Лекція 14. Алгоритми та додаткові функції роботи з рядками.

План

1. Додаткові функції роботи з символами (string.h) та приклади їх застосування
2. Робота з прбілами та функція видалення
3. Різні функції розбиття на слова (токени) та їх застосування
4. Приклади функцій, що оцінюють, рахують та змінюють слова
5. Порядкове читання текстових файлів

Функції роботи з символами

#include <ctype.h>

Функція	Сенс
`isalpha`	перевіряє чи є символ літерою
`isalnum`	перевіряє чи є символ числом чи літерою
`isdigit`	перевіряє чи є символ числом
`islower`	перевіряє чи є символ малою літерою
`isupper`	перевіряє чи є символ великою літерою
`tolower`	перетворює літеру у нижній регістр
`toupper`	перетворює літеру у верхній регістр

*Ці функції працюють лише для англійських літер

Додаткові функції роботи з рядками

#include <string.h> або <cstring>)

Копіювання:

void * memcpy (void * destination, const void * source, size_t num);	Копіює блок у пам’ятті
void * memmove (void * destination, const void * source, size_t num);	Переносить блок у пам’ятті

Пошук:

void * memchr (const void *, int, size_t);	Повертає покажчик на символ у блоці памятті
char * strchr (const char *, int);	Повертає покажчик на перше входження символа у рядок
	Копіює n символів

http://www.cplusplus.com/reference/cstring/

Фунції пошуку символів та рядків

void * memchr (const void *, int, size_t);	Повертає покажчик на символ у блоці памятті
char * strchr (const char *, int);	Повертає покажчик на перше входження символа у рядок
char * strrchr (const char *, int);	Повертає покажчик на останнє входження символа у рядок
char * strpbrk (const char *, const char *);	Повертає покажчик на перше входження символа з набору символів (**str2**) у рядок
size_t strcspn (const char * str1, const char * str2);	Повертає індекс на першого входження символа з набору символів (**str2**) у рядок
char * strstr (const char *str1, const char *str2);	Повертає покажчик на перше входження підрядка (**str2**) у рядок (**str1**)

Фунція, що повертає номер позиції першого з набору символів у рядку

strcspn : повертає індекс першого входження

```
#include <string.h>
int main ()
{     char str[] = "fcba73";
   char keys[] = "1234567890";
   int i;
   i = strcspn (str,keys);
   printf ("The first number in str is at position
%d.\n",i+1);
return 0; }
```

Output:

```
The first number in str is at position 5
```

http://www.cplusplus.com/reference/cstring/strcspn/

Фунція, що повертає покажчик на позицію символа у рядку

strrchr – шукає перше входження символа c

```
#include <string.h>
int main () {
   char str[] = "This is a sample string";
   char *pch;
   pch= (str,'s');
   printf ("Last occurence of 's' found at %d \n",
);
   return 0; }
```

Output:

```
Last occurrence of 's' found at 18
```

Недолік – для пошуку індекса треба робити перетворення:
`pch-str+1`

http://www.cplusplus.com/reference/cstring/strrchr/

Особливості роботи з функціями, що повертають покажчик

На прикладі strpbrk – проходження та пошук вивід покажчиків на шукані символи

```
char str[] = "This is a sample string";
char key[] = "aeiou";
char *pch;
printf("str = %s \n", str);
pch = strpbrk(str, key);//перше входження
while (pch != ) {
   printf("%s\n", pch);
   pch = strpbrk(pch + 1, key); }
```

NULL – якщо символа не знайдено

pch = strpbrk(pch + 1, key); - забезпечує повторний пошук у піжлрядку

Особливості роботи з функціями, що повертають покажчик

На прикладі strpbrk – проходження та пошук вивід покажчиків на шукані символи

Output:

```
str = This is a sample string
is is a sample string
is a sample string
a sample string
ample string
e string
ing
```

Оригінальний приклад strpbrk

```
#include <string.h>
int main () {
   char str[] = "This is a sample string" ;
   char key[] = "aeiou" ;
   char *pch;
   printf ( "Vowels in '%s': " ,str);
   pch = strpbrk (str, key);
   while (pch != NULL) {
      printf ( "%c\n" , *pch);
      pch = strpbrk (pch+1, key); }
   printf ("\n");
 return 0; }
```

