

Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего
профессионального образования

**Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики**

Кафедра Компьютерной фотоники и видеоинформатики

Отчет по практике

Выполнил: Кириллов О. В.

Группа: V3316

Преподаватель: Кудрявцев А. С.

Санкт-Петербург, 2018

ОГЛАВЛЕНИЕ

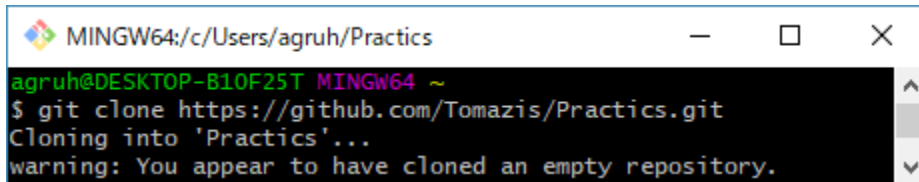
Цель работы.....	3
Лабораторная работа №1. Знакомство с системой контроля версий Git.....	4
Лабораторная работа № 2 «Форматирование и стиль».....	6
Лабораторная работа № 3 «TDD: Разработка через тестирование».....	10
Задание. Подсчет количества и характеристик зерен по фотографии.....	12
Приложение А. Слайды презентации с семинара по C++11 на тему: «Регулярные выражения».....	14

Цель проведения практики

Освоение навыков использования C++ и изучение приемов разработки программного обеспечения. Практика проходит в компьютерном классе и состоит из лекционных занятий и практических заданий.

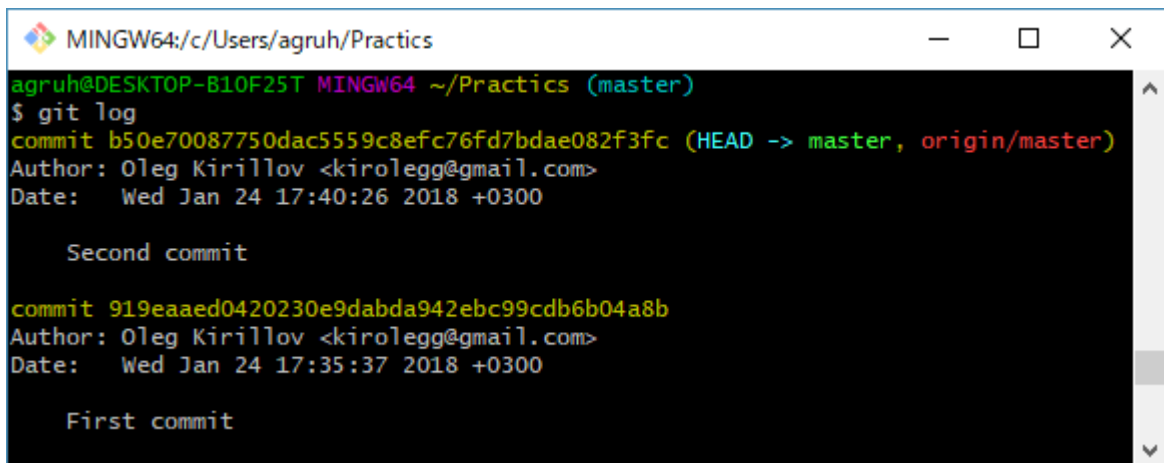
Лабораторная работа №1. Знакомство с системой контроля версий Git

1. Завел аккаунт. <https://github.com/Tomazis>
2. Создал репозиторий для лабораторных работ. <https://github.com/Tomazis/Practics>
3. Склонировал репозиторий.



```
MINGW64:/c/Users/agruh/Practics
agruh@DESKTOP-B10F25T MINGW64 ~
$ git clone https://github.com/Tomazis/Practics.git
Cloning into 'Practics'...
warning: You appear to have cloned an empty repository.
```

4. Создал папку lab1.
5. Написал программу, вычисляющую функцию факториала.
6. Сделал 2 коммита. Можно увидеть через команду git log.



