

Двійковий файловий ввід-вивід



Лекція №16

Дисципліна «Програмування»



Функція fwrite()

Прототип функції:

```
size_t fwrite(const void * restrict ptr, size_t size,  
              size_t nmemb, FILE * restrict fp);
```

Функція `fwrite()` записує двійкові дані до файлу.

Тип `size_t` визначений в термінах стандартних типів C. Це тип, що повертається операцією `sizeof`.

Зазвичай їм є тип `unsigned int`, але реалізації можуть обирати інший тип.

Вказівник `ptr` – це адреса порції даних, що призначена для запису.

Аргумент `size` являє собою розмір в байтах порції даних, що підлягають запису, а `nmemb` – кількість таких порцій.



Функція fwrite()

Щоб зберегти об'єкт даних (такий як масив) розміром 256 байтів, можна записати:

```
char buffer[256];  
fwrite(buffer, 256, 1, fp);
```

Цей виклик `fwrite()` записує одну порцію даних розміром 256 байтів з буфера `buffer` до файлу.

Щоб зберегти масив з 10 елементів типу `double`, можна написати:

```
double mas[10];  
fwrite(mas, sizeof(double), 10, fp);
```

Цей виклик `fwrite()` записує дані з масиву `mas` до файлу 10 порціями даних, кожна з яких має розмір `double`.



Внутрішнє представлення

```
int num = 12345;
```



Зберігає 12345 як двійкове число в num



00110000	00111001
----------	----------

Внутрішнє представлення

```
fprintf(fp, "%d", num);
```



Записує до файлу двійкові коди символів '1', '2', '3', '4', '5'



00110001	00110010	00110011	00110100	00110101
----------	----------	----------	----------	----------

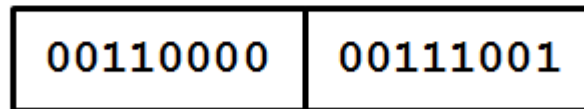


Внутрішнє представлення

```
fwrite(&num, sizeof(int), 1, fp);
```



Записує до файлу двійковий код значення 12345



(передбачається, що тип `int` має розмір 16 бітів)



Функція fread()

Прототип функції:

```
size_t fread(void * restrict ptr, size_t size,  
             size_t nmemb, FILE * restrict fp);
```

Функція `fread()` приймає такий самий набір аргументів, як і `fwrite()`.

`ptr` являє собою адресу області пам'яті, куди поміщаються дані, які прочитані з файлу;

`fp` ідентифікує файл, з якого відбувається читання.

Цю функцію слід використовувати для читання даних, які були записані до файлу за допомогою `fwrite()`.



Функція fread()

Для того щоб прочитати масив з 10 елементів типу **double**, який був збережений у файлі, слід застосувати наступний код:

```
double mas[10];  
fread(mas, sizeof(double), 10, fp);
```

Цей виклик копіює 10 значень розміру **double** до масиву **mas**.

Функція **fread()** повертає кількість успішно прочитаних елементів.

Зазвичай вона дорівнює **nmemb**, однак може бути й меншою, якщо виникла помилка запису або був досягнутий кінець файлу.



Функції `feof()` і `ferror()`

Коли стандартні функції вводу повертають **EOF**, це зазвичай означає, що **досягнутий кінець файлу**.

Тим не менш, повернення **EOF** може також вказувати на **виникнення помилки читання**.

Функція `feof()` повертає ненульове значення, якщо при останньому виклику функції вводу був **виявлений кінець файлу**, і нуль – в іншому випадку.

Функція `ferror()` повертає ненульове значення, якщо виникла **помилка читання або запису**, і нуль – в іншому випадку.



