Final Project Pemrograman Lanjut

Pembuatan aplikasi komik sederhana menggunakan bahasa C

Disusun oleh :

1. Syafira Nurul Hanifa	NPM. 23081010216
2. Jurita Sitanggang	NPM. 23081010228
3. Yazid Fadhlurrohman	NPM. 23081010274
4. Rega Yuancahya Faiz Kristama	NPM. 23081010033
5. Mochamad David Ainur R	NPM. 23081010239

MATA KULIAH PEMROGRAMAN LANJUT

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" JAWA TIMUR

SURABAYA

2024

DAFTAR ISI

BAB 1

IMPLEMENTASI MATERI

A. Materi struct

Structure atau struct adalah kumpulan dari beberapa variabel dengan beragam tipe data yang dibungkus dalam satu variabel. Program ini menggunakan struct untuk membungkus beberapa tipe data sekaligus dan menyimpan informasi yang berbeda tetapi saling terkait (seperti nama komik, nama komikus, popularitas, rating, dan tahun rilis) dalam satu variabel agar kode lebih mudah dipahami dan mudah untuk mengelola data

```
#define MAX_COMICS 100
#define MAX_NAME_LENGTH 100
#define MAX_KOMIKUS_LENGTH 100

typedef struct {
    char name[MAX_NAME_LENGTH];
    char komikus[MAX_KOMIKUS_LENGTH];
    int populer;
    float rating;
    int tahun;
} Comic;

Comic comics[MAX_COMICS];
```

Gambar 1. Penggunaan struct

Pada awal program kami mendefinisikan max comics, max name length, dan max komikus length yang kami tetapkan jumlahnya yaitu 100 untuk memastikan bahwa data yang dimasukkan tidak melebihi kapasitas yang telah ditentukan, sehingga menghindari potensi kesalahan memori selanjutnya kami mendefinisikan sebuah struktur bernama Comic menggunakan typedef. Struktur ini berfungsi sebagai template untuk menyimpan informasi terkait setiap komik. Struktur Comic memiliki lima anggota: name, yang merupakan array karakter untuk menyimpan nama komik; komikus, yang juga merupakan array karakter untuk menyimpan nama komikus; populer, yang merupakan variabel bertipe integer untuk menyimpan tingkat popularitas komik; rating, yang merupakan variabel bertipe float untuk menyimpan rating komik dalam rentang 0 hingga

10; dan tahun, yang merupakan variabel bertipe integer untuk menyimpan tahun rilis komik.Terakhir, sebuah array bernama comics dideklarasikan dengan tipe Comic dan ukuran MAX_COMICS. Array ini akan digunakan untuk menyimpan hingga 100 objek Comic, memungkinkan program untuk mengelola dan mengakses data komik dengan mudah. Dengan struktur ini, program dapat dengan menyimpan, mengelola, dan memanipulasi informasi tentang berbagai komik yang ada

B. Materi Fungsi

Fungsi merupakan kumpulan kode yang memiliki tujuan tertentu dan dapat dipanggil oleh kode yang lain. Pada materi ini kami membuat beberapa fungsi yang digunakan untuk membuat beberapa menu yaitu menu tambah komik untuk user menginputkan komik, tampilkan komik untuk menampilkan komik yang sudah user inputkan, cari komik untuk user mencari komik berdasarkan kategori yang ingin dicari dan fungsi main yang merupakan fungsi utama untuk memanggil fungsi sebelumnya dan menjalankan program

1. Fungsi tambah komik

```c
void tambah_komik() {
    Comic komik_baru;
    FILE *file = fopen("komik.txt", "w");
    if (!file) {
        printf("file tidak dapat dibuka untuk membaca.\n");
        return;
    }

    while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d", comics[hitung].name, comics[hitung].komikus,
                  &comics[hitung].populer, &comics[hitung].rating, &comics[hitung].tahun) == 5) {
        hitung++;
    }
    fclose(file);

    printf("masukkan nama komik: ");
    getchar(); // Bersihkan buffer
    fgets(komik_baru.name, MAX_NAME_LENGTH, stdin);
    komik_baru.name[strcspn(komik_baru.name, "\n")] = '\0';

    printf("masukkan nama komikus: ");
    fgets(komik_baru.komikus, MAX_KOMIKUS_LENGTH, stdin);
    komik_baru.komikus[strcspn(komik_baru.komikus, "\n")] = '\0';

    printf("masukkan popularitas (angka): ");
    while (scanf("%d", &komik_baru.populer) != 1) {
        printf("input invalid! masukkan angka: ");
        while (getchar() != '\n');
    }

    printf("masukkan rating (0-10): ");
    while (scanf("%f", &komik_baru.rating) != 1 || komik_baru.rating < 0 || komik_baru.rating > 10) {
        printf("input invalid! masukkan rating antara 0-10: ");
        while (getchar() != '\n');
    }
```

```
    printf("masukkan tahun: ");
    while (scanf("%d", &komik_baru.tahun) != 1) {
        printf("input invalid! masukkan angka: ");
        while (getchar() != '\n');
    }

    comics[hitung] = komik_baru;
    hitung++;

    file = fopen("komik.txt", "w");
    if (!file) {
        printf("file tidak dapat dibuka untuk menulis.\n");
        return;
    }

    for (int i = 0; i < hitung; i++) {
        fprintf(file, "%s\n%s\n%d\n%.2f\n%d\n", comics[i].name, comics[i ].komikus, comics[i].populer,
                comics[i].rating, comics[i].tahun);
    }

    printf("komik berhasil ditambahkan.\n");
    fclose(file);
}
```