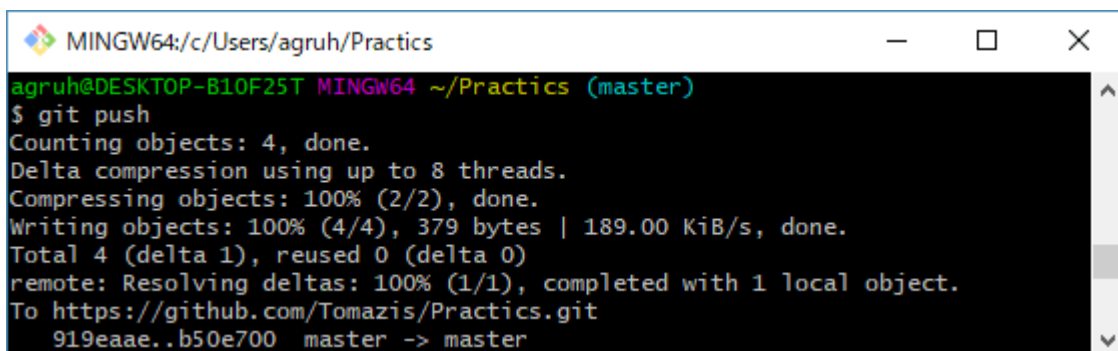
```
MINGW64:/c/Users/agruh/Practics
agruh@DESKTOP-B10F25T MINGW64 ~/Practics (master)
$ git log
commit b50e70087750dac5559c8efc76fd7bdae082f3fc (HEAD -> master, origin/master)
Author: Oleg Kirillov <kirolegg@gmail.com>
Date:   Wed Jan 24 17:40:26 2018 +0300

    Second commit

commit 919eaaed0420230e9dabda942ebc99cdb6b04a8b
Author: Oleg Kirillov <kirolegg@gmail.com>
Date:   Wed Jan 24 17:35:37 2018 +0300

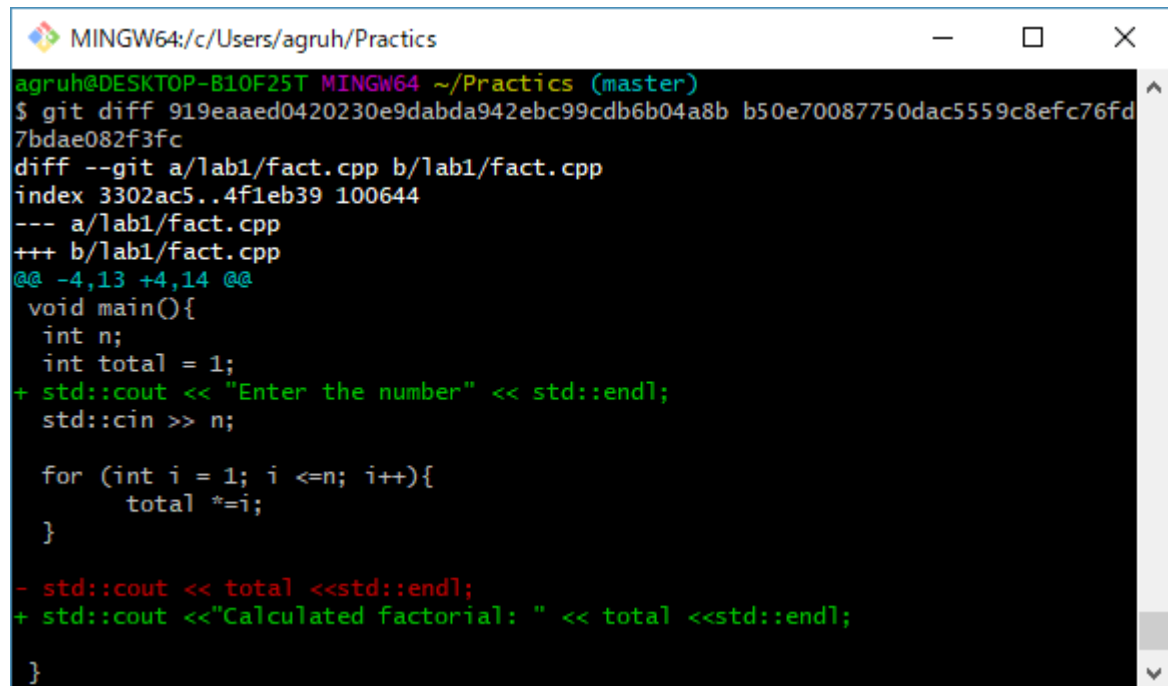
    First commit
```