Застосування fread() і fwrite()

```
#include <stdio.h>
#include <windows.h>
#include <stdlib.h>
#include <string.h>

#define BUFSIZE 4096
#define SLEN 81

void append(FILE *source, FILE *dest);
char *s_gets(char *st, int n);

int main(void)
{
    FILE *fs;           // fs для вхідного файлу
    FILE *fa;           // fa для файлу призначення
    int files = 0;      // кількість файлів, що додаються
    char file_app[SLEN]; // ім'я файлу призначення
    char file_src[SLEN]; // ім'я вхідного файлу
    int ch;
```



Застосування fread() і fwrite()

```
SetConsoleOutputCP(1251);

puts("Введіть і'мя файлу призначення:");
s_gets(file_app, SLEN);
if((fa = fopen(file_app, "a+")) == NULL)
{
    fprintf(stderr, "Не вдається відкрити %s\n", file_app);
    exit(EXIT_FAILURE);
}
if(setvbuf(fa, NULL, _IOFBF, BUFSIZE) != 0)
{
    fputs("Не вдається створити вихідний буфер\n", stderr);
    exit(EXIT_FAILURE);
}
puts("\nВведіть ім'я першого файлу "
      "(або пустий рядок для завершення):");
while(s_gets(file_src, SLEN) && file_src[0] != '\0')
{
```



Застосування fread() і fwrite()

```
if(strcmp(file_src, file_app) == 0)
    fputs("Додати файл у кінець самого себе неможливо",
          stderr);
else
{
    if((fs = fopen(file_src, "r")) == NULL)
        fprintf(stderr, "Не вдається відкрити %s\n",
                 file_src);
    else
    {
        if(setvbuf(fs, NULL, _IOFBF, BUFSIZE) != 0)
        {
            fputs("Не вдається створити вхідний буфер\n",
                  stderr);
            continue;
        }
        append(fs, fa);
    }
}
```



Застосування fread() і fwrite()

```
if(ferror(fs) != 0)
    fprintf(stderr, "Помилка при читанні файлу %s.\n",
            file_src);
if(ferror(fa) != 0)
    fprintf(stderr, "Помилка при запису файлу %s.\n",
            file_app);
fclose(fs);
files++;
printf("Вміст файлу %s додано.\n", file_src);
puts("Введіть ім'я наступного файлу "
      "(або порожній рядок для завершення):");
}
}
}
printf("Додавання завершено.\n");
printf("Кількість доданих файлів: %d.\n\n", files);
rewind(fa);
printf("Вміст %s:\n", file_app);
```



Застосування fread() і fwrite()

```
puts("=====");  
while ((ch = getc(fa)) != EOF)  
    putchar(ch);  
puts("\n=====");  
puts("Задачу завершено.");  
puts("=====\n");  
fclose(fa);  
return 0;  
}  
  
void append(FILE *source, FILE *dest)  
{  
    size_t bytes;  
    static char temp[BUFSIZE]; // виділити пам'ять один раз  
  
    fputs("\n", dest);  
    while ((bytes = fread(temp, sizeof(char), BUFSIZE, source)) > 0)  
        fwrite(temp, sizeof(char), bytes, dest);  
}
```