Output:

```
Vowels in 'This is a sample string': i i a a e i
```

http://www.cplusplus.com/reference/cstring/strpbrk/

Використання фунції, пошуку рядка у підрядку для його заміни

strstr – шукає перше рядка у підрядку

```
#include <string.h>
int main () {
   char str[] ="This is a simple string";
   char *pch;
   pch = strstr (str,"simple"); //знаходженя першої позиції
   strncpy (pch,"sample",6); //заміна
   puts (str);
   return 0; }
```

Output:

This is a sample string

http://www.cplusplus.com/reference/cstring/strstr/

Використання фунції, розбиття на токени

strtok – розбиває рядок на токени, за роздільниками

```
#include <string.h>
int main () {
   char str[] ="- This, a sample string.";
   char * pch;
   printf ("str = %s: \n",str);
   pch = strtok (str," ,.-"); //знаходженя першої позиції
   while (pch != NULL)
   {  printf ("%s\n",pch); //вивід токена
      pch = strtok (NULL, " ,.-"); //покажчик на новий токен
   return 0; }
```

http://www.cplusplus.com/reference/cstring/strtok/

Використання фунції, розбиття на токени

strtok – розбиває рядок на токени, за роздільниками

Output:

```
str = - This, a sample string.
This
a
sample
string
```

Переваги: працює для будь-яких символів, можна добирати роздільники

Недоліки: не дає повного розуміння роботи з рядками

http://www.cplusplus.com/reference/cstring/strtok/

Алгоритми роботи з рядками через покажчики.

Приклади роботи з функціями обробки рядків

Функции для работы со строками

- Копирование
- Конкатенация (склеивание)
- Сравнение
- Поиск
- Длина строки и заполнение символами
- Перевод строка-число и число-строка
- Форматированный ввод и вывод в буфер
- Работа с локалью

Довідка по функціям Сі

Приклади алгоритмів обробки рядків

1. **Романов Е. Л. Си/Си++. От дилетанта до профессионала**

   2.4. Стандартные программные контексты

   Видалення слова із заданим номером

   Пошук рядка у підрядку

   Підрахунок кількості слів

   Перевертання рядка

   Пошук коментарів

   5.2. Указатели и ссылки

   Пошук всіх заданих фрагментів у рядку з поверненням покажчика

   Сортування слів у рядку (вибором).

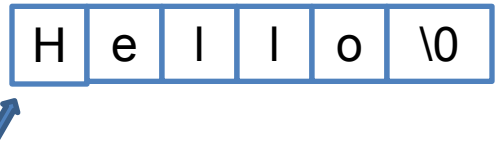
Приклад завдань string з розбиттям на слова та друком і видаленням слів:
http://learn.ztu.edu.ua/pluginfile.php/890/mod_resource/content/8/kr2_exampl1.pdf

Приклад роботи з символьними рядками через покажчики:

Цикл for виконується доти, доки *s=‘\0’, що є спеціальним символом кінця рядка.

```
int main()
{
  int num=0;
  char line[100]; //статичне виділення
пам’яті під масив із 100 символів
  gets(line); //ввід символів з клавіатури
  char *s; //змінна покажчик на char
  s=line; //s вказує на початок line
  for( ; *s; s++ ) //s проходить по line
     num++; //num рахує к-сть символіів
  printf("Kilkist simvoliv=%d", num);
  return 0;
}
```

line[]:



num = 8

На останньому кроці *s=‘\0’ , що є спеціальним символом кінця рядка.

Тому результат перевірки умови в for є true, що забезпечує вихід із циклу

Результат роботи програми:

```
hello
*s  = h, num= 1
*s  = e, num= 2
*s  = l, num= 3
*s  = l, num= 4
*s  = o, num= 5
Kilkist simvoliv=5
```