7. Отправил на github через git push.



```
MINGW64:/c/Users/agruh/Practics
agruh@DESKTOP-B10F25T MINGW64 ~/Practics (master)
$ git push
Counting objects: 4, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 379 bytes | 189.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Tomazis/Practics.git
    919eaae..b50e700  master -> master
```

8. Различия между двумя коммитами через git diff.



```
MINGW64/c/Users/agruh/Practics
agruh@DESKTOP-B10F25T MINGW64 ~/Practics (master)
$ git diff 919eaaed0420230e9dabda942ebc99cdb6b04a8b b50e70087750dac5559c8efc76fd
7bdae082f3fc
diff --git a/lab1/fact.cpp b/lab1/fact.cpp
index 3302ac5..4f1eb39 100644
--- a/lab1/fact.cpp
+++ b/lab1/fact.cpp
@@ -4,13 +4,14 @@
 void main(){
     int n;
     int total = 1;
+ std::cout << "Enter the number" << std::endl;
     std::cin >> n;

     for (int i = 1; i <=n; i++){
         total *=i;
     }

- std::cout << total <<std::endl;
+ std::cout <<"Calculated factorial: " << total <<std::endl;
 }
}
```

Лабораторная работа № 2 «Форматирование и стиль»

1244. Джентльмены

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Вспомним старый анекдот:

Говорит как-то один джентльмен другому:

— *А не перекинуться ли нам в картишки?*

— *Знаете, я не играл уже лет десять...*

— *А я — лет пятнадцать...*

Так, слово за слово, решили они вспомнить молодость. Первый джентльмен попросил лакея принести колоду карт, и, перед тем, как начать раздачу, взвесил её в руке.

— *Мне кажется, здесь не хватает одной карты...* — начал он и передал колоду другому джентльмену.

— *Да, девятки тих,* — подтвердил тот.

Дана неполная колода карт. Ваша программа должна определить, какие карты в колоде отсутствуют.

Исходные данные

В первой строке дано целое положительное число, задающее вес данной неполной колоды в миллиграммах. Во второй строке — целое число N , $2 \leq N \leq 100$ — количество карт в полной колоде. Далее следует N строк, в каждой из которых записано целое число от 1 до 1000 — вес очередной карты в миллиграммах. Гарантируется, что суммарный вес всех карт в полной колоде превосходит вес неполной колоды.

Результат

Если задача не имеет решения, выведите единственное число 0. Если решение существует, но не единственно, то выведите -1. Наконец, если можно однозначно восстановить, каких карт недостаёт в неполной колоде по сравнению с полной, то следует через пробел вывести номера отсутствующих карт в порядке возрастания.

Примеры

исходные данные	результат
270 4 100 110 170 200	2 4
270 4 100 110 160 170	-1
270 4 100 120 160 180	0

Автор задачи: Александр Петров

Источник задачи: Ural State University Personal Programming Contest, March 1, 2003

Метки: [динамическое программирование](#) ([скрыть метки для нерешенных задач](#))

2. Алгоритм выборки ответов:

Для каждой карты проверяем недостающий вес и находим решение.

3.

```
#include <cstdio>
#include <cstring>
#include <iostream>

using namespace std;

int main() {
    int weight, n;
    int w[100], dp[100001], choice[100001];
    cin >> weight >> n;
    weight *= -1;
    for (int i = 0; i < n; i++) {
        cin >> w[i];
        weight += w[i];
    }
    memset(dp, 0, sizeof(dp));
    memset(choice, 0, sizeof(choice));
    dp[0] = 1;
    choice[0] = -1;
    for (int i = n - 1; i >= 0; i--) {
        for (int j = weight; j >= w[i]; j--) {
            if (dp[j - w[i]] != 0) {
                dp[j] += dp[j - w[i]];
                if (choice[j] == 0) {
                    choice[j] = i + 1;
                }
            }
        }
    }
    if (dp[weight] >= 2) {
        cout << "-1" << endl;
    } else if (dp[weight] == 0) {
        cout << "0" << endl;
    } else {
        while (weight != 0) {
```

```

    cout << choice[weight] << " ";
    weight -= w[choice[weight] - 1];
}
cout << endl;
}
system("PAUSE");
return 0;
}

```

4.

ID	Дата	Автор	Задача	Язык	Результат проверки	№ теста	Время работы	Выделено памяти
7727300	21:28:45 24 янв 2018	Tomazis	1244. Джентльмены	Visual C++ 2017	Accepted		0.015	1 064 КБ

Лабораторная работа № 3 «TDD: Разработка через тестирование»

1.