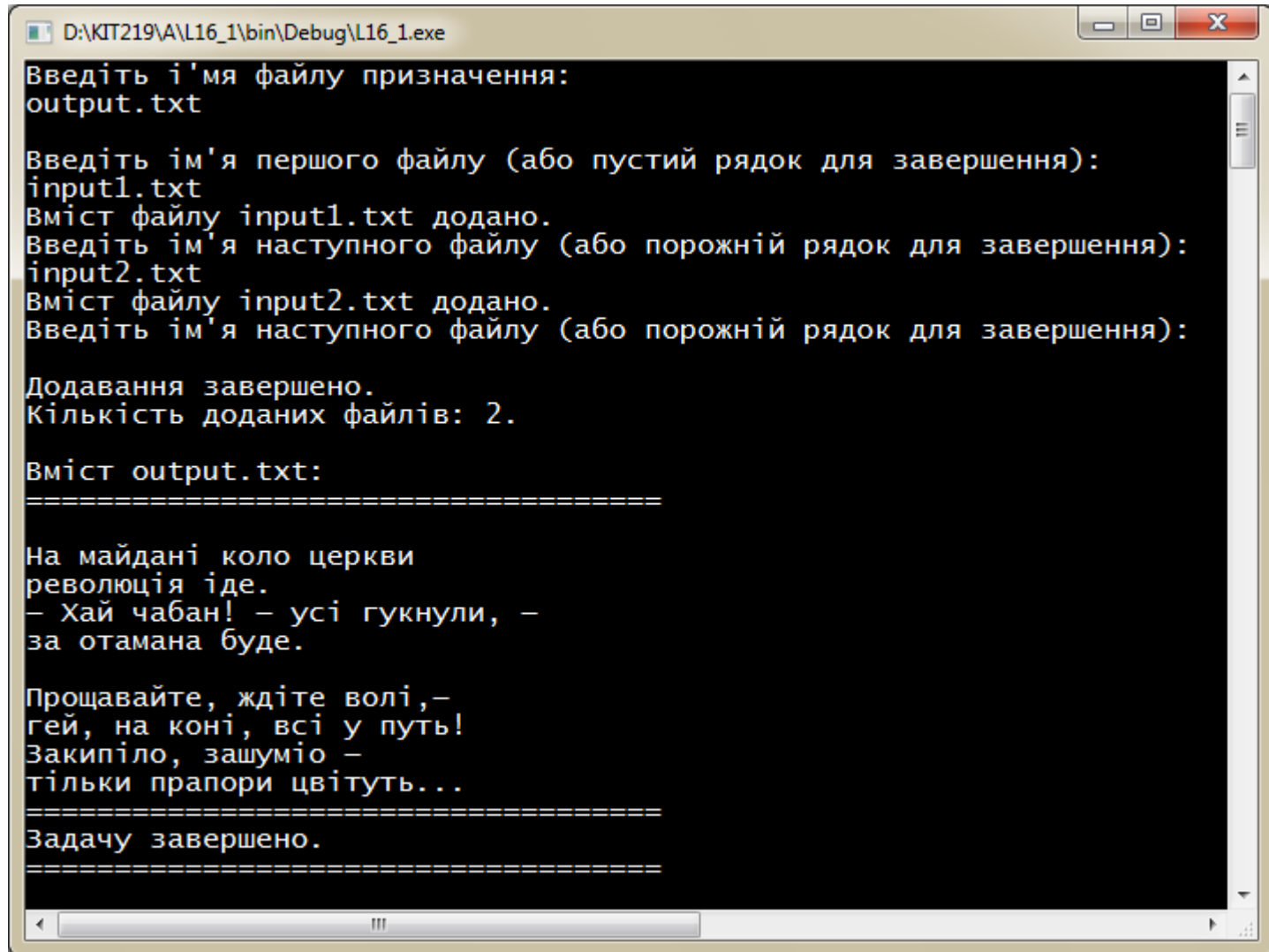


Застосування fread() і fwrite()

```
char *s_gets(char *st, int n)
{
    char *ret_val;
    char *find;

    ret_val = fgets(st, n, stdin);
    if(ret_val)
    {
        find = strchr(st, '\n'); // пошук символу нового рядка
        if(find)                 // якщо адреса не є NULL,
            *find = '\0';        // помістити туди нульовий символ
        else
            while(getchar() != '\n')
                continue;
    }
    return ret_val;
}
```

Застосування fread() і fwrite()



D:\KIT219\A\L16_1\bin\Debug\L16_1.exe

```
Введіть імя файлу призначення:  
output.txt  
  
Введіть ім'я першого файлу (або пустий рядок для завершення):  
input1.txt  
Вміст файлу input1.txt додано.  
Введіть ім'я наступного файлу (або порожній рядок для завершення):  
input2.txt  
Вміст файлу input2.txt додано.  
Введіть ім'я наступного файлу (або порожній рядок для завершення):  
  
Додавання завершено.  
Кількість доданих файлів: 2.  
  
Вміст output.txt:  
=====
```

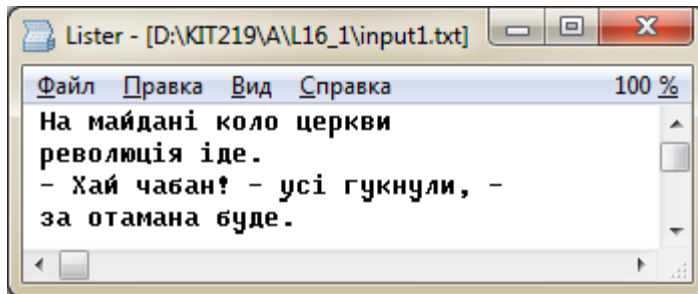
На майдані коло церкви
революція іде.
– Хай чабан! – усі гукнули, –
за отамана буде.

