Лабораторная работа 10. Строки с нуль-окончанием

строки с нуль-окончанием (справочник: https://ru.cppreference.com/w/cpp/string/byte)
http://cppstudio.com/cat/309/325/?ysclid=mhllwjjvpa875224501

1. Создать три переменных для хранения строк с нуль-окончанием длиной до 20 символов (char s[21] …)
Ввести в них имя, отчество и фамилию человека.
Программа должна в трех новых переменных того же типа сформировать и вывести три строки, содержащих:
a) фамилию, имя и отчество, разделенные пробелами
b) только инициалы, разделенные точками,
c) полную фамилию и инициалы имени и отчества, все заглавными буквами

2. Ввести с клавиатуры одну строку (string)
a) вывести количество символов ‘S’ и ‘s’ (по отдельности и общее их количество)
b) вывести количество цифр в этой строке.
c) вывести количество заглавных букв в строке
d) сформировать и вывести строку, в которой заменить в исходной строке все вхождения ‘1’ на ‘5’, ‘a’ на ‘A’
e) за один вызов printf() вывести все результаты для пунктов a),b),c), d) в представленной ниже форме:
Пример:

```
Исходная строка = “123abc-S-s-+++SSSaaaAAAssss5678”,
количество S = 4,
количество s = 5,
количество s и S = 9,
количество цифр = 7,
количество заглавных букв = 7
новая строка = “523Abc-S-s-+++SSSAAAAAAssss5678”
```

3. Создать и протестировать функции, выполняющие для строки следующие операции:
Сформировать и вывести новую строковую переменную, для которой

a) заменить все буквосочетания "*min*" на "*max*"
b) заменить на "ku-ku" все слова из пяти символов (последовательности 5 символов, ограниченные пробелами)
c) если исходная строка включает подстроку "ku-ku",
то каждое такое вхождение заменить его на "ku-ka-re-ku"
если нет, то вставить " ku-ku ku-ku ku-ku" в середину строки.
d) - удалить все подстроки “ku-ku”
e) - удалить все символы до конца строки, стоящие после последнего «ku-ku»
f) количество слов в строке
между словами может быть несколько пробелов или любые знаки пунктуации
g) имеется ли в линейной записи заданной математической формулы баланс открывающих и закрывающих скобок.
Например, "(a+b))" – нет баланса, "(a+b/(c*d))" – есть баланс
h) Удалить из введенной строки все последовательности символов, заключенные в круглые скобки, вместе со скобками. Вывести полученную строку

4. Преобразовать число, заданное строкой в римской системе счисления, в число десятичной системы

Функции `<cstring>`

Название	Описание Заголовок функции	Пример кода
strlen	Длина строки `size_t strlen(const char* str);`	`const char* str = "Hello";` `size_t len = strlen(str); // 5`

strcpy	Копирование строки
	`char* strcpy(char* dest, const char* src);`
		`char src[] = "Hello";`
		`char dest[10];`
		`strcpy(dest, src);`

strncpy	Копирование n символов
	`char* strncpy(char* dest, const char* src, size_t count);`
		`char src[] = "Hello";`
		`char dest[10];`
		`strncpy(dest, src, 3);`
		`dest[3] = '\0';`

strcat	Конкатенация строк
	`char* strcat(char* dest, const char* src);`
		`char str1[20] = "Hello";`
		`char str2[] = " World";`
		`strcat(str1, str2);`

strncat	Конкатенация n символов
	`char* strncat(char* dest, const char* src, size_t count);`
		`char str1[20] = "Hello";`
		`char str2[] = " World";`
		`strncat(str1, str2, 3);`

strcmp	Сравнение строк `int strcmp(const char* lhs, const char* rhs);`	`int result = strcmp("apple", "banana");` `// result < 0`
strncmp	Сравнение n символов `int strncmp(const char* lhs, const char* rhs, size_t count);`	`int result = strncmp("apple", "application", 3);` `// result == 0`
strchr	Поиск первого вхождения символа `char* strchr(const char* str, int ch);`	`char* pos = strchr("Hello", 'e');` `// pos указывает на "ello"`

Название	**Описание** **Заголовок функции**	**Пример кода**
strrchr	Поиск последнего вхождения символа `char* strrchr(const char* str, int ch);`	`char* pos = strrchr("Hello", 'l');` `// pos указывает на "lo"`
strstr	Поиск подстроки `char* strstr(const char* str, const char* target);`	`char* pos = strstr("Hello World", "World");` `// pos указывает на "World"`

strtok	Разбиение строки на токены
	`char* strtok(char* str, const char* delim);`
		`char str[] = "a,b,c";`
		`char* token = strtok(str, ",");`
		`while(token) { /* обработка */`
		`token = strtok(nullptr, ","); }`

memset	Заполнение памяти
	`void* memset(void* dest, int ch, size_t count);`
		`char buffer[10];`
		`memset(buffer, 'A', 5);`
		`buffer[5] = '\0'; // "AAAAA"`

memcpy	Копирование памяти
	`void* memcpy(void* dest, const void* src, size_t count);`
		`char src[] = "Hello";`
		`char dest[10];`
		`memcpy(dest, src, 6); // копирует 6 байт`

memmove	Безопасное копирование (с перекрытием) `void* memmove(void* dest, const void* src, size_t count);`	`char str[] = "Hello";` `memmove(str + 2, str, 3); // "HeHel"`

memcmp	Сравнение областей памяти
	`int memcmp(const void* lhs, const void* rhs, size_t count);`
		`char a[] = "apple";`
		`char b[] = "apple";`
		`int result = memcmp(a, b, 5); // 0`

