PRAKTIKUM 6
OPERASI FILE

1. Tujuan

a. Mahasiswa mampu mengetahui konsep pengoperasian file pada C
b. Mahasiswa mampu membedakan jenis file teks maupun binner
c. Mahasiswa mampu melakukan pengoperasian file menggunakan C

2. Materi

File merupakan sebuah organisasi dari sejumlah data record yang tersimpan dengan format tertentu. Masing-masing record bisa terdiri dari satu atau beberapa field, dan setiap field terdiri dari satu atau beberapa byte. Lalu terdapat file data yaitu suatu file yang digunakan sebagai tempat penyimpanan data sehingga data dapat tersimpan secara permanen.

Pengoperasian file adalah bentuk program yang melibatkan data yang tersimpan dalam betuk file. Data yang disimpan dalam bentuk file tersebut akan dilakukan pengoperasian yang dapat ditulis atau juga dibaca oleh program, secara umum format file yang dapat diakses ialah berbentuk .csv, .txt dan lainnya. File dibagi menjadi dua jenis yaitu file teks dan juga file biner, untuk lebih jelasnya silahkan lihat dibawah ini:

a. File Teks

   File teks adalah file yang dapat menyimpan data dengan pola karakter atau string. File ini sangat umum untuk kita temukan, file teks biasanya dibuat dengan teks editor dengan format .txt, .csv, dan format lainnya. File teks sendiri ialah file yang mudah untuk dilakukan baca tulis.

b. File Biner

   File biner adalah file dengan pola penyimpanan yang berada dalam disk dan berbentuk biner (0 dan 1), file biner digunakan untuk menyimpan data kompleks, misalnya seperti struct. File biner sendiri mampu menyimpan file yang lebih banyak, memiliki keamanan tinggi dan sukit untuk dibaca. Contoh format file biner adalah .exe, dan .bin seperti yang ada pada komputer kita.

Perlu di ingat bahwa program pengoperasian file ialah menggunakan pointer, baik dalam pengoperasian file teks dan juga file biner. Semua fungsi pengoperasian file terdapat pada library `stdio.h`.

Pada pengoperasian file terdapat beberapa fungsi yang digunakan untuk melakukan pengoperasian file yang dapat dilihat pada tabel dibawah ini:

Tabel 6.1 Function File Processing

Function	Deskripsi
fopen( )	Membuka file
fclose( )	Menutup file
fprintf( )	Mencetak file
fscanf( )	Membaca isi file
feof( )	Mendeteksi baris
fputc( )	Menulis 1 karakter pada file
fgetc( )	Membaca 1 karakter pada file
fseek( )	Mengatur pointer file ke posisi tertentu
fputw( )	Menulis 1 angka pada file
fgetw( )	Membaca 1 angka pada file
ftell( )	Return ke posisi sekarang
rewind( )	Mengatur file pointer ke awal file

Secara umum pengoperasian file itu terdari tiga fungsi yaitu read (r), write (w), dan append (a), namun juga terdapat beberapa metode akses file yang tersedia pada bahasa C, dapat dilihat pada tabel dibawa ini:

Tabel 6.2 Mode Akses File

Mode Akses	Keterangan
r	Membuka sebuah file untuk dibaca
w	Membuat sebuah file untuk ditulis
a	Membuka file untuk penambah data
r+	Membuka sebuah file untuk dibaca / ditulis
w+	Membuat sebuah file untuk dibaca / ditulis
a+	Membuka file untuk penambahan data / dibaca
rb	Membuka sebuah file binary untuk dibaca
wb	Membuka sebuah file binary untuk ditulis
ab	Membuka file binary untuk penambahan data
rb+	Membuka sebuah file binary untuk dibaca / ditulis
wb+	Membuat sebuah file binary untuk dibaca / ditulis
ab+	Membuka file binary untuk penambahan data / dibaca