Прощавайте, ждіте волі,–
гей, на коні, всі у путь!
Закипіло, зашуміо –
тільки прапори цвітуть...

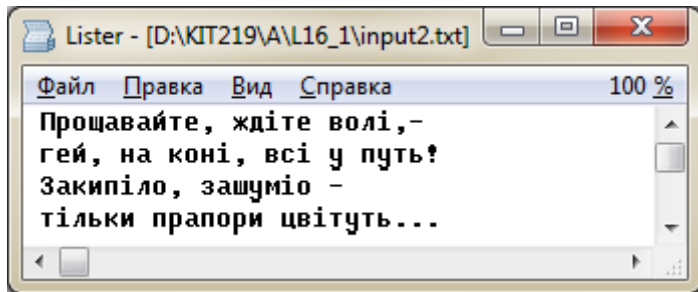
```
=====
```

Задачу завершено.
=====

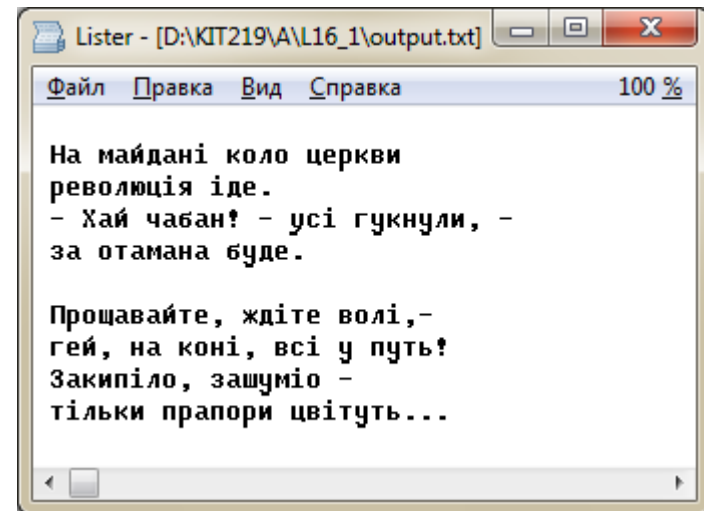
Застосування fread() і fwrite()



```
Листер - [D:\KIT219\A\L16_1\input1.txt]
Файл  Правка  Вид  Справка  100 %
На майдані коло церкви
революція іде.
- Хай чабан! - усі гукнули, -
за отамана буде.
```



```
Листер - [D:\KIT219\A\L16_1\input2.txt]
Файл  Правка  Вид  Справка  100 %
Прощавайте, ждіте волі,-
гей, на коні, всі у путь!
Закипіло, зашуміо -
тільки прапори цвітуть...
```



```
Листер - [D:\KIT219\A\L16_1\output.txt]
Файл  Правка  Вид  Справка  100 %

На майдані коло церкви
революція іде.
- Хай чабан! - усі гукнули, -
за отамана буде.

Прощавайте, ждіте волі,-
гей, на коні, всі у путь!
Закипіло, зашуміо -
тільки прапори цвітуть...
```



Довільний доступ до файлу

```
#include <stdio.h>
#include <windows.h>
#include <stdlib.h>
#define ARSIZE 1000

int main(void)
{
    double numbers[ARSIZE];
    double value;
    const char *file = "numbers.dat";
    int i;
    long pos;
    FILE *iofile;

    SetConsoleOutputCP(1251);

    // створення набору значень типу double
    for(i = 0; i < ARSIZE; i++)
        numbers[i] = 100.0 * i + 1.0 / (i + 1);
    // спроба відкрити файл
```



