CME 112- Programming Languages II

Week 10
File Operations
Sequential Access Files

Assist. Prof. Dr. Caner Özcan

Ne kadar okursan oku, bilgine, yakışır şekilde davranmıyorsan cahilsin demektir.
- Sadi Sirazi

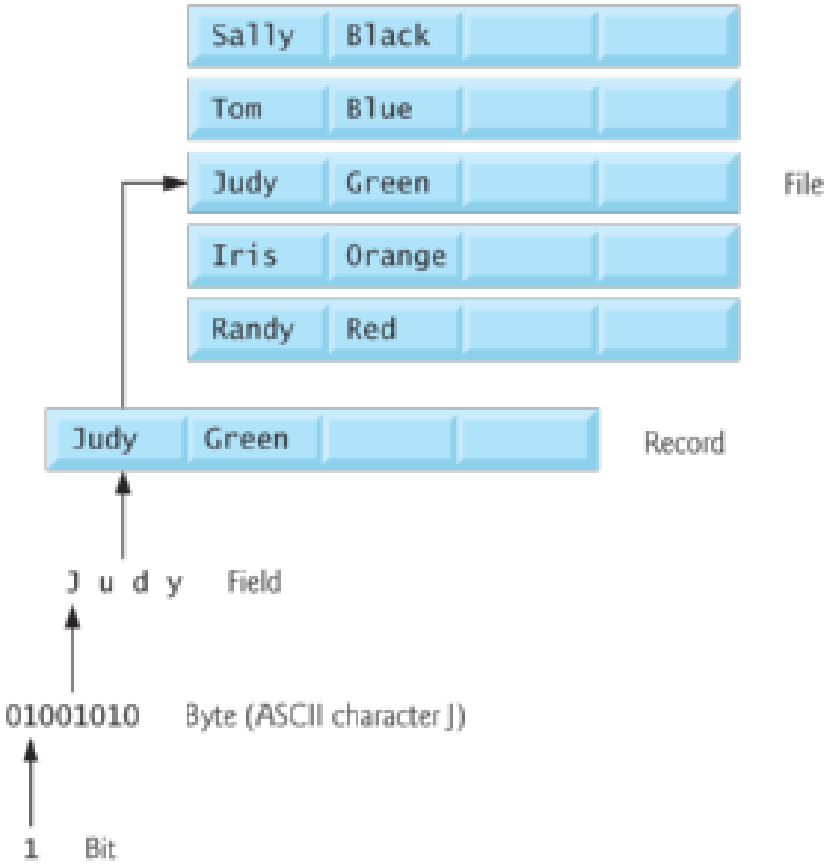
File Operations

- Storage of data in variables and arrays is temporary—such data is lost when a program terminates.
- Files are used for permanent retention of data.
- Computers store files on secondary storage devices, especially disk storage devices.

Data Hierarchy

- All data items processed by a computer are reduced to combinations of zeros and ones.
  - **Bit**: The smallest data item in a computer can assume the value 0 or the value 1.
  - **Byte**: Digits, letters, and special symbols are referred to as characters. Since computers can process only 1s and 0s, every character in a computer's character set is represented as a pattern of 1s and 0s (called a byte). 1 byte = 8 bits
  - **Field**: Composed of characters. Field is a group of character that conveys meaning.
    - Ex: person name
  - **Record**: A group of related fields.
    - Represented by a struct or a class
    - Ex: In a payroll system, a record for a particular employee that contained his/her identification number, name, address, etc.
  - **File**: A group of related records.
    - Ex: Payroll file.
  - **Database**: A group of related files.

Data Hierarchy



Data Hierarchy

- **Record Key:** To facilitate the retrieval of specific records from a file, at least one field in each record is chosen as a record key.
  - Ex: In a school management system student id number could be chosen as a record key.
- **Sequential File:** Most popular way of organizing records in a file.
  - Records typically sorted by record key.

File and Stream

- C views each file as a sequence of bytes
- File ends with the end-of-file marker, or, file ends at a specified byte
- Stream created when a file is opened. Streams provide communication channels between files and programs.
- Provide communication channel between files and programs
- Opening a file returns a pointer to a FILE structure
- Example file pointers:
  - stdin - standard input (enables reading data from keyboard)
  - stdout - standard output (enables printing data on screen)
  - stderr - standard error (screen)

