Рядки. Функції по роботі з рядками.

Мета роботи.

Метою даної лабораторної роботи є отримання навиків по роботі з рядками шляхом використання засобів стандартної бібліотеки для обробки рядків мовою С++.

Короткі теоретичні відомості до роботи.

Оголошення рядків

Символи — це фундаментальні базові одиниці програм. Програми можуть містити символьні константи, які є насправді цілими значеннями, представленими як знак в одинарних лапках. Значення символьної константи — це ціле значення відповідного знаку в машинному наборі символів. Наприклад, символьна константа ‘a’ має своїм внутрішнім представленням ціле число 97, оскільки код літери ‘a’ має саме це значення.

З окремих символів в мові С можна складати рядки, взявши символи в подвійні лапки. При цьому рядок буде розглядатись як одне ціле, ще одна одиниця мови. Наприклад, вираз "Hello, World!" є рядком.

В мові С рядки реалізовані як масив символів, який закінчується символом з кодом 0. Таким чином, рядок фактично є покажчиком на перший свій символ. В цьому сенсі, рядок подібний до масивів.

Рядок може бути оголошений або як масив символів (char []), або як покажчик на об’єкти типу char (char*).

Наступні оголошення створюють рядкові змінні та ініціалізують їх значенням "Ivan".

```
char name[]="Ivan";
char nameptr="Ivan";
```

Перша декларація створює масив name з п’яти комірок, що містить елементи 'I', 'v', 'a', 'n' та закінчується нульовим символом.

Друга декларація створює змінну nameptr типу “покажчик на char”, що вказує на рядок "Іван", позиція якого в пам’яті явно програмістом не визначається (таку відповідальність бере на себе компілятор).

Попередній опис масиву еквівалентний за своїм результатом програмному коду:

```
char name[]={'I','v','a','n', '\0'};
```

Тут запис '\0' визначає символьну константу, яка відповідає символу з кодом 0.

Тип рядкового літерала є «масив з належною кількістю константних символів». Рядковий літерал можна присвоїти змінної типу *char**. Це дозволяється, тому що в попередніх визначеннях C і C++ типом рядковоаго літерала був *char**. Однак зміна рядковоаго літерала через такий покажчик є помилкою.

```
char *str = "C & C++";
str[2] = '?';                    // Помилка під час виконання!
```

Те, що рядкові літерали є константами, є не тільки очевидним, але й дозволяє при реалізації провести значну оптимізацію методів зберігання і доступу до рядкових літералів. Якщо ж потрібнен рядок, який можна змінювати, слід оголосити та ініціалізувати масив символів.

```
char str[] = "C & C++";          // Масив з 8 символів
str [2] = '?';                   // Правильно
```

Пам'ять під рядкові літерали виділяється статично, тому їх можна повертати в якості значення функції.

```
const char* error_massage()
{
      // Після виходу з функції пам'ять, що містить рядок, не буде звільнено
      return "Недостатньо параметрів";
}
```

Оскільки ми не знаємо, скільки в рядку міститься символів, але знаємо, що в кінці стоїть символ кінця рядка, цикл для обробки рядка пишеться наступним чином:

```
for (int i = 0; str[i] != '\0'; i++) {...}
```

Можна опустити порівняння з нулем, для C++ це буде еквівалентно:

```
 for (int i = 0; str[i]; i++) {...}
```

Можна використовувати покажчики для обробки рядків:

```
char str [50];
for (char *p = str; *p; p++) {...}
```

Приклад 1. Функція, яка змінює всі входження букви «z» на «a», «a» - на «b», «b» - на «c» і т.д. Інші символи залишаються без змін.

```
char* Change(char *str)

{
      for (char *p = str; *p; p++)
            if (*p == 'z')
                  *p = 'a';
            else if ('a' <= *p && *p <= 'b')
                  (*p)++;
      return str;
}
```

Приклад 2. Функція формує рядок (dest), що складається з символів вхідного рядка (source), що не входять до заданого переліку (symbols)

```
char* NotEntered(char *dest, const char *source, const char *symbols)

{
	int i, j;

	for (i = 0, j = 0; source[i]; i++)
		if (!strchr(symbols, source[i]))
			dest[j++] = source[i];
	dest[j] = '\0'; // Обов'язково додаємо ознаку кінця рядка
	return dest;
}
```

Функції для обробки рядків

Мова С має достатню кількість бібліотечних функцій (макросів) для обробки рядків та символів, щоб зробити створення відповідних програмних кодів комфортним для програмістів.

Наступна таблиця демонструє головні засоби для роботи з окремими символами.

isdigit	Повертає істинне значення, якщо аргумент є цифрою, інакше повертає значення “неправда”
isalpha	Повертає істинне значення, якщо аргумент є літерою, інакше повертає значення “неправда”
isalnum	Повертає істинне значення, якщо аргумент є цифрою або літерою, інакше повертає значення “неправда”
isxdigit	Повертає істинне значення, якщо аргумент є шістнадцятковою цифрою, інакше повертає значення “неправда”
islower	Повертає істинне значення, якщо аргумент є літерою в нижньому регістрі, інакше повертає значення “неправда”
isupper	Повертає істинне значення, якщо аргумент є літерою у верхньому регістрі, інакше повертає значення “неправда”
tolower	Якщо аргумент є літерою у верхньому регістрі, конвертує її в літеру в нижньому регістрі, інакше залишає символ без змін
toupper	Якщо аргумент є літерою в нижньому регістрі, конвертує її в літеру у верхньому регістрі, інакше залишає символ без змін
isspace	Повертає істинне значення, якщо аргумент є пробільним символом, інакше повертає значення “неправда”
iscntrl	Повертає істинне значення, якщо аргумент є контрольним символом, інакше повертає значення “неправда”

