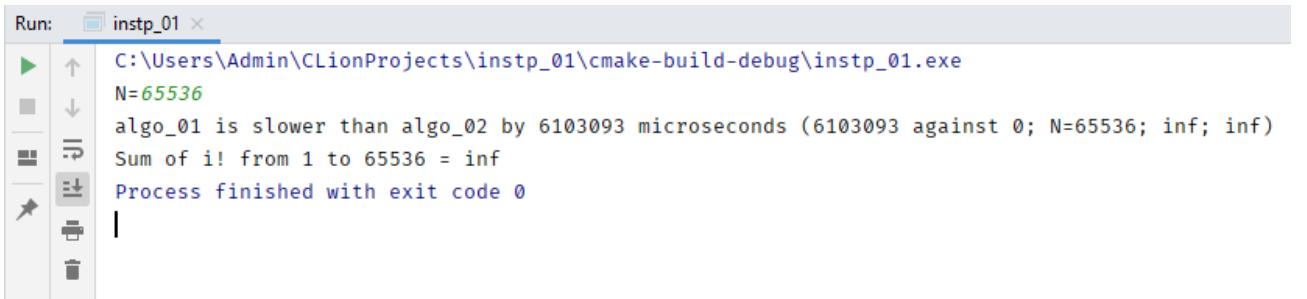
5. Экспериментальные оценки временной и пространственной сложности алгоритмов

K	Время, algo_01, микросекунды	Занимаемое пространство, algo_01, байты	Время, algo_02, микросекунды	Занимаемое пространство, algo_02, байты
0	0	8	0	8
132	0	8	0	8
264	0	8	0	8
396	995	8	0	8
528	997	8	0	8
660	996	8	0	8
792	996	8	0	8
924	1994	8	0	8
256	0	8	0	8
4096	25922	8	0	8
65536	6153707	8	997	8

Таблица 1: данные экспериментов

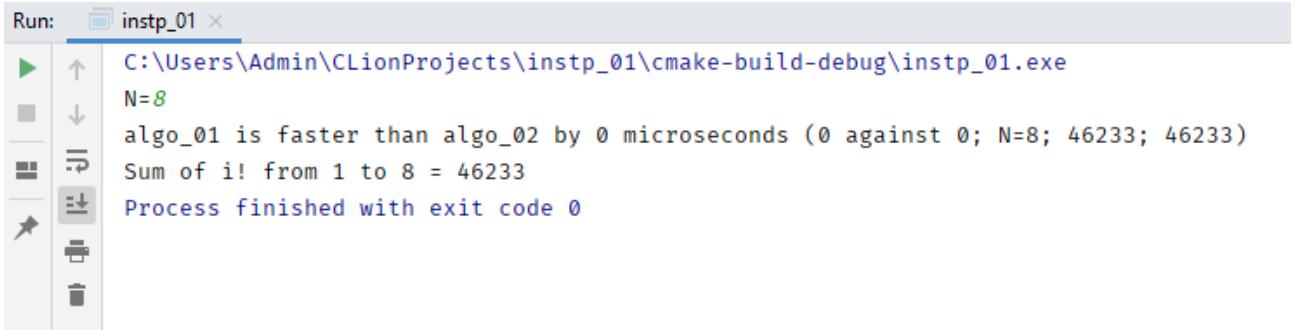
Приложение А

Результаты работы программы



```
Run: instp_01 ×
C:\Users\Admin\CLionProjects\instp_01\cmake-build-debug\instp_01.exe
N=65536
algo_01 is slower than algo_02 by 6103093 microseconds (6103093 against 0; N=65536; inf; inf)
Sum of i! from 1 to 65536 = inf
Process finished with exit code 0
```

Рисунок 1: результат работы программы на больших K



```
Run: instp_01 ×
C:\Users\Admin\CLionProjects\instp_01\cmake-build-debug\instp_01.exe
N=8
algo_01 is faster than algo_02 by 0 microseconds (0 against 0; N=8; 46233; 46233)
Sum of i! from 1 to 8 = 46233
Process finished with exit code 0
```

Рисунок 2: результат работы программы на малых K