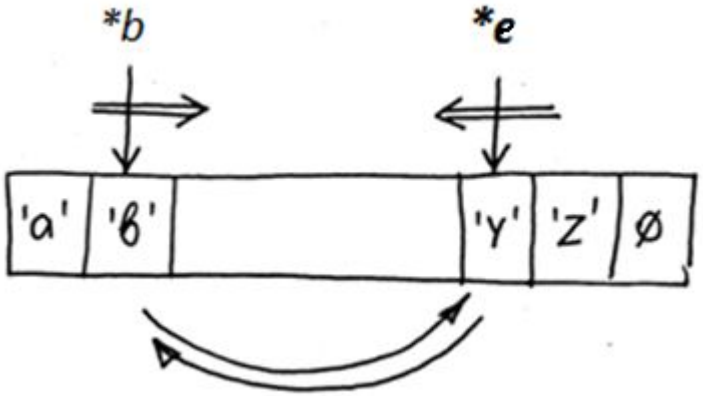
Підрахунок кількості слів

```
//CountWord рахує кількість слів поданих через один або більше пробілів
int CountWord(char c[]){
int nw = 0;
if (c[0] != ' ') nw = 1;    //Якщо рядок не починається з пробіла + 1 слово, якщо з пробіла рахується далі
for (int i = 1; c[i] != 0; i++)
   if (c[i] != ' ' && c[i - 1] == ' ')
nw++; //  Комбінація не пробіл, а перед ним пробіл - це початок слова
return nw;
}
```

Функція перевертання слова

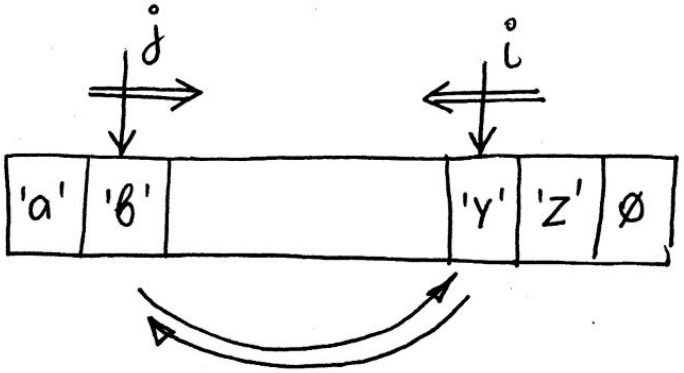
```
void polindrom (char *s) {
 char *begin = s, *end = s, tmp;
 for (; *end; end++); // Цикл
переміщення показчика в кінець рядка
 end--; //зрушення з \0
 for (; begin < end; begin++, end--){
  tmp = *begin;
  *begin = *end;
  *end = tmp;
}}
```

Показчики виконують роль індексів:



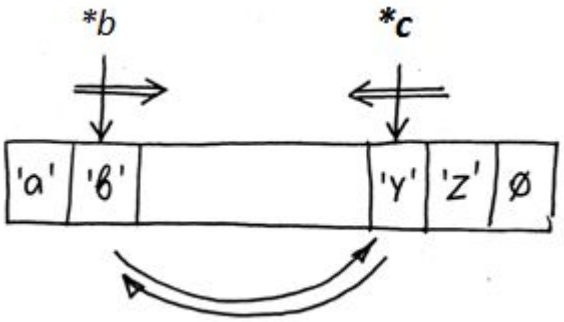
Перевертання слова

```
void swap(char c[]){
int i,j;
// Цикл пошука кінца рядка для i
for (i=0; c[i] !='\0'; i++);
// Цикл попарного обміну
for (j=0,i--; i>j; i--,j++)
      { char s; s=c[i]; c[i]=c[j]; c[j]=s; }}
```



Те ж саме, але з показчиками

```
void swap1(char *c){
char *b;
b = c; //показчик на початок рядка
// Цикл переміщення показчика в кінець
for (; *c; c++);  c--ж
// Цикл попарного обміну
for (; c > b ; c--,b++)
{ char s; s=*c; *c=*b; *b=s; }}
```



Показчики виконують роль індексів

Пошук у рядку заданого фрагменту

```
char *find (char *p, char *q){
for (; *p; p++){
    char *m = p, *q1 = q; //запам'ятовуємо покажчики
    // цикл до знаходження першої відмінності
        for (; *q1 && *m == *q1; q1++, m++);
        if ( *q1 == 0) return p;  //Якщо дійшли до кінця
підрядка, відмінностей не знайдено - повернення
покажчика
    } // інакше продовжити пошук
return NULL;}

int main()
 { char c[80]="find first abc and next abc and last abc",*q="abc", *s;
    for (s=find(c, q); s!=NULL; s=find(s+strlen(q),q)) puts(s);
```