Наступний приклад показує, як можна застосувати такі функції, щоб перетворити шістнадцятеричні символи на відповідні числові значення.

```
char hex2dec( char ch )
{
    if( isdigit(ch) )
        return ch - '0';
    else if( isupper(ch) )
        return ch - 'A' + 10;
    else if( islower(ch) )
        return ch - 'a' + 10;
    return 0;
}
```

В наступній таблиці наведено деякі функції, які конвертують рядки в значення інших типів даних.

atof	Конвертує рядок в значення типу double.
atoi	Конвертує рядок в значення типу int.
atol	Конвертує рядок в значення типу long
strtod	Конвертує рядок в значення типу double.
strtol	Конвертує рядок в значення типу long.
strtoul	Конвертує рядок в значення типу unsinged int.

Оскільки ми не можемо безпосередньо порівнювати або додавати рядки засобами мови С, існують бібліотечні функції, які вирішують такі завдання. Деякі з таких функцій наведені в наступній таблиці.

strcpy	Копіювання рядків
strncpy	Копіювання n символів одного рядка в інший
strcat	Склеювання рядків
strncat	Додання n символів одного рядка до іншого
strcmp	Порівняння рядків
strncmp	Порівняння n символів одного рядка з іншим
strchr	Пошук першого входження визначеного аргументом символу до рядку
strstr	Пошук першого входження визначеного аргументом рядку до іншого рядку
strlen	Визначення довжини рядку

Порядок виконання роботи.

1. Створити проект, який містить консольну програму Win32.
2. Зчитати з клавіатури два довільні рядки, обрати та виконати наступні операції над цими рядками з використанням функцій стандартної бібліотеки:

a. на оцінку задовільно – копіювання рядків, копіювання n символів одного рядка в інший, склеювання рядків, додання n символів одного рядка до іншого, порівняння рядків, порівняння n символів одного рядка з іншим;
b. на оцінку добре – пошук першого входження визначеного аргументом символу до рядку, пошук першого входження визначеного аргументом рядку до іншого рядку.

3. Без використання функцій стандартної бібліотеки виконати завдання згідно варіанту до роботи, таблиця 1 (завдання на оцінку відмінно).

УВАГА:

Обов'язковими для виконання є пункти які не помічено відмітками: «для отримання оцінки задовільно», «для отримання оцінки добре» та «для отримання оцінки відмінно».

Оцінка задовільно ставиться у випадку виконання всіх обов'язкових пунктів роботи та пункту який має відмітку «для отримання оцінки задовільно».

Оцінка добре ставиться у випадку виконання всіх обов'язкових пунктів роботи та пунктів які мають відмітку «для отримання оцінки задовільно» та «для отримання оцінки добре».

Оцінка відмінно ставиться у випадку виконання всіх обов'язкових пунктів роботи та пунктів які мають відмітку «для отримання оцінки задовільно», «для отримання оцінки добре» та «для отримання оцінки відмінно».

Варіанти до роботи.

Таблиця 1. Перелік завдань до пункту 3.

№	Завдання
1.	Підрахувати кількість слів в рядку
2.	Виділити перші n слів з рядка
3.	Порівняти два рядки, ігноруючи відмінності в регістрах
4.	Розбити рядок на дві частини: до першого входження заданого символу і після нього
5.	Вирівняти рядок по лівому краю до заданої довжини
6.	Скопіювати рядок в інший рядок заданої довжини і розмістити текст першого рядка по центру
7.	Видалити з рядка задану кількість символів, починаючи із заданої позиції
8.	Визначити в рядку номер позиції слова із заданим номером
9.	Замінити символи рядка з одного заданого алфавіту на символи іншого алфавіту
10.	Знайти останнє входження в рядок заданого підрядка
11.	Довести довжину рядка до заданої, вставляючи пропуски між словами
12.	Знайти в рядку перший символ, який входити в інший заданий рядок
13.	Видалити з рядка слово із заданим номером
14.	Замінити символи одного рядка заданою кількістю символів іншого рядка, починаючи із заданої позиції
15.	Знайти в рядку перший символ, який не входити в інший заданий рядок
16.	Порівняти два рядки, ігноруючи кількість пропусків між словами
17.	Видалити з початку і з кінця рядка заданий символ
18.	Виділити з рядка задану кількість слів, починаючи із слова із заданим номером
19.	Вставити в рядок інший рядок, починаючи із заданої позиції
20.	Знайти перше входження в рядок заданого підрядка
21.	Дописати один рядок у початок іншого
22.	Замінити в рядку всі множинні входження заданої комбінації символів одним символом
23.	Замінити в рядку одну задану комбінацію символів іншою заданою комбінацією
24.	Переписати всі символи рядка в зворотному порядку
25.	Визначити довжину слова із заданим номером
26.	Виділити із заданого рядка підрядок заданої довжини, починаючи із заданої позиції
27.	Розбити рядок на декілька рядків кожен з яких містить слово з початкового рядка
28.	Переставити всі сусідні слова у рядку місцями (попарно)
29.	Додати до початку та кінця кожного слова в рядку символи “<” та “>” відповідно.
30.	Надрукувати на екран всі слова рядку довжина яких більше ніж n та менше m