Untuk membuat suatu program pengoperasian file kita harus memahami beberapa hal berikut:

a. Membuka File

Untuk melakukan pemrosesan hal pertama yang harus dilakukan ialah membuka file, kita dapat menggukana fungsi `fopen( )`, fungsi tersebut selain untuk membuka juga akan membuat file secara otomatis jika belum dibuat. Penggunaan fungsi `fopen( )` terdapat pada library `stdio.h`. berikut ini adalah bentuk umum dari fungsi `fopen( )`:

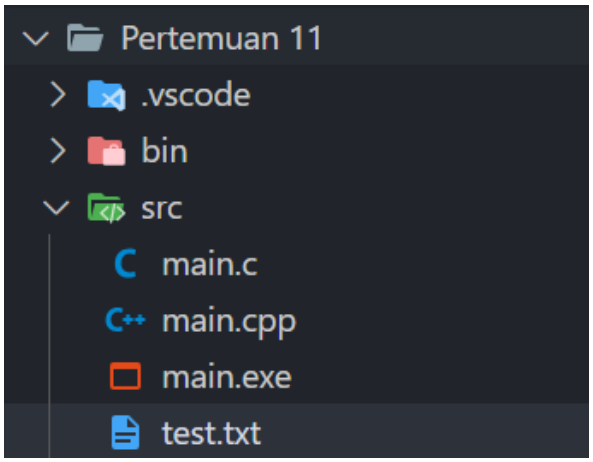
```
FILE *variabel = fopen("nama file .format file", "mode akses");
```

Contoh:

```
FILE *in=fopen("test.txt","r");
```

Gambar 6.1 Membuka File dengan fopen( )

File program .C yang kita buat harus berada satu folder dengan file yang akan kita lakukan proses, seperti contoh diatas dengan file test.txt.



Gambar 6.2 Pengoperasian File Test.txt

b. Menutup File

Ketika membuka suatu file maka kita juga harus menutup file tersebut, untuk menutup file kita dapat menggunakan fungsi `fclose( )`.

c. Memproses file

Pemrosesan file sendiri terbagi menjadi tiga seperti yang sudah disebutkan diatas yaitu membaca (read), menulis (write) dan menyisipkan data (append).

1. Membuka dan Menulis File

   Hal pertama dalam melakukan pengoperasian file ialah membuka atau membuat file dan menuliskan atau menyimpan data pada file yang kita

buat. Perhatikan contoh program dibawah ini:

```
char nama[100];
int umur;
printf("Masukkan nama : ");
scanf("%[^\n]", &nama); //input %[^\n] untuk spasi
fflush(stdin); //membersihkan buffer input standar (stdin)
printf("Masukkan umur : ");
scanf("%d", &umur);
fflush(stdin); //membersihkan buffer input standar (stdin)

FILE *out=fopen("text.txt","w");
//membuka file text.txt dengan mode write pada variabel pointer *out
fprintf(out,"%s#%d\n",nama, umur);
//menambahkan inputan nama dan umur pada file dengan memanggil variabel out
fclose(out); //menutup file
```

Gambar 6.3 Menulis Data File

2. Membuka dan Membaca File

Setelah berhasil melakukan penulisan pada file, selanjutnya kita akan membaca file tersebut dan menampilkannya. Untuk membaca isi file kita dapat menampungnya pada sebuah variabel, kita dapat menggunakan `fscanf(nama variabel fie, format);` silahkan perhatikan program berikut ini:

```
FILE *in=fopen("text.txt","r");
//membuka file text.txt dengan mode read pada variabel pointer *in
    if(!in)//melakukan cek apakah file tersedia atau tidak
    {
       printf("tidak ada file");
    }
    else
    {
       while(!feof(in)) //mendeteksi baris pada isi file dengan perulangan
       {
          fscanf(in,"%[^#]#%d\n", &nama, &umur);
          //menampung isi data file pada variabel nama dan umur
          fflush(stdin);//membersihkan buffer input standar (stdin)
          printf("%s %d\n", nama, umur);
          //menampilkan isi file yang berada pada variabel nama dan umur
       }
       fclose(in);
    }
    getchar();
```