Тема: Приклади алгоритмів обробки рядків

Приклад обробки математичного виразу

Завдання. Дано рядок, що зображає арифметичний вираз виду «<цифра> ± <цифра> ± ... ± <цифра>», де на місці знака операції «±» знаходиться символ «+» або «-» (наприклад, «4 + 7-2- 8 »). Вивести значення даного виразу (ціле число).

```
char *p, g[100],c;
int k=0;
   gets(g);
   p = g;//показчик приймає значення початку рядка
         //для проходу по константному масиву g
   for(;*p;p++)
   {
       if (*p == '+') {c='+';continue;} //запам'ятовування знаку +
       if (*p == '-') {c='-';continue;}//запам'ятовування знаку -
       if (*p >='0' && *p <='9')//дія, якщо цифра
       {
            if (c == '+') k+=*p -'0';
            else  if (c == '-')k-=*p -'0';
                else k=*p-'0';
       }
   }
   printf ("k=%d",k);
```

Тема: Приклади алгоритмів обробки рядків

Функції розбиття на слова та їх використання

Розбиття речення на слова. Потрібно врахувати, що програма не вміє просто «бачити слово», для неї необхідно формальна умова його виявлення. Таким може бути або кінець слова (не буква), або його початок (буква).

1. **Функція повертає покажчик на початок слова: char * strwordb (char* s)**
   a) **Отримує показчик на символу у рядку**
   b) **Перевіряє чи є він літерою (isalnum) , якшо так повертає показчик на нього**
   c) **Якщо ні шукає далі**
   d) **Якщо досягнуто кінець рядка повертається показчик вна нього**
2. **Функція повертає покажчик на кінець слова: char * strworde (char* s)**
   a) **Отримує показчик на символу у рядку**
   b) **Перевіряє чи є він (!isalnum) не літерою , якшо так повертає показчик на нього**
   c) **Якщо ні шукає далі**
   d) **Якщо досягнуто кінець рядка повертається показчик вна нього**
3. **Для використання цих функцій створєємо цикл, що ходить по рядку з виділенням слів: від початку слова шукаємо кінець, від кінця слова шукаємо початок наступного.**
4. **Ці функції можна використати для збереження слів у вільний масив (розділові знаки не збережуться)**

Тема: Приклади алгоритмів обробки рядків

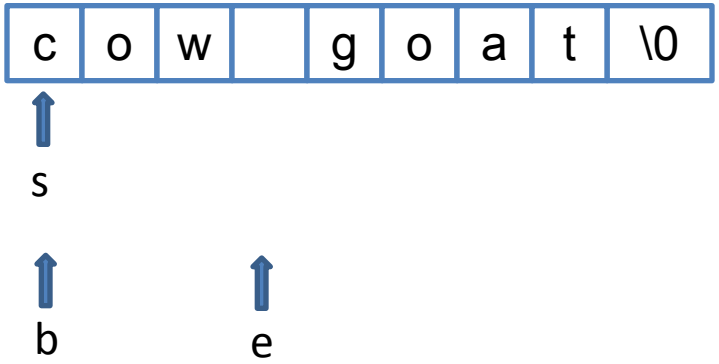
isalnum(int c) **перевірка, чи є символ літерою або цифрою;**

1 – повертає якщо символ є літерою або цифрою; 0 – у протилежному випадку

Функція повертає покажчик на початок слова

```
char *strwordb (char* s)
{  for( ; *s ; s++ ) //прохід по рядку
   if (isalnum(*s)) //якщо символ літера
        return s; //то повертаємо посилання
   return s;}
```

line[]:



Функція повертає покажчик на кінець слова

```
char *strworde (char* s)
{  for( ; *s ; s++ ) //прохід по рядку
    if (!isalnum(*s)) //якщо символ пробіл чи знак
        return s; //то повертаємо посилання
   return s;}
```

b – e = 4 3

max = 4 3

Функція повертає максимальну довжину слова

```
int MaxWord (char *s)
{ int max=0;
 char *b = strwordb(s),  char *e = NULL;
 for ( ;*b ;b = strwordb(e) ) {
    e=strworde(b);
    if( (e - b) > max) max = (e - b);}
return max;}
```

Знаходження найдовшого слова

```
int  main()
{
 char line[] = "cow goat";
 printf("Najdovshe slovo = %d\n", MaxWord (line));
 return 0;
}
```