1247. Проверка последовательности

Ограничение времени: 0.5 секунды

Ограничение памяти: 64 МБ

Дана последовательность целых чисел A_1, A_2, \dots, A_S и целое положительное число N . Известно, что все элементы последовательности $\{A_i\}$ удовлетворяют ограничению $0 \leq A_i \leq 100$. Кроме того, известно, что сумма всех элементов последовательности равна $S + N$. Напишите программу, которая для заданной последовательности $\{A_i\}$ и числа N дает ответ на вопрос: верно ли, что при любых $1 \leq i \leq j \leq S$ выполняется неравенство

$$A_i + A_{i+1} + \dots + A_j \leq (j - i + 1) + N?$$

Исходные данные

В первой строке через пробел даны числа S и N , каждое из которых положительно и не превосходит 30000. Далее следует S строк, в которых, по одному числу в строке, записаны элементы последовательности $\{A_i\}$.

Результат

Выведите слово «YES», если указанное выше неравенство выполняется при всех значениях параметров i и j , и слово «NO» в противном случае.

Примеры

исходные данные	результат
4 3 2 3 0 2	YES
4 5 1 0 5 3	NO

Автор задачи: Александр Мироненко

Источник задачи: Ural State University Personal Programming Contest, March 1, 2003

Метки: нет ([скрыть метки для нерешенных задач](#))

2. Для каждого числа в последовательности, проверяем если $\text{sum} - i - mn > N$, если меньше, то неравенство выполнено для этой последовательности.

3. Возможные ошибочные ситуации:

1. Если пользователь введет отрицательное число.

2. Если пользователь введет не число.

4.

ID	Дата	Автор	Задача	Язык	Результат проверки	№ теста	Время работы	Выделено памяти
7738239	20:40:57 30 янв 2018	Tomazis	1247. Проверка последовательности	Visual C++ 2017	Accepted		0.031	268 КБ

5.

```

int main() {
    int S, N;
    vector<int> seq;
    int mn, sum;
    for (int k = 0; k <= 1; k++) {
        switch (k)
        {
            case 0:
                S = 4;
                N = 3;
                seq = { 2,3,0,2 };
                break;
            case 1:
                S = 4;
                N = 5;
                seq = { 1,0,5,3 };
                break;
            default:
                break;
        }
        mn = 0;
        sum = 0;
        //cin >> S >> N;

        bool answer = true;

        for (int i = 1; i <= S; i++) {
            //cin >> x;
            sum += seq[i - 1];

            if (sum - i - mn > N) answer = false;

            mn = min(mn, sum - i);
        }

        if (answer) {
            cout << "YES" << endl;
        }
        else {
            cout << "NO" << endl;
        }
    }
    return 0;
}

```

Coverage report			
Name	Total lines	Covered lines	Coverage %
consoleapplicati	26	26	100

6.

```

#include <stdio.h>
#include <tchar.h>
#include <algorithm>
#include <cstring>
#include <iostream>
#include <vector>

```

```
using namespace std;

int main() {
    int s, n;
    int mn = 0, sum = 0;
    int x;
    bool answer = true;
    cin >> s >> n;
    for (int i = 1; i <= s; i++) {
        cin >> x;
        sum += x;
        if (sum - i - mn > n) {
            answer = false;
        }
        cout << mn << " " << sum - i << endl;
        mn = min(mn, sum - i);
    }
    if (answer) {
        cout << "YES" << endl;
    } else {
        cout << "NO" << endl;
    }
    return 0;
}
```

Задание. Подсчет количества и характеристик зерен по фотографии.

Требуется подсчитать количество зерен и измерить их характеристики по фотографии.

Для решения данной задачи используется библиотека для работы с изображениями на C++ OpenCV. Для GUI используется Qt 5.11.

Описание алгоритма.

При загрузке изображения opencv хранит изображение в палитре BGR.

Первым шагом будет перевод изображения в формат HSV.

Из этого формата нам нужен только Saturation.

Далее нужно сделать границы объектов изображения более четкими применив laplacian filter.

Потом нужно подобрать диапазон цветов для выделения зерен.

После чего производится размытие и эрозия с расширением изображения.

Теперь наше изображение готово к началу обработки по методу Watershed.