Gambar 6.4 Menampilkan Data File

3. Membuka dan Menyisipkan Data File

Menyisipkan data berfungsi untuk menambahkan data baru pada baris berikutnya, sebab jika tidak menyisipkan data ketika melakukan inputan maka data yang sudah diinputkan atau yang sudah ada sebelumnya akan ditimpah atau dilakukan overwriting. Namun jika

menyisipkan data menggunakan append maka data yang sudah ada tidak akan hilang, dan akan menambahkan data baru pada baris berikutnya. Perhatikan program berikut ini:

```
char nama[100];
int umur;
printf("Masukkan nama : ");
scanf("%[^\n]", &nama); //input %[^\n] untuk spasi
fflush(stdin); //membersihkan buffer input standar (stdin)
printf("Masukkan umur : ");
scanf("%d", &umur);
fflush(stdin); //membersihkan buffer input standar (stdin)

FILE *out=fopen("text.txt","a");
//membuka file text.txt dengan mode append pada variabel pointer *out
fprintf(out,"%s#%d\n",nama, umur);
//menambahkan inputan nama dan umur yang baru pada file dengan memanggil variabel out
fclose(out); //menutup file
printf("Sukses menambah data.");
getchar();
```

Gambar 6.5 Menyisipkan Data File

Lalu mengapa kita perlu menggunakan pengoperasian file pada bahasa C?, berikut ini adalah alasan mengapa kita harus menggunakan pengoperasian file pada bahasa C:

a. Dapat digunakan kembali: Data yang disimpan dalam file dapat diakses, diperbarui, dan dihapus di mana saja dan kapan saja sehingga memberikan kemampuan penggunaan kembali yang tinggi,
b. Portabilitas: Tanpa kehilangan data apa pun, file dapat ditransfer ke file lain di sistem komputer. Risiko kesalahan pengkodean diminimalkan dengan fitur ini,
c. Efisien: Sejumlah besar masukan mungkin diperlukan untuk beberapa program. Penanganan file memungkinkan Anda mengakses bagian file dengan mudah menggunakan beberapa instruksi yang menghemat banyak waktu dan mengurangi kemungkinan kesalahan,
d. Kapasitas Penyimpanan: File memungkinkan Anda menyimpan data dalam jumlah besar tanpa harus khawatir menyimpan semuanya secara bersamaan dalam suatu program.

3. Aktifitas Mahasiswa

a. Silahkan duplikat folder Pertemuan_1 dan beri nama baru menjadi Pertemuan_6 (tujuannya agar tidak melakukan setting ulang compiler)
b. Selanjutnya pada file main.c silahkan ketikkan program berikut:
    1. Membuat file text format .txt dengan akses mode write

```c
// C Program to create a file
#include <stdio.h>
#include <stdlib.h>

int main()
{
    // file pointer
    FILE* fptr;

    // creating file using fopen() access mode "w"
    fptr = fopen("file.txt", "w");

    // checking if the file is created
    if (fptr == NULL) {
        printf("file tidak dapat dibuka!");
        exit(0);
    }
    else {
        printf("Program berhasil dibuat");
    }

    return 0;
}
```