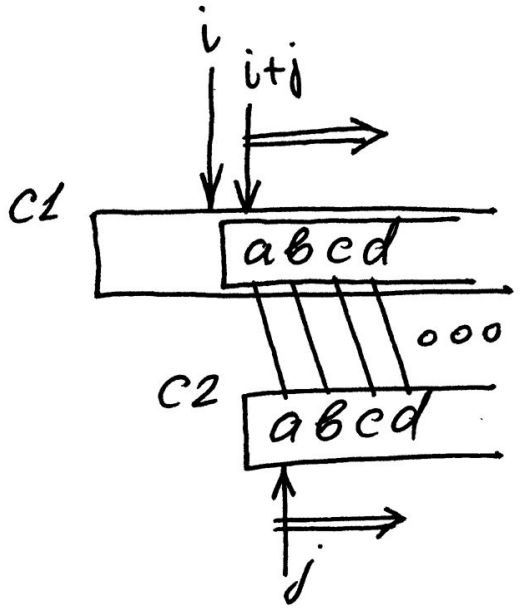
```
void Del(char *b,const char *e)
//Функція видаляє все, що знаходиться між покажчиком b та e
{ for(;*e;++b,++e)
    *b=*e;
  *b=*e;
}
// DelSpace - видаляє лишні пробіли
void DelSpace(char *s) //Завдання функції визначити проміжок 2 та більше пробілів та видалити рядок між ними переписавши весь хвіст
{
char *begspace = s,  *endspace = s;//покажчики на перший та останній пробіл
for (; *s; s++)
{
if (*s == ' ' && *(s + 1) == ' ' )//два підряд пробіли
if (begspace == endspace) { begspace = s; endspace = s + 1; }// та перші у серії - починаємо рахувати
else  endspace++;//не перші - росте хвіст - endspace++
else { //наступний не пробіл - запускаємо видалення
Del(begspace, endspace); begspace = endspace;
}
}
```

Тема: Приклади алгоритмів обробки рядків

Контекст:Вкладені цикли і принцип відносності

При аналізі процесу, що проходить у внутрішньому циклі, зовнішній можна вважати «умовно нерухомим».

```
// --- Пошук підрядка в рядку
int search (char c1 [], char c2 [])
{ Int I, j;
  for (i = 0; c1 [i] != '\0'; i ++) {
       for (j = 0; c2 [j] != '\0'; j ++)
            if (c1 [i + j] != c2 [j]) break;
       if (c2 [j] == '\0') return i + 1;  }
return -1;}
```



Використання у main

```
char c1[100], c2[100];
printf ("Введіть перший рядок: ");
gets(c1);
printf ("Введіть підрядок, що будемо шукати: ");
gets(c2);
if (search(c1, c2)==-1) printf ("Підрядка не знайдено\n");
else printf ("Номер початку підрядка = %d\n", search(c1,
c2))
```

Зафіксувавши зовнішній цикл, c1 [i + j] слід розуміти як j-ий символ щодо поточного, на якому знаходиться зовнішній цикл. Звідси ми бачимо паралельний рух з попарним порівнянням символів за двома рядками, але другий розглядається від початку, а перший рядок, від i-го символу.

Функції вводу/виводу у текстовий файл

Посимвольний в/в з файлу	Параметри і результат
int getc(FILE *fd)	Код символу або EOF
int putc(int ch, FILE *fd)	Код символа или EOF

Помилка відкриття файлу

Якщо виклик функції FOPEN пройшов невдало, то вона поверне значення NULL. Помилки під час роботи з файлами зустрічаються досить часто, тому кожен раз, коли ми Окриваем файл, необхідно перевіряти результат роботи

Функції прядкового вводу/виводу

```
FILE *input = NULL;
char c;
input = fopen("text.txt", "rt");
   if (input == NULL) {
   printf("Error opening file");
}
while (fgets(== 1) {
fprintf(stdout, "%c", c);
}

fclose(input);
```

Функції посимвольного вводу/виводу

```
FILE *f;
  char c;
  int i=0;
  char s[100];
  FILE *fd1=fopen("test.txt","r"); //Читання файлу
  FILE *fd2=fopen("res.txt","w"); // Створення файлу для запису
            if (fd1==NULL || fd2==NULL) return -1;
            while ((c=getc(fd1)) != EOF)//посимвольне читання з
файлу
                { printf("%c",c);
                if ((isalpha(c)) && isupper(c)) s[i++]=c; }// Запис
великих латинських літер у масив
            for(i--;i>=0;i--) putc(tolower(s[i]),fd2); // Запис
маленьких латинських літер у файл у зворотньому порядку
```

В цьому прикладі показується посимвольне копіювання вхідного файлу у вихідний.