Проводим сегментацию изображения, превращая его в бинарное.

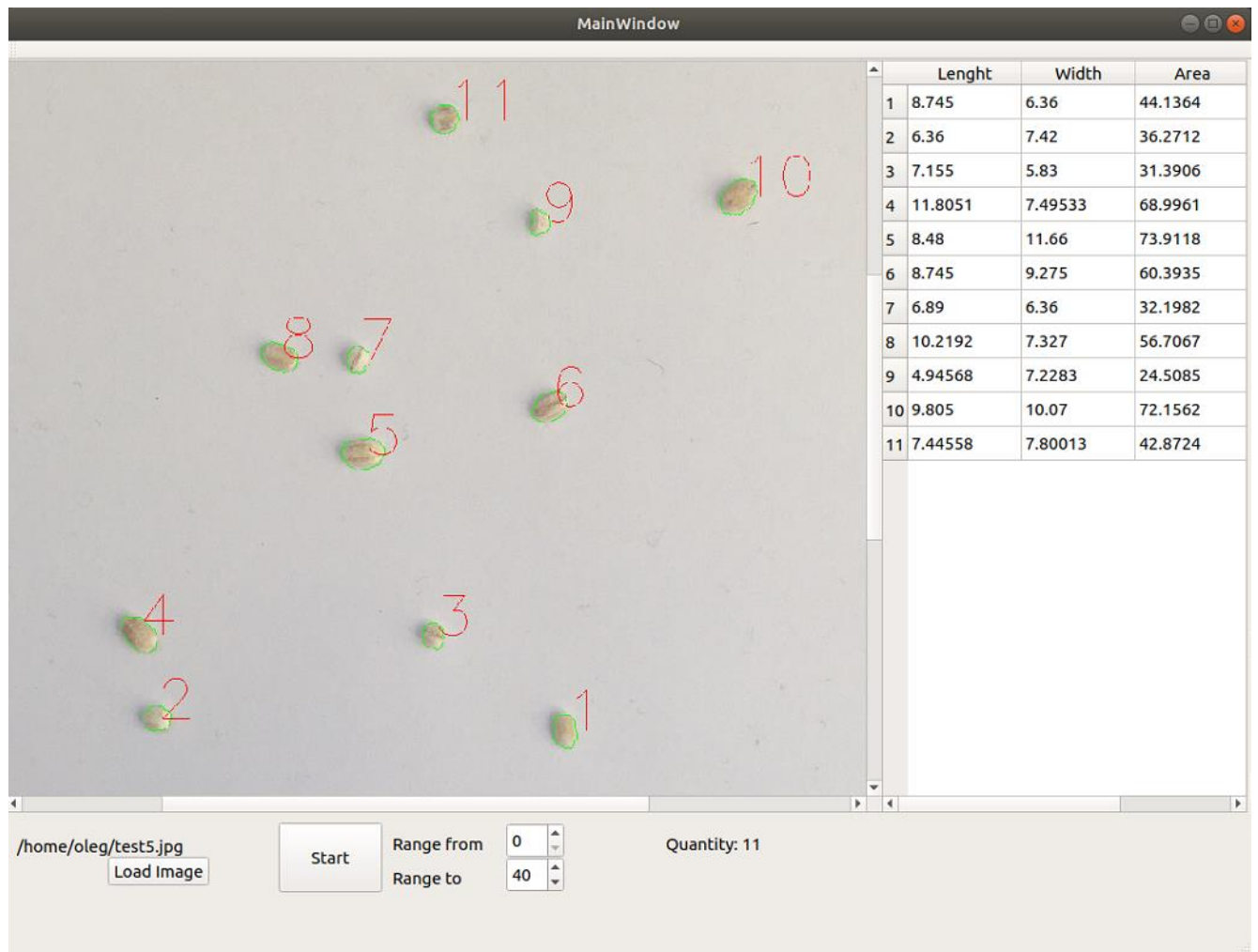
Следует провести opening transformation для убирания мелких объектов.

Последним этапом будет выделения заднего и переднего плана. При вычитании планов можно выделить тонкие контуры объектов, что нам идеально подходит.

Остается создать маркеры и провести watershed. Места для закрашивания будут помечены -1.

Для того чтобы пронумеровать и подсчитать характеристики зерен нужно создать у каждого контура прямоугольник. Подчитать его высоту и ширину, приписать рядом с ним номер. Для площади можно высчитать встроенной функцией площадь контура.