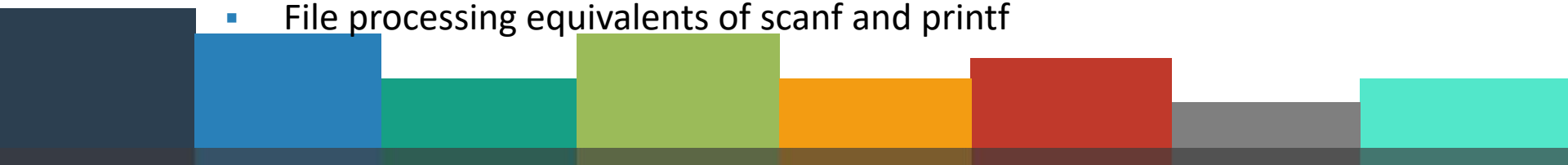
File and Stream

- **File structure:** Opening a file returns a pointer to FILE structure that contain information used to process file.
- **File descriptor:** Index into operating system array called the open file table.
- **File Control Block (FCB):** Found in every array element, system uses it to administer the file.
- Standard input, standard output and standard error are manipulated using file pointers stdin, stdout and stderr

File and Stream

Read/Write functions in standard library

- fgetc
  - Reads one character from a file
  - Takes a FILE pointer as an argument
  - fgetc(stdin) equivalent to getchar()
- fputc
  - Writes one character to a file
  - Takes a FILE pointer and a character to write as an argument
  - fputc('a', stdout) equivalent to putchar('a')
- fgets
  - Reads a line from a file
- fputs
  - Writes a line to a file
- fscanf / fprintf
  - File processing equivalents of scanf and printf



Creating a Sequential Access File

- C imposes no file structure.
- No notion of records in a file.
- Programmer must provide file structure.
- Creating a File:
  - **FILE *myPtr;**
    - Creates a **FILE** pointer called **myPtr**
  - **myPtr = fopen("myFile.dat", openmode);**
    - Function **fopen** returns a **FILE** pointer to file specified
    - Takes two arguments – file to open and file open mode
    - If open fails, NULL returned
  - **fprintf**
    - Used to print to a file
    - Like printf, except first argument is a FILE pointer (pointer to the file you want to print in)

Creating a Sequential Access File

- feof(File Pointer)
  - Returns true if end-of-file indicator (no more data to process) is set for the specified file
- fclose(File Pointer)
  - Closes specified file
  - Performed automatically when program ends
- Details
  - Programs may process no files, one file, or many files
  - Each file must have a unique name and should have its own pointer

Creating a Sequential Access File

- File open modes

Mode	Description
`r`	Open a file for reading.
`w`	Create a file for writing. If the file already exists, discard the current contents.
`a`	Append; open or create a file for writing at end of file.
`r+`	Open a file for update (reading and writing).
`w+`	Create a file for update. If the file already exists, discard the current contents.
`a+`	Append; open or create a file for update; writing is done at the end of the file.

Creating a Sequential Access File


```c
#include <stdio.h>

int main(void)
{
    int hesapNo;
    char ad[30];
    double bakiye;
    FILE *mfPtr; // musteri.dat dosyasi işaretçisi
    if((mfPtr = fopen("musteri.dat","w")) == NULL)
        printf("Dosya acilamadi\n");
    else
    {
        printf("Hesap no, isim ve bakiye girin \n");
        printf("Veri girisini bitirmek icin EOF gir"); //EOF = Ctrl + z
        printf("? ");
        scanf("%d%s%lf",&hesapNo,ad,&bakiye);

        while(!feof(stdin))
        {
            fprintf(mfPtr,"%d %s %.2f \n",
                hesapNo,ad,bakiye);
            printf("? ");
            scanf("%d%s%lf",&hesapNo,ad,&bakiye);
        }

        fclose(mfPtr);
    }
    return 0;
}
```

Creating a Sequential Access File


```
Enter the account, name, and balance.
Enter EOF to end input.
? 100 Jones 24.98
? 200 Doe 345.67
? 300 White 0.00
? 400 Stone -42.16
? 500 Rich 224.62
? ^Z
```

Reading Data From Sequential Access File

- Create a FILE pointer, link it to the file to read
  - **myPtr = fopen("myFile.dat", "r");**
- Use **fscanf** to read from the file
  - Like scanf, except first argument is a FILE pointer
  - **fscanf(myPtr, "%d%s%f", &myInt, myString, &myFloat);**
- Data read from beginning to end
- File position pointer
  - Indicates number of next byte to be read / written
  - Not really a pointer, but an integer value (specifies byte location)
  - Also called byte offset
- **rewind(myPtr)**
  - Repositions file position pointer to beginning of file (byte 0)

Reading Data From Sequential Access File


```c
#include <stdio.h>

int main(void)
{
    int hesapNo;
    char ad[40];
    double bakiye;
    FILE *mfPtr; // musteri.dat dosyasi işaretçisi
    if((mfPtr = fopen("musteri.dat","r")) == NULL)
        printf("Dosya acilamadi\n");
    else
    {
        printf("%-10s%-13s%s\n", "HesapNo","Ad","Bakiye");
        fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);

        while(!feof(mfPtr))
        {
            printf("%-10d%-13s%7.2f\n", hesapNo,ad,bakiye);
            fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);
        }
        fclose(mfPtr);
    }
    return 0;
}
```