Довільний доступ до файлу

```
if((iofile = fopen(file, "wb")) == NULL)
{
    fprintf(stderr, "Не вдається відкрити файл %s для виводу\n",
               file);
    exit(EXIT_FAILURE);
}
// запис у файл масиву в двійковому форматі
fwrite(numbers, sizeof(double), ARSIZE, iofile);
fclose(iofile);

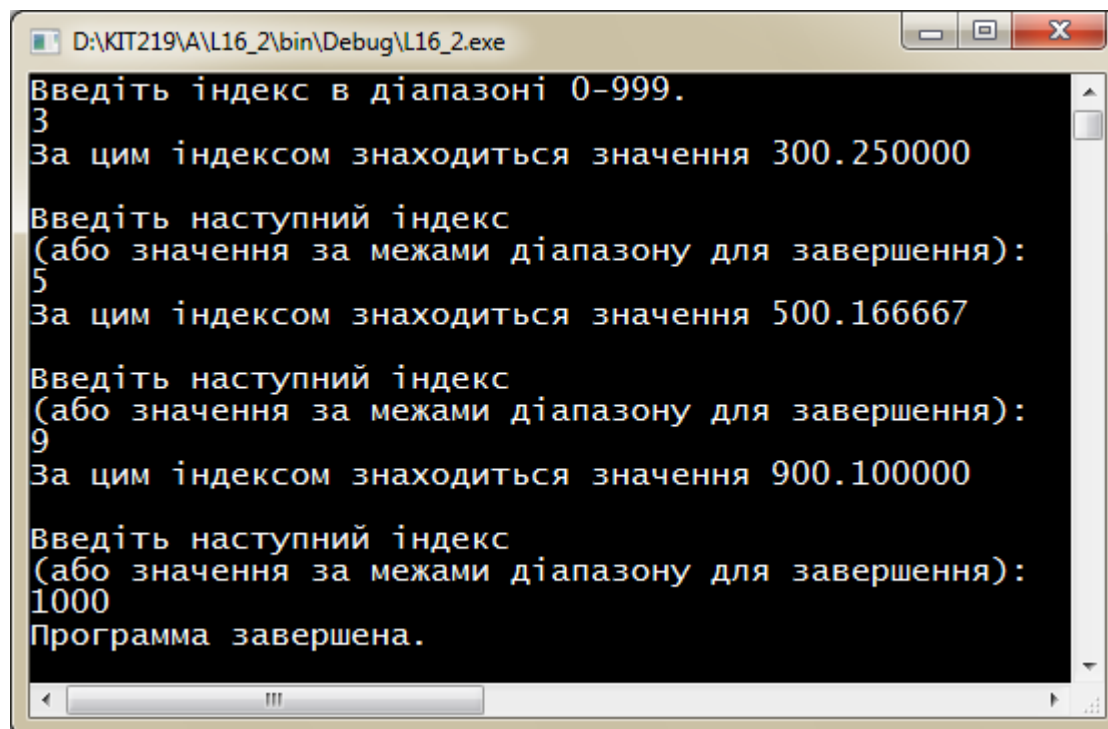
if((iofile = fopen(file, "rb")) == NULL)
{
    fprintf(stderr, "Не вдається відкрити файл %s для"
                  " довільного доступу.\n", file);
    exit(EXIT_FAILURE);
}
```



Довільний доступ до файлу

```
// читання обраних елементів з файлу
printf("Введіть індекс в діапазоні 0-%d.\n", ARSIZE - 1);
while(scanf("%d", &i) == 1 && i >= 0 && i < ARSIZE)
{
    pos = (long)i * sizeof(double); // обчислення зміщення
    fseek(iofile, pos, SEEK_SET);    // перехід в необхідну
                                     // позицію
    fread(&value, sizeof(double), 1, iofile);
    printf("За цим індексом знаходиться значення %f\n",
           value);
    printf("\nВведіть наступний індекс\n(або значення"
           "\n за межами діапазону для завершення):\n");
}
// завершення
fclose(iofile);
puts("Програма завершена.");
return 0;
}
```

Довільний доступ до файлу



```
D:\KIT219\A\L16_2\bin\Debug\L16_2.exe
Введіть індекс в діапазоні 0-999.
3
За цим індексом знаходиться значення 300.250000
Введіть наступний індекс
(або значення за межами діапазону для завершення):
5
За цим індексом знаходиться значення 500.166667
Введіть наступний індекс
(або значення за межами діапазону для завершення):
9
За цим індексом знаходиться значення 900.100000
Введіть наступний індекс
(або значення за межами діапазону для завершення):
1000
Програма завершена.
```