Лабораторна робота №15
Елементарні операції з текстовими файлами і рядками

Мета роботи: отримання практичних навичок роботи з текстовими файлами і маніпуляції рядками.

Зміст:

Короткі теоретичні відомості

Рядки в С

Для роботи з рядками в *С* немає спеціального типу даних як в деяких інших мовах програмування.

Рядок у *С* – це послідовність байт (масив символів, величин типу ***char***), що завершується в кінці спеціальною ознакою – байтом '\0'. Тобто в мові *С* використовується так звана концепція «*zero termination string*». Це означає, що для позначення закінчення кожного рядка (масиву типу ***char***) використовується нуль-символ ('\0').

Наприклад:

```
char err_str1[6] = "Error";
char err_str2[6] = {'E', 'r', 'r', 'o', 'r', '\0'};
```

У цьому прикладі оголошення рядків будуть ідентичні:

```
err_str1[0] = err_str2[0] = 'E';
err_str1[1] = err_str2[1] = 'r';
err_str1[2] = err_str2[2] = 'r';
err_str1[3] = err_str2[3] = 'o';
err_str1[4] = err_str2[4] = 'r';
err_str1[5] = err_str2[5] = '\0';
```

Для друку цілого рядка за допомогою функції *printf* використовують ключ ***%s***. Ніякий інший ключ не дозволить надрукувати рядок цілком (тобто до тих пір поки не зустрінеться символ закінчення рядка '\0'). Рядок так само може розглядатися як масив символів і друкуватися елемент за елементом за допомогою ключа ***%c***.

Наприклад:

```
char i = 0;
char err_str1[6] = "Error";

printf("%s\n", err_str1);
while(err_str1[i]) printf("%c", err_str1[i++]);
```

Часто при оголошенні рядка з ініціалізацією можна не вказувати розмір рядка. При наявності ініціалізації розмір рядка буде підрахований автоматично – він дорівнює кількості символів у константі +1 (для зберігання нуль символу).

Наприклад:

```
char err_str1[] = "Error";
```

У цьому прикладі розмірність масиву *err_str1* буде дорівнювати 5 + 1 = 6.

Оскільки всі рядки в мові С закінчуються нулем, який має значення "хибно", то умова в операторі *while(*str)* буде істинною до тих пір, поки програма не досягне кінця рядка. Така умова часто використовується коли треба здійснити наступну обробку: «виконувати до кінця рядка».

Таблиця ASCII

Вся інформація, включаючи і символи, зберігається в комп'ютері у вигляді чисел.

Існує декілька способів кодування символів за допомогою цифр. Серед найбільш поширених з них є **ASCII** таблиці та **UNICODE**.

ASCII (англ. *American Standard Code for Information Interchange*) – американська стандартна таблиця кодування для друкованих символів і деяких спеціальних кодів. У цій таблиці кожному символу відповідає число від 0 до 255, яке називається **ASCII**-кодом символу.

Цей стандарт базується на англійському алфавіті. Коди **ASCII** представляють текст в комп'ютерах, комунікаційному обладнанні та інших пристроях, які працюють з текстом. **ASCII** був створений в 1963 році, але вперше опублікований як стандарт в 1967 році. Останні зміни були внесені в 1986 році.

Перші 32 символи (0-31) цієї таблиці є недруковані.

Таблиця **ASCII** має такий вигляд:

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Робота з файлами

Маніпуляції з файлами реалізуються за допомогою функцій стандартної бібліотеки. З кожним файлом асоціюється *потік*, і обмін відбувається через потоки. Технічно обслуговування потоку здійснюється з використанням службової структури типу `FILE`. Функції стандартної бібліотеки для маніпуляції з файлами приймають вказівник на цю структуру. Структура `FILE` описана у заголовочному файлі `<stdio.h>`. До початку роботи файл повинен бути відритий в належному режимі. По завершенні роботи програми файл закривається операційною системою, проте хорошим тоном вважається явне закриття файлів після того, як робота з ними закінчена. Слід також проявляти обережність при роботі із вказівниками.

fopen— функція відкриття файлів

```
FILE *fopen(
   const char *filename,
   const char *mode
);
```

Параметри

filename — ім'я файлу (повний чи відносний шлях), що підлягає відкриттю.

mode — режим доступу до файлу.

Параметри що повертаються

Функція повертає вказівник на відкритий файл. Якщо під час відкриття файлу відбулася помилка, повертається **NULL**. Завжди перевіряйте параметр що повертається на валідність.

Значення **NULL** повертається , наприклад, в ситуації коли файл не існує, або шлях до нього вказано некоректною

Режим доступу

Режим доступу кодується наступними комбінаціями символів:

"r" – відкриває файл на читання. Якщо файл не існує або не може бути відкритий, функція повертає **NULL**.

"w" – відкриває порожній файл на запис. Якщо файл існує, його вміст знищується.

"a" – файл відкриває на дописування в кінець файлу.

