

## File position pointer

- File position pointer
  - Indica la posizione da cui leggere/scrivere I prossimi byte
  - Non un puntatore, ma un valore intero (specifica locazioni di bytes)
  - E` anche chiamato byte offset
- `rewind( cFPtr )`
  - Riposiziona il file position pointer all'inizio del file (byte 0).

1

## Struttura in memoria

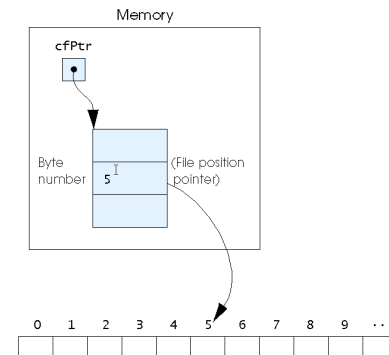


Fig. 11.14 The file position pointer indicating an offset of 5 bytes from the beginning of the file.

2

## Files ad accesso sequenziale:

- Se vengono modificati i dati possono essere distrutti.
- I campi possono variare in dimensione
  - (non sono la rappresentazione interna)
  - 1, 34, -890 sono tutti interi `int`, ma possono avere differenti rappresentazioni.

300 white 0.00 400 Jones 32.87 (vecchio valore)

Se vogliamo cambiarlo:

300 Worthington 0.00

300 White 0.00 400 Jones 32.87

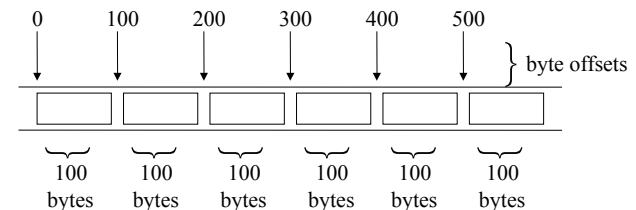
300 Worthington 0.00ones 32.87

I dati vengono sovrascritti

3

## Files ad accesso diretto

- Files ad accesso diretto:
  - Accedono i record individualmente e direttamente
  - I dati possono essere inseriti senza distruggere altri dati
  - I dati precedenti possono essere modificati o cancellati.
- Realizzati usando record di lunghezza fissa
  - ( I files sequenziali non hanno lunghezza fissa



4

## Creare un file ad accesso diretto (binario)

### ■ Dati:

- Non sono formattati (sono solo bytes)
  - Tutti i dati dello stesso tipo usano la stessa memoria (`int`, per esempio)
  - Tutti i record dello stesso tipo hanno la stessa lunghezza
  - I dati sono “illeggibili” per l’utente

5

```
1 /* Fig. 11.11: fig11_11.c
2   Creating a randomly accessed file sequentially */
3 #include <stdio.h>
4
5 /* clientData structure definition */
6 struct clientData {
7     int acctNum; /* account number */
8     char lastName[ 15 ]; /* account last name */
9     char firstName[ 10 ]; /* account first name */
10    double balance; /* account balance */
11 }; /* end structure clientData */
12
13 int main()
14 {
15     int i; /* counter */
16
17     /* create clientData with no information */
18     struct clientData blankClient = { 0, "", "", 0.0 };
19
20     FILE *cfPtr; /* credit.dat file pointer */
21
22     /* fopen opens the file; exits if file cannot be opened */
23     if ( ( cfPtr = fopen( "credit.dat", "wb" ) ) == NULL ) {
24         printf( "File could not be opened.\n" );
25     } /* end if */
```

7

## Posizionamento del puntatore di posizione all'interno del file

### ■ fseek

- Setta il file position pointer ad una certa posizione
- `fseek( pointer, offset, symbolic_constant );`
  - `pointer` – puntatore al file
  - `offset` – file position pointer (0 e` la prima locazione del file)
  - `symbolic_constant` – specifica da dove partire
  - `SEEK_SET` – dall'inizio del file
  - `SEEK_CUR` – dalla corrente locazione
  - `SEEK_END` – alla fine del file

6

```
26 else {
27
28     /* output 100 blank records to file */
29     for ( i = 1; i <= 100; i++ ) {
30         fwrite( &blankClient, sizeof( struct clientData ), 1, cfPtr );
31     } /* end for */
32
33     fclose ( cfPtr ); /* fclose closes the file */
34 } /* end else */
35
36 return 0; /* indicates successful termination */
37
38 } /* end main */
```