2. Membuat file, menuliskan file, membaca file, menampilkan file, dan menutup file dengan format file .C (pengoperasian file dengan format file .C)

```c
// C program to Open a File,
// Write in it, And Close the File
#include <stdio.h>
#include <string.h>

int main()
{
    // deklarasi file pointer
    FILE* filePointer;
    // membuat data untuk dituliskan pada file dengan variabel array dataToBeWritten
    char dataToBeWritten[50] = "Dasar Pemrograman Praktikum Semester 1 ";
    // membuka file file-test.c dengan fungsi fopen()
    // menggunakan mode write dengan w untuk menulis file
    filePointer = fopen("file-test.c", "w");
    // melakukan cek file file-test.c untuk dilakukan pengoperasian
    if (filePointer == NULL)
    {
        printf("file-test.c file tidak dapat dibuka.");
    }
    else
    {
        printf("file sudah dibuka dan dapat dioperasikan.\n");
        // menuliskan data pada variabel array dataToBeWritten pada file file-test.c
        if (strlen(dataToBeWritten) > 0)
        {
            // menuliskan data ke file dengan fungsi fputs()
```

```c
            fputs(dataToBeWritten, filePointer);
            fputs("\n", filePointer);
        }
        // menutup file dengan fungsi fclose()
        fclose(filePointer);

        printf("Data berhasil di tulis pada file file-test.c\n");
        printf("file sekarang di tutup.");
    }

    printf("\n\n");

    // membuka file file-test.c dengan fungsi fopen()
    // menggunakan mode read dengan r untuk membaca file
    filePointer = fopen("file-test.c", "r");
    // melakukan cek file file-test.c untuk dilakukan pengoperasian
    if (filePointer == NULL)
    {
        printf("file-test.c tidak dapat dibuka.");
    }
    else
    {
        printf("file sudah dibuka dan dapat dioperasikan.\n");
        // membaca data variabel array dataToBeWritten pada file file-test.c
        // menggunakan fungsi fegets()
        while (fgets(dataToBeWritten, 50, filePointer) != NULL)
        {
            //menampilkan data
            printf("isi file-test.c adalah %s", dataToBeWritten);
        }
        // menutup file dengan fungsi fclose()
        fclose(filePointer);

        printf("berhasil membaca data pada file file-test.c\n");
        printf("file sekarang di tutup.");
    }
    return 0;
}
```

***Note**: Jika tidak berhasil melakukan run dengan VsCode, silahkan run program dengan compiler online.*

3. Pengoperasian file dengan file binary (membuat file, menulis file dan membaca file)

```c
// C program to write to a Binary file using fwrite()
#include <stdio.h>
#include <stdlib.h>

struct threeNum //membuat struktur
{
    int n1, n2, n3;
};

int main()
{
    int n;
    // deklarasi struktur
    struct threeNum num;
    //membuat pointer file
    FILE* fptr;
    if ((fptr = fopen("program.bin", "wb")) == NULL) //membuka dan membuat file
    {
        printf("Error! opening file");
        // melakukan cek file,
        //jika file biner tidak terbaca program berhenti.
        exit(1);
    }
```

```c
    int flag = 0;
    // mengembalikan nilai pointer pada file (menulis data).
    for (n = 1; n < 5; ++n)
    {
        num.n1 = n;
        num.n2 = 5 * n;
        num.n3 = 5 * n + 1;
        //menulis data pada file binary
        flag = fwrite(&num, sizeof(struct threeNum), 1,fptr);
    }
    // melakukan cek jika data sudah di tulis pada file binary
    if (!flag)
    {
        printf("Write Operation Failure");
    }
    else
    {
        printf("Write Operation Successful");
    }
    fclose(fptr); //menutup file

    printf("\n\n");

    if ((fptr = fopen("program.bin", "rb")) == NULL) //membuka dan membaca file
    {
        printf("Error! opening file");
        // melakukan cek file,
        //jika file biner tidak terbaca program berhenti.
        exit(1);
    }
    // mengembalikan nilai pointer pada file (menampikan data).
    for (n = 1; n < 5; ++n)
    {
        //membaca data file perbaris
        fread(&num, sizeof(struct threeNum), 1, fptr);
        //menampilkan data file yang sudah dibaca perbaris
        printf("n1: %d\tn2: %d\tn3: %d\n", num.n1, num.n2, num.n3);
    }
    fclose(fptr); //menutup file

    return 0;
}
```