Пример работы программы.



Приложение А. Слайды презентации с семинара по C++11 на тему:
«Регулярные выражения»

Regex Регулярные выражения

Кириллов Олег
Группа V3316

Использование regex

- Regex используется в таких программах как
 - Фильтр спама
 - Программы распознавания языка
- Доступен на множестве языков
 - В Linux можно увидеть в программах, например grep/egrep, sed, awk
- Поддерживаемые форматы
 - basic
 - extended
 - ECMAScript
 - awk
 - grep
 - egrep

Пример

Нам нужно сопоставить строку Viagra найденную в спам-письме

- Мы можем пройти по всем вариантам
- Умные спаммеры найдут пути выразить Viagra различными способами отличными от изменения регистра
 - Хотя спам-фильтры могут не поймать слово, если оно будет просто проверять на нижние и верхние регистры букв
 - Но человек может понять
- Вот несколько примеров слов которые regex может поймать
 - v.i.a.g.r.a, vIagra, vi_ag_ra, vi@gr@, ViAgRa, Viagr

Metacharacters

Metacharacter	Explanation
*	Match the preceding character if it appears 0 or more times
+	Match the preceding character if it appears 1 or more times
?	Match the preceding character if it appears 0 or 1 times
.	Match any one character
^	Match if this expression begins a string
\$	Match if this expression ends a string
[chars]	Match if the next character in the string contains any of the chars in []
[char ₁ -char _j]	Match if the next character in the string contains any characters in the range from char ₁ to char _j
[[:class:]]	Match if the next character in the string is a character that is part of the :class: specified. The :class: is a category like upper case letters (:upper:) or digits (:digit:) or punctuation marks (:punct:).
[^chars]	Match if the next character in the string is not one of the characters listed in []
\	The next character should be interpreted literally, used to escape the meaning of a metacharacter
{n}	Match if the string contains n consecutive occurrences of the preceding character
{n,m}	Match if the string contains between n and m consecutive occurrences of the preceding character
{n,}	Match if the string contains at least n consecutive occurrences of the preceding character
	Match any of these strings (an "OR")
(...)	The items in ... are treated as a group, match the entire sequence

regex в C++

Основные функции regex в C++

- `regex_match`
- `regex_search`
- `regex_replace`

Основные экземпляры `match_results`

- `smatch` – для строковых литералов
- `wsmatch` – для широких строковых литералов
- `smatch` – для строковых объектов
- `wsmatch` – для широких строковых объектов

regex_match и regex_search

```
std::string str = "Hello world";
```

Создадим регулярное выражение

```
std::tr1::regex rx("ello");
```

Выражение

```
regex_match(str.begin(), str.end(), rx)
```

Вернет false потому что, строка `str` содержит больше символов, чем регулярное выражение `rx`

```
regex_search(str.begin(), str.end(), rx)
```

Вернет true потому что, `rx` совпадает с подстрокой `str`

regex_match и regex_search

```
std::string str = "Hello world";
```

Создадим регулярное выражение

```
std::tr1::regex rx("ello");
```

Выражение

```
regex_match(str.begin(), str.end(), rx)
```

Вернет false потому что, строка str содержит больше символов, чем регулярное выражение rx

```
regex_search(str.begin(), str.end(), rx)
```

Вернет true потому что, rx совпадает с подстрокой str

Как получить результат

C++ сохраняет результат в match_result, но для этого вам нужно создать объект по шаблону

```
typedef match_results<const char*> cmatch
```

```
std::tr1::cmatch res;
```

```
str = "<h2>Egg prices</h2>";
```

```
std::tr1::regex rx("<h(.)>([^\<]+)");
```

```
;
```

```
std::tr1::regex_search(str.c_str(), res, rx);
```

```
std::cout << res[1] << ". " << res[2] << "n";
```

Код выше выведет

```
2. Egg prices
```

regex_replace

```
std::string str = "Hello world";  
std::tr1::regex rx("world");  
std::string replacement = "planet";  
std::string str2 = std::tr1::regex_replace(str, rx, replacement);
```

В этом коде слово "world" в строке "Hello world" словом "planet"

Источники

https://www.johndcook.com/blog/cpp_regex/

<http://www.cplusplus.com/reference/regex/>