Sample Application


```
1  #include <stdio.h>
2
3  int main(void)
4  {
5      int secim, hesapNo;
6      double bakiye;
7      char ad[40];
8      FILE *mfPtr;
9      if((mfPtr = fopen("musteri.dat","r")) == NULL)
10         printf("Dosya acilamadi\n");
11     else
12     {
13         printf("Secim yapiniz\n"
14             "1-Hesapta para olmayan hesaplar\n"
15             "2-Borclu olan hesaplar\n"
16             "3-Hesapta para olan hesaplar\n"
17             "4-Cikis\n");
18         scanf("%d",&secim);
```

Sample Application


```
19          while(secim !=4)
20          {
21              fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);
22              switch(secim)
23              {
24                  case 1:
25                      printf("\nPara olmayan hesaplar :\n");
26                      while(!feof(mfPtr))
27                      {
28                          if(bakiye==0)
29                              printf("%-10d%-13s%7.2f\n", hesapNo,ad,bakiye);
30                          fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);
31                      }
32                      break;
33                  case 2:
34                      printf("\Borclu hesaplar :\n");
35                      while(!feof(mfPtr))
36                      {
37                          if(bakiye<0)
38                              printf("%-10d%-13s%7.2f\n", hesapNo,ad,bakiye);
39                          fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);
40                      }
41                      break;
```

Sample Application

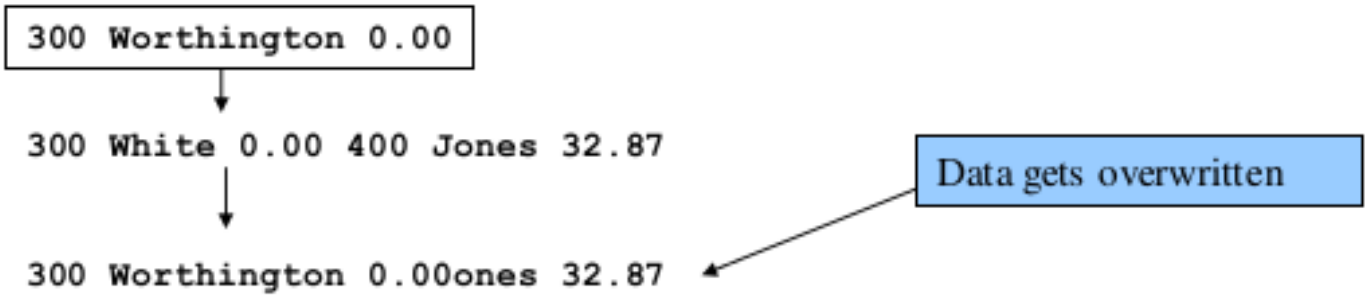

```
42                  case 3:
43                      printf("\nPara olan hesaplar :\n");
44                      while(!feof(mfPtr))
45                      {
46                          if(bakiye>0)
47                              printf("%-10d%-13s%7.2f\n", hesapNo,ad,bakiye);
48                          fscanf(mfPtr,"%d%s%lf",&hesapNo,ad,&bakiye);
49                      }
50                      break;
51              }
52              rewind(mfPtr);
53              printf("\n?");
54              scanf("%d",&secim);
55          }
56          printf("Program sonlandi\n");
57          fclose(mfPtr);
58      }
59  }
```

Sample Application


```
Enter request
 1 - List accounts with zero balances
 2 - List accounts with credit balances
 3 - List accounts with debit balances
 4 - End of run
? 1

Accounts with zero balances:
300       White            0.00

? 2

Accounts with credit balances:
400       Stone          -42.16

? 3

Accounts with debit balances:
100       Jones           24.98
200       Doe            345.67
500       Rich           224.62

? 4
End of run.
```

One Risk at Sequential Access File

- Sequential access file
  - Cannot be modified without the risk of destroying other data
  - Fields can vary in size
    - Different representation in files and screen than internal representation
    - 1, 34, -890 are all ints, but have different sizes on disk
- 300 White 0.00 400 Jones 32.87 (old data in file)
- If we want to change White's name to Worthington



Next Week

- File Operations
- Random Access Files

References

- Doç. Dr. Fahri Vatansever, “Algoritma Geliştirme ve Programlamaya Giriş”, Seçkin Yayıncılık, 12. Baskı, 2015.
- Kaan Aslan, “A’dan Z’ye C Klavuzu 8. Basım”, Pusula Yayıncılık, 2002.
- Paul J. Deitel, “C How to Program”, Harvey Deitel.
- “A book on C”, All Kelley, İra Pohl

Thanks for listening

CANER ÖZCAN
canerozcan@karabuk.edu.tr

"I think everybody in this country should learn how to program a computer because it teaches you how to think."

– Steve Jobs