

Lecture 11. C++ <String>

```
std::string str2 = " ";
std::string str = "ABCDE";
```

```
for (size_t i=0; i < str.size(); i++) {
    std::cout << str[i] << " ";
}
```

Output => A B C D E

Этот определение языка предназначено для перебора всех элементов в "str".

- "size_t"- в стандартной библиотеке C++ тип определяется как беззнаковый тип, который способен представить размер самого большого возможного объекта в системе. Обычно используется для индексации и измерения размеров в контексте работы с массивами, строками и так далее.

```
for (size_t i=0; i < str.size(); i++) {
    std::cout << (int)str[i] << " ";
}
```

Output => 65 66 67 68 69

Делаем все тоже самое что и предыдущий, но выводим уже значения элементов по таблице ASCII.

// size() // length()

```
std::cout << str.size() << " " << str.length();
----- -----
```

Она возвращает один и тот же результат

Output => 5 5

// empty()

```
std::string str2 = " ";
std::string str = "ABCDE";
```

```
std::cout << str.empty() << " " << str2.empty();
```

проверяет наличие элементов в строке.

Output => 0 1

```
Str = "ABC";
Str += "D";
std::cout << Str; Output => ABCD
```



добавляет элементы
к концу строки

```
// push-back()
Str::push-back("E");
std::cout << Str; Output => ABCEDE
```

```
// pop-back()
Str::pop-back();
std::cout << Str; Output => ABCD
```

Удаляем последний элемент строки.

```
// append()
std::string Str = "data";
std::string str2 = "base";
str.append(str2);
std::cout << str;
Output => database
```

std::cout << str[4] << " " << str.at[4]

отображает элемент под индексом 4

ABCDE

Output => E E

// clear()

Str.clear();

Нельзя забывать что эта функция удаляет
все элементы со строки.

Что бы избавиться от лишних операций ранее было выделено для str
много бесполезовавшей памяти str.shrink-to-fit();

```
// insert()
std::string str2 = "";
std::string str = "ABCDE";
```

```
str2.clear();
str2 = "XYZ";
str.insert(3, str2);
std::cout << str; Output => ABCXYZDE
```

Используется для вставки символов
или подстроки в определенное место
текущей строки.

Еще примеров к insert();

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

```
str.insert(2, 1, "X"); // Вывод: HeXllo
```

```
str.insert(3, "World"); // Вывод: HeXWorldllo
```

```
str.insert(3, "University", 5); // Вывод: HeXUniveWorldllo
```

// erase()

```
str = "ABCDEF";
```

```
str.erase(3, 1); // Вывод: ABDEF
```

// Удаление 1 символа по индексу 3

```
str.erase(1, 3); // Вывод: AF
```

// Удаление подстроки, начиная с индекса 1, длиной 3 символа

```
str.erase(0); // Вывод: "
```

// Удаление всех символов после 0

// replace()

Используется для замены подстроки в строке другой подстрокой.

```
str = "Hello, World!";
```

```
str.replace(str.find("World"), 5, "Universe"); // Вывод: Hello, Universe!
```

```
str.replace(7, std::string::npos, "Everyone"); // Вывод: Hello, Everyone
```

// getline()

Используется для считывания строки из входного потока (например из std::cin) и сохраняет её в переменную типа 'std::string'.

Она получает типами строки до символа новой строки ('\n') или до достижения конца файла.

```
std::getline(std::sin, str, '/');
```

// при вводе: get /line , будет считан только "get"
вместо "/" можно подставить что угодно.

std::getline(входной_поток; строка; разделитель);