8

```

1 /* Fig. 11.12: fig11_12.c
2   Writing to a random access file */
3 #include <stdio.h>
4
5 /* clientData structure definition */
6 struct clientData {
7     int acctNum; /* account number */
8     char lastName[ 15 ]; /* account last name */
9     char firstName[ 10 ]; /* account first name */
10    double balance; /* account balance */
11 }; /* end structure clientData */
12
13 int main()
14 {
15     FILE *cfPtr; /* credit.dat file pointer */
16
17     /* create clientData with no information */
18     struct clientData client = { 0, "", "", 0.0 };
19
20     /* fopen opens the file; exits if file cannot be opened */
21     if ( ( cfPtr = fopen( "credit.dat", "rb" ) ) == NULL ) {
22         printf( "File could not be opened.\n" );
23     } /* end if */
24     else {
25

```

```

26     /* require user to specify account number */
27     printf( "Enter account number"
28           " ( 1 to 100, 0 to end input )\n? " );
29     scanf( "%d", &client.acctNum );
30
31     /* user enters information, which is copied into file */
32     while ( client.acctNum != 0 ) {
33
34         /* user enters last name, first name and balance */
35         printf( "Enter lastname, firstname, balance\n? " );
36
37         /* set record lastName, firstName and balance value */
38         fscanf( stdin, "%s%s%lf", client.lastName,
39               client.firstName, &client.balance );
40
41         /* seek position in file of user-specified record */
42         fseek( cfPtr, ( client.acctNum - 1 ) *
43               sizeof( struct clientData ), SEEK_SET );
44
45         /* write user-specified information in file */
46         fwrite( &client, sizeof( struct clientData ), 1, cfPtr );
47
48         /* enable user to specify another account number */
49         printf( "Enter account number\n? " );
50         scanf( "%d", &client.acctNum );

```

9

10

```

51     } /* end while */
52
53     fclose( cfPtr ); /* fclose closes the file */
54 } /* end else */
55
56 return 0; /* indicates successful termination */
57
58 } /* end main */

```

```

1 /* Fig. 11.15: fig11_15.c
2   Reading a random access file sequentially */
3 #include <stdio.h>
4
5 /* clientData structure definition */
6 struct clientData {
7     int acctNum; /* account number */
8     char lastName[ 15 ]; /* account last name */
9     char firstName[ 10 ]; /* account first name */
10    double balance; /* account balance */
11 }; /* end structure clientData */
12
13 int main()
14 {
15     FILE *cfPtr; /* credit.dat file pointer */
16
17     /* create clientData with no information */
18     struct clientData client = { 0, "", "", 0.0 };
19
20     /* fopen opens the file; exits if file cannot be opened */
21     if ( ( cfPtr = fopen( "credit.dat", "rb" ) ) == NULL ) {
22         printf( "File could not be opened.\n" );
23     } /* end if */

```

## Program Output

```

Enter account number ( 1 to 100, 0 to end input )
? 37
Enter lastname, firstname, balance
? Barker Doug 0.00
Enter account number
? 29
Enter lastname, firstname, balance
? Brown Nancy -24.54
Enter account number
? 96
Enter lastname, firstname, balance
? Stone Sam 34.98
Enter account number
? 88
Enter lastname, firstname, balance
? Smith Dave 258.34
Enter account number
? 33
Enter lastname, firstname, balance
? Dunn Stacey 314.33
Enter account number
? 0

```

11

12

```

24 else {
25     printf( "%-6s%-16s%-11s%10s\n", "Acct", "Last Name",
26           "First Name", "Balance" );
27
28     /* read all records from file (until eof) */
29     while ( !feof( cfPtr ) ) {
30         fread( &client, sizeof( struct clientData ), 1, cfPtr );
31
32         /* display record */
33         if ( client.acctNum != 0 ) {
34             printf( "%-6d%-16s%-11s%10.2f\n",
35                   client.acctNum, client.lastName,
36                   client.firstName, client.balance );
37         } /* end if */
38
39     } /* end while */
40
41     fclose( cfPtr ); /* fclose closes the file */
42 } /* end else */
43
44 return 0; /* indicates successful termination */
45
46 } /* end main */

```

Acct	Last Name	First Name	Balance
29	Brown	Nancy	-24.54
33	Dunn	Stacey	314.33
37	Barker	Doug	0.00
88	Smith	Dave	258.34
96	Stone	Sam	34.98

## Program Output