4. Pengoperasian file dengan file .csv

```c
// C program for the above approach
#include <conio.h>
#include <stdio.h>
#include <string.h>

int main()
{
    // membuka dan membuat file untuk append data
    FILE* fp = fopen("operation.csv", "a+");
    char name[50];
    int accountno, amount;
    if (!fp) {
        // jika file tidak ada maka program berhenti
        printf("Can't open file\n");
        return 0;
    }
    // menambahkan record data
    printf("Enter Account Holder Name : ");
    scanf("%s", &name);
```

```c
    printf("Enter Account Number : ");
    scanf("%d", &accountno);
    printf("Enter Available Amount : ");
    scanf("%d", &amount);
    // menyimpan data pada file
    fprintf(fp, "%s, %d, %d\n", name, accountno, amount);
    printf("\nNew Account added to record");
    fclose(fp); //menutup file

    printf("\n\n\n");

    // membuka dan membaca file
    FILE* fpp = fopen("operation.csv", "r+");
    if (!fpp)
    {
        // jika file tidak ada maka program berhenti
        printf("Can't open file\n");
        return 0;
    }
    else
    {
        // membuat buffer array
        char buffer[1024];
        int row = 0;
        int column = 0;
        while (fgets(buffer, 1024, fpp))
        {
            column = 0;
            row++;
            // Tpenyesuaian kolom data
            if (row == 0)
                continue;
            // split data
            char* value = strtok(buffer, ", ");

            while (value) {
                // Column 1
                if (column == 0) {
                    printf("Name :");
                }
                // Column 2
                if (column == 1) {
                    printf("\tAccount No. :");
                }
                // Column 3
                if (column == 2) {
                    printf("\tAmount :");
                }
                printf("%s", value); //menampilkan data
                value = strtok(NULL, ", ");
                column++;
            }
            // printf("\n");
        }
        // Close the file
        fclose(fpp);
    }
    return 0;
}
```

c. Ikuti perintah praktikum!

4. Tugas

Silahkan buat aplikasi rekam medis dan lakukan penyimpanan data yang ada pada program aplikasi rekam medis yang sudah dibuat menggunakan pengolahan file dalam bentuk format file .**csv**!

Tema: "Rekam Medis Klinik Dokter Aisyiyyah"

5. Daftar Pustaka

Solichin Achmad, 2003, Pemrograman Bahasa C dengan Turbo C, IlmuKomputer.com

Mieke Yuliana, 2010, Operasi File 2, Politeknik Elektronika Negeri Surabaya

Kusuma Purba Daru, 2020, Algortima dan Pemrograman, Deepublish

Suryana Febriyanno, Arsyah U. Ilhami, dan Pratiwi Mutiana, 2023, Algoritma dan Pemrograman dengan Bahasa C/C++, CV. Mitra Cendekia Media

Vandoro Hery, 2016, Operasi FILE dalam Bahasa C, Mahirkoding.com (https://www.mahirkoding.com/operasi-file-dalam-bahasa-c/, diakses 26 Desember 2023)

Muhardian Ahmad, 2019, Belajar Pemrograman C #17: Cara Membaca dan Menulis File di C, Petanikode.com (https://www.petanikode.com/c-file/, diakses 26 Desember 2023)

Sobat Ambisius, 2021, Belajar Bahasa C: #11 File Processing, Sobatambisius.com (https://www.sobatambisius.com/2021/09/belajar-bahasa-c-11-file-processing.html, diakses 26 Desember 2023)

Rishabh Prabhu, dkk, 2023, Basics of File Handling in C, Geeksforgeeks.org (https://www.geeksforgeeks.org/basics-file-handling-c/, diakses 27 Desember 2023)

Sahilkatar, 2023, Relational Database from CSV Files in C, Geeksforgeeks.org (https://www.geeksforgeeks.org/relational-database-from-csv-files-in-c/, diakses 27 Desember 2023)

Kaydeenz Jr., Modul - Operasi File dalam Bahasa C, Scribd.com (https://www.scribd.com/document/631052775/MODUL-Operasi-File-dalam-bahasa-C, diakses 26 Desember 2023)