"r+" – відкриває існуючий файл на читання і запис.

"w+" – відкриває порожній файл на читання і запис.

fclose — функція закриття файлів

```
int fclose(
   FILE *stream
);
```

fclose повертає 0 якщо потік успішно закрито, і **EOF** в іншому випадку.

fgetc — функція читання символу

Читає символ з потоку.

```
int fgetc(
   FILE *stream
);
```

fgetc повертає прочитаний символ як число типу **int** або повертає **EOF**, якщо досягнуто кінець файлу. Значення що було прочитане має тип **int,** тому його слід явно приводити до типу **char**.

Робоче завдання

Скласти програму, в якій:

1. Зчитується символьний рядок даних з файлу *text.txt* (текстовий файл потрібно завантажити с сайту чи скопіювати текст після таблиці із варіантами та зберегти в каталозі проекту).
2. Проводиться обробка символьних даних згідно варіанту.
3. Отримані дані записуються в новий файл *new_text.txt*.

ПРИМІТКА. Словом вважати послідовність символів, обмежену пробілами, комами, крапками, лапками, або їх комбінацією.

Номер	Завдання
1	В символьному рядку замінити всі символи лапок (") на символи дужок що відкриваються та закриваються (лапки з непарними номерами замінити на дужки що відкриваються, а з парними номерами - на дужки що закриваються).
2	В символьному рядку видалити найдовше слово.
3	Для кожного символу, який не є буквою і пробілом, вказати скільки разів він зустрічається в тексті. Результат записати в новий файл. ПРИМІТКА. Результат записати у вигляді: «символ - скільки-раз-зустрічається кінець-рядка».
4	Читати з клавіатури два числа M і N. Видалити з символьного рядка N символів починаючи з M-тій позиції. ПРИМІТКА. Числа M і N повинні бути менше 100.
5	Підрахувати скільки разів в рядку зустрічається слово «structure», а також знайти починаючи з якого символу це слово зустрічається в тексті. Результат записати в новий файл. ПРИМІТКА. Результат записати у вигляді: «поточний-номер-слова-structure: номер-символа-початку-слова-structure кінець-рядка».
6	Визначити відсоток голосних і приголосних букв від загальної кількості букв в символьному рядку. ПРИМІТКА. Результат записати у вигляді: «голосні / приголосні: відсоток кінець-рядка».
7	У рядку замінити всі безперервні послідовності пробілів одним пропуском.
8	Визначити процентне співвідношення буквених і небуквених символів в рядку. ПРИМІТКА. Результат записати у вигляді: «букви / не букви: відсоток-в-речовинному-форматі кінець-рядка».
9	Замінити всі непарні слова в рядку пропусками.

10	У рядку замінити всі літери, написані верхнім регістром на ті ж букви, але в нижньому регістрі.
11	Підрахувати кількість однобуквених, двобуквених і т.д. слів в рядку. Результат записати в новий файл. ПРИМІТКА. Результат записати у вигляді: «кількість-букв-в-слові - скільки-раз-зустрічається кінець-рядка».
12	Виписати в новий файл всі парні слова.
13	Підрахувати середню довжину слів з рядку.
14	Знайти найдовше слово в тексті символьного рядку і замінити в ньому всі символи на #. ПРИМІТКА. Якщо в тексті зустрічається декілька слів з найбільшою довжиною, то зробити заміну в кожному з них.
15	Розташувати символи рядка в зворотному порядку.
16	Видалити з символьного рядка все однобуквені слова.
17	Знайти найкоротший слово в тексті символьного рядка і замінити в ньому всі символи на символ $. ПРИМІТКА. Якщо в тексті зустрічається декілька слів з найменшою довжиною, то зробити заміну в кожному з них.
18	В символьному рядку змінити регістр всіх букв.
19	Підрахувати скільки разів в рядку зустрічаються символи «a», «b» та «c». Результат записати в новий файл. ПРИМІТКА. Результат записати у вигляді: «символ - скільки-раз-зустрічається кінець-рядка».
20	В символьному рядку замінити всі символи відкриваються дужок на символ *, а символи закриваються дужок на #.
21	Читати з клавіатури два числа M та N (M, N <20). Видалити з ліченої рядка N слів починаючи з M-того слова.

text.txt

A "structure declaration" names a type and specifies a sequence of variable values (called "members" or "fields" of the structure) that Can have different types. An optional identifier, called a "tag," (GIVEs) the name of the (structure type and can be Used in subsequent).

Контрольні питання

1. Чи існує якийсь специфічний тип даних для зберігання рядків у мові *C*?
2. Який мінімальний розмір масиву повинен бути для зберігання рядка *'abc'*? Чому?
3. Чи припустимо явно не вказувати розмір масив, в якому буде зберігатися рядок? Якщо допустимо, то в яких випадках?
4. Що зберігається в таблиці **ASCII**?
5. Які ще таблиці кодування вам відомі?
6. Які функції роботи з файлами вам відомі?