Функции `<cctype>`

Название	Описание Заголовок функции	Пример кода
isalnum	Проверяет, является ли символ буквенно-цифровым `int isalnum(int ch);`	`char c = 'A';` `if(isalnum(c)) { /* true */ }`
isalpha	Проверяет, является ли символ буквой `int isalpha(int ch);`	`char c = 'z';` `if(isalpha(c)) { /* true */ }`
islower	Проверяет, является ли символ строчной буквой `int islower(int ch);`	`char c = 'a';` `if(islower(c)) { /* true */ }`
isupper	Проверяет, является ли символ заглавной буквой `int isupper(int ch);`	`char c = 'Z';` `if(isupper(c)) { /* true */ }`
isdigit	Проверяет, является ли символ цифрой `int isdigit(int ch);`	`char c = '5';` `if(isdigit(c)) { /* true */ }`
isxdigit	Проверяет, является ли символ шестнадцатеричной цифрой `int isxdigit(int ch);`	`char c = 'F';` `if(isxdigit(c)) { /* true */ }`
iscntrl	Проверяет, является ли символ управляющим `int iscntrl(int ch);`	`char c = '\n';` `if(iscntrl(c)) { /* true */ }`
isgraph	Проверяет, является ли символ печатным (кроме пробела) `int isgraph(int ch);`	`char c = '!';` `if(isgraph(c)) { /* true */ }`

Название	Описание Заголовок функции	Пример кода
isspace	Проверяет, является ли символ пробельным `int isspace(int ch);`	`char c = ' ';` `if(isspace(c)) { /* true */ }`
isblank	Проверяет, является ли символ пробелом или табуляцией `int isblank(int ch);`	`char c = '\t';` `if(isblank(c)) { /* true */ }`
isprint	Проверяет, является ли символ печатным `int isprint(int ch);`	`char c = 'A';` `if(isprint(c)) { /* true */ }`
ispunct	Проверяет, является ли символ знаком пунктуации `int ispunct(int ch);`	`char c = '.';` `if(ispunct(c)) { /* true */ }`
tolower	Преобразует символ в нижний регистр `int tolower(int ch);`	`char c = 'A';` `char lower = tolower(c); // 'a'`
toupper	Преобразует символ в верхний регистр `int toupper(int ch);`	`char c = 'a';` `char upper = toupper(c); // 'A'`

Примеры использования функций:

Проверка категорий символов

cpp

```cpp
#include <cctype>
#include <iostream>

int main() {
    char str[] = "Hello123!";

    for(int i = 0; str[i] != '\0'; i++) {
        if(isalpha(str[i]))
            std::cout << str[i] << " - буква\n";
        else if(isdigit(str[i]))
            std::cout << str[i] << " - цифра\n";
        else if(ispunct(str[i]))
            std::cout << str[i] << " - пунктуация\n";
    }
    return 0;
}
```

Преобразование регистра строки

cpp

```cpp
#include <cctype>
#include <iostream>

int main() {
    char str[] = "Hello World!";

    // В верхний регистр
```

```cpp
    for(int i = 0; str[i] != '\0'; i++) {
        str[i] = toupper(str[i]);
    }
    std::cout << str << std::endl; // "HELLO WORLD!"

    // В нижний регистр
    for(int i = 0; str[i] != '\0'; i++) {
        str[i] = tolower(str[i]);
    }
    std::cout << str << std::endl; // "hello world!"

    return 0;
}
```

Анализ строки

cpp

```cpp
#include <cctype>
#include <iostream>

int main() {
    char text[] = "Hello, World 123!";
    int letters = 0, digits = 0, spaces = 0, punct = 0;

    for(int i = 0; text[i] != '\0'; i++) {
        if(isalpha(text[i])) letters++;
        else if(isdigit(text[i])) digits++;
        else if(isspace(text[i])) spaces++;
        else if(ispunct(text[i])) punct++;
    }

    std::cout << "Букв: " << letters << std::endl;
    std::cout << "Цифр: " << digits << std::endl;
```

```
    std::cout << "Пробелов: " << spaces << std::endl;
    std::cout << "Знаков пунктуации: " << punct << std::endl;

    return 0;
}
```

Проверка шестнадцатеричных цифр

cpp

```
#include <cctype>
#include <iostream>

bool isValidHex(const char* str) {
    for(int i = 0; str[i] != '\0'; i++) {
        if(!isxdigit(str[i])) {
            return false;
        }
    }
    return true;
}

int main() {
    std::cout << std::boolalpha;
    std::cout << "123ABC: " << isValidHex("123ABC") << std::endl; // true
    std::cout << "12G45F: " << isValidHex("12G45F") << std::endl; // false
    return 0;
}
```

1. **Тип параметра**: Функции принимают `int`, но обычно используются с `char`
2. **Возвращаемое значение**: Ненулевое значение означает истину, 0 - ложь
3. **Локализация**: Поведение зависит от текущей локали
4. **Безопасность**: Функции безопасны и не имеют побочных эффектов
5. **Диапазон**: Параметр должен быть представлен как `unsigned char` или EOF