|  |  |  |
| --- | --- | --- |
| Министерство науки и высшего образования  Российской Федерации | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования | | |
| «Новосибирский государственный технический университет» | | |
|  | | |
|  | | |
|  | | |
| Практическое задание №8 | | |
| по дисциплине «Методы построения и анализа алгоритмов» | | |
|  | | |
| **кафедра теоретической и прикладной информатики** | | |
|  | | |
|  | Факультет: | ПМИ |
| Группа: | ПМИ-03 |
| Бригада: | Место для ввода текста. |
| Студенты: | Сидоров Даниил, |
|  | Малыгин Сергей |
|  |  |
| Преподаватель: | Щукин Георгий Анатольевич |
|  |  |
|  | | |
| Новосибирск | | |
| 2021 | | |

**1.Результаты замеров:**

|  |  |  |  |
| --- | --- | --- | --- |
| First | Second | Динамическое программирование | Наивная реализация |
| 10 | 10 | 8.3e-06 | 0.0060218 |
| 15 | 15 | 6.9e-06 | 4.12521 |
| 16 | 16 | 5.2e-06 | 13.9189 |
| 100 | 100 | 0.0001338 | - |
| 1 000 | 1 000 | 0.0069475 | - |
| 10 000 | 10 000 | 0.639161 | - |
|  |  |  |  |
| 30 | 10 | 2.72e-05 | 7.05525 |
| 10 | 30 | 1.29e-05 | 16.6119 |
| 5 | 50 | 8e-06 | 0.0362443 |
| 50 | 5 | 2.35e-05 | 0.0709177 |
| 7 | 50 | 1.01e-05 | 3.28207 |
| 50 | 7 | 2.17e-05 | 5.38458 |

**2.Программа:**

**lcs.cpp:**

#include "lcs.h"

#include "vector"

#include <cstdlib>

#include <algorithm>

#include <iterator>

using namespace std;

string random\_string(int size) {

static const std::string alphabet = "0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";

return random\_string(size, alphabet);

}

string random\_string(int size, const std::string &alphabet) {

std::string result;

auto random\_char = [&]() { return alphabet[rand() % alphabet.size()]; };

std::generate\_n(std::back\_inserter(result), size, random\_char);

return result;

}

string lcs(const string &first, const string &second)

{

vector<vector<int>> mx(first.size() + 1, vector<int>(second.size() + 1, 0));

int i,j;

for (i = 1; i < first.size() + 1; ++i)

{

for (j = 1; j < second.size() + 1; ++j)

{

if (first[i - 1] == second[j - 1])

{

mx[i][j] = mx[i - 1][j - 1] + 1;

}

else {

mx[i][j] = max(mx[i - 1][j], mx[i][j - 1]);

}

}

}

string result;

i = first.size();

j = second.size();

while (i > 0 && j > 0)

{

if (first[i - 1] == second[j - 1])

{

result.insert(result.begin(),first[i - 1]);

i -= 1;

j -= 1;

}

else if (mx[i - 1][j] == mx[i][j])

{

i -= 1;

}

else {

j -= 1;

}

}

return result;

}

string naive\_solution(const string &first, const string &second)

{

if (first.empty() || second.empty()) return "";

string sub1 = first.substr(1);

string sub2 = second.substr(1);

if (first[0] == second[0])

{

return first[0] + naive\_solution(sub1, sub2);

}

else

{

string a = naive\_solution(sub1, second);

string b = naive\_solution(first, sub2);

if (a.length() > b.length()) return a; else return b;

}

}

**Main.cpp**

#include <chrono>

#include <iostream>

#include <string>

#include "lcs.h"

#define CATCH\_CONFIG\_RUNNER

#include "catch.hpp"

int main(int argc, char\* argv[]) {

int result = Catch::Session().run(argc, argv);

std::string str1, str2,rand1,rand2;

for (int i = 10; i <= 10000; i = i \* 10)

{

std::cout << std::endl << "i: " << i << std::endl;

rand1 = random\_string(i);

rand2 = random\_string(i);

auto t1 = std::chrono::high\_resolution\_clock::now();

str1 = lcs(rand1, rand2);

auto t2 = std::chrono::high\_resolution\_clock::now();

auto seconds = std::chrono::duration<double>(t2 - t1).count();

std::cout << std::endl << "Dynamic programming: " << seconds << std::endl;

t1 = std::chrono::high\_resolution\_clock::now();

str1 = naive\_solution(rand1, rand2);

t2 = std::chrono::high\_resolution\_clock::now();

seconds = std::chrono::duration<double>(t2 - t1).count();

std::cout << "Naive solution: " << seconds << std::endl;

}

return result;

}