Лабораторна робота № 9

СТРІЧКИ.

Мета роботи:

Вивчити поняття і застосування стрічок.

Обладнання та програмне забезпечення:

– IBM сумісна персональна обчислювальна машина;

– онлайн компілятор мови програмування Сі, доступний за посиланням https://repl.it/languages/c

Завдання до роботи:

Написати програму на мові програмування Сі, котра використовує наступні дії над стрічками:

– пошук заданої послідовності символів;

– заміну і видалення символів і груп символів;

– порівняння і сортування стрічок;

– знаходження кількості символів у стрічках.

Теоретичні відомості

Стрічки

В Сі **стрічка** визначається як символьний масив довільної довжини, що закінчується нульовим символом. Нульовий символ визначається як '\ 0'.

Тому необхідно оголошувати символьні масиви на один символ більші.

Наприклад, якщо необхідно оголосити масив s, що містить десятисимвольну стрічку, слід написати:

```
char s[11];
```

Стрічкові константи це список символів, взятих в подвійні лапки. Нижче показані дві стрічкові константи:

```
"Hello world"
"This is a string constant"
```

Для роботи зі стрічками використовується модуль `string.h` стандартної бібліотеки мови Сі [1, 2]

Константи і типи модуля `string.h`:

NULL - нульовий (порожній) вказівник, який нікуди не вказує. Використовується для того, щоб показати, що дана змінна-вказівник ні на що не посилається.

size_t - беззнакове ціле, що має той же тип, що і результат оператора sizeof

Функції модуля `string.h`:

Прототип	Пояснення
`void *memcpy(void *dest, const void *src, size_t n);`	копіює *n* байтів з області пам'яті *src* в *dest*, які не повинні перетинатися, у іншому випадку результат невизначений (можливо як правильне копіювання, так і ні)
`void *memmove(void *dest, const void *src, size_t n);`	копіює *n* байтів з області пам'яті *src* в *dest*, які на відміну від `memcpy` можуть перекриватися
`void *memchr(const void *s, char c, size_t n);`	повертає вказівник на перше входження символа *c* в перших *n* байтах *s*, або NULL, якщо не знайдено
`int memcmp(const void *s1, const void *s2, size_t n);`	порівнює перші *n* символів в областях пам'яті s1 та s2
`void *memset(void *, int z, size_t n);`	заповнює область пам'яті одним байтом *z*
`char *strcat(char *dest, const char *src);`	дописує стрічку *src* в кінець *dest*
`char *strncat(char *dest, const char *, size_t n);`	дописує не більше *n* початкових символів рядка *src* (або всю *src*, якщо її довжина менше) у кінець *dest*
`char *strchr(const char *s, int c);`	шукає символ *c* у рядку *s*, починаючи з голови і повертає його адресу, або NULL якщо не знайдений
`char *strrchr(const char *s, int c);`	шукає символ *c* у рядку *s*, починаючи з хвоста і повертає його адресу, або NULL якщо не знайдений
`int strcmp(const char *s1, const char *s2);`	лексикографічне порівняння стрічок (повертає "0", якщо стрічки однакові, додатнє, якщо перша стрічка більша, і від'ємне, якщо менша)
`int strncmp(const char *s1, const char *s2, size_t n);`	лексикографічне порівняння перших *n* байтів стрічок
`int strcoll(const char *s1, const char *s2);`	лексикографічне порівняння двох стрічок відповідно до поточної локалі, визначеної категорією LC_COLLATE.
`char *strcpy(char *to, const char *from);`	копіює стрічку з одного місця в інше
`char *strncpy(char *to, const char *from, size_t n);`	копіює до *n* байт стрічки з одного місця в інше
`char *strerror(int);`	повертає в стрічковому вигляді повідомлення про помилку errno
`size_t strlen(const char *);`	повертає довжину стрічки
`size_t strspn(const char *s, const char *accept);`	повертає к-сть початкових байтів стрічки *s*, які містяться в стрічці *accept*
`size_t strcspn(const char *s, const char *reject);`	повертає к-сть початкових байтів стрічки *s*, які НЕ містяться в стрічці *reject*
`char *strpbrk(const char *s, const char *accept);`	знаходить перше входження у стрічку *s* будь-якого символу, перерахованого в *accept*
`char *strstr(const char *haystack, const char *needle);`	знаходить перше входження рядка *needle* в *haystack*
`char *strtok(char *, const char *);`	перетворює стрічку у послідовність токенів.
`size_t strxfrm(char *dest, const char *src, size_t n);`	створює відтрансльовану копію рядка, таку, що дослівне порівняння її (strcmp) буде еквівалентно порівнянню з коллатором

Приклад №1

```c
#include <string.h>
#include <stdio.h>
int main(void)
{
  char s1[80], s2[80];
  scanf("%s", s1); scanf("%s",s2);
// визначення довжин 2 стрічок
  printf("lengths: %lu %lu\n", strlen(s1), strlen(s2));
  if(!strcmp(s1, s2)) printf("The strings are equal\n");
// об’єднання 2 стрічок
  strcat(s1, s2);
  printf("%s\n", s1);
return 0;
}
```

Приклад №2

```c
#include <string.h>
#include <stdio.h>
int main(void)
{
  char s1[80], s2[80];
  scanf("%s", s1); // ввід слова
  for(int i=0; i<strlen(s1);++i)
    printf("%c ", s1[i]); // посимвольний вивід слова
  printf("\n");
  for(int i=strlen(s1); i>0-1;--i)
    printf("%c ", s1[i]); // посимвольний вивід слова в зворотному порядку
  return 0;
}
```

Приклад №3

```c
#include <stdio.h>
int main(void)
{
  int *p, mas[10];
  p = mas;
  mas[5] = 5;
  p[5] = 100; // Присвоєння за допомогою індексу
  printf("%d\n", mas[5]);
  *(p + 5) = 20; // Присвоєння за вказівником
  printf("%d\n", mas[5]);
  return 0;
}
```

ПОРЯДОК ВИКОНАННЯ РОБОТИ:

1. Опрацювати і засвоїти матеріал наведений в теоретичних відомостях.
2. Написати програму, яка реалізує введення прислів'я або афоризму із 7+ слів однією стрічкою, розбиває її на масив окремих слів із зворотнім порядком символів в словах і виводить їх в окремих рядках.
3. Написати програму, яка реалізує пострічкове введення списку Вашої групи (улюбленої футбольної команди, будь-якого колективу з 10+ учасників) в форматі Номер-[пробіл]-Прізвище-[пробіл]-Ім'я, а потім 1) підраховує і виводить кількість відмінних (різних) імен та середню довжину прізвищ; 2) формує і виводить в алфавітному порядку список виду Ім'я-[пробіл]-Прізвище.
4. Написати програму, яка перевіряє на правильність введену адресу електронної пошти. Правильний формат: префікс@домен1.домен2[.домен3]. Приклади: 1) правильні – johndeer@gmail.com, ivan.franko@lnu.edu.ua; 2) неправильні – jackdoe@gmail, Іван2020@укр.нет.
5. В звіті навести копії екранів та написаний код.
6. Зробити висновки.