Gambar 2. Fungsi tambah komik

Pada fungsi tambah komik ini berguna untuk user menginputkan komik yang ingin ditambahkan lalu data komik baru yang sudah diinputkan akan masuk kedalam array dan menyimpan semua data termasuk data baru ke dalam file komik txt. Pada fungsi ini mencakup proses pembacaan data lama, pengumpulan input data baru dari pengguna, dan penulisan komik baru kedalam file. Untuk fungsi ini pertama kami mendeklarasikan Sebuah variabel komik_baru bertipe Comic digunakan untuk menyimpan data komik baru yang diinputkan oleh pengguna lalu program akan membaca Data komik yang sudah ada dibaca dari file komik.TXT menggunakan fscanf dan setiap data yang berhasil dibaca disimpan ke dalam elemen array comics, dan variabel hitung ditingkatkan untuk melacak jumlah komik. Lalu user akan diminta untuk data berupa nama komik, nama komikus, popularitas (angka), rating (dalam rentang 0–10), dan tahun terbit (angka).Setelah input selesai, data dari komik_baru disimpan ke dalam array comics di indeks berikutnya yang kosong dan disimpan ke dalam file komik. Setelah selesai maka program akan menampilkan pesan bahwa komik telah berhasil ditambahkan, yang menandakan fungsi selesai dengan sukses.

2. Fungsi tampilkan komik

```c
void tampilkan_komik() {
    hitung = 0;

    FILE *file = fopen("komik.txt", "r");
    if (!file) {
        printf("File tidak dapat dibuka untuk membaca.\n");
        return;
    }

    while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d", comics[hitung].name, comics[hitung].komikus,
                  &comics[hitung].populer, &comics[hitung].rating, &comics[hitung].tahun) == 5) {
        hitung++;
    }
    fclose(file);

    if (hitung == 0) {
        printf("Tidak ada komik untuk ditampilkan.\n");
        return;
    }

    int pilih;
    printf("\ntampilkan komik berdasarkan:\n");
    printf("1. nama komik\n");
    printf("2. nama komikus\n");
    printf("3. popularitas\n");
    printf("4. rating\n");
    printf("5. tahun\n");
    printf("pilih menu: ");
    scanf("%d", &pilih);
```

```c
switch (pilih) {
    case 1:
        for (int i = 0; i < hitung - 1; i++) {
            for (int j = i + 1; j < hitung; j++) {
                if (strcmp(comics[i].name, comics[j].name) > 0) {
                    Comic temp = comics[i];
                    comics[i] = comics[j];
                    comics[j] = temp;
                }
            }
        }
        printf("\ndaftar komik:\n");
        for (int i = 0; i < hitung; i++) {
            printf("nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
                   comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
        }
        break;
    case 2:
        for (int i = 0; i < hitung - 1; i++) {
            for (int j = i + 1; j < hitung; j++) {
                if (strcmp(comics[i].komikus, comics[j].komikus) > 0) {
                    Comic temp = comics[i];
                    comics[i] = comics[j];
                    comics[j] = temp;
                }
            }
        }
        printf("\ndaftar komik:\n");
        for (int i = 0; i < hitung; i++) {
            printf("komikus: %s | nama: %s | populer: %d | rating: %.2f | tahun: %d\n",
                   comics[i].komikus, comics[i].name, comics[i].populer, comics[i].rating, comics[i].tahun);
        }
        break;
```

```
        case 3:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].populer < comics[j].populer) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("populer: %d | nama: %s | komikus: %s | rating: %.2f | tahun: %d\n",
                       comics[i].populer, comics[i].name, comics[i].komikus, comics[i].rating, comics[i].tahun);
            }
            break;
        case 4:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].rating < comics[j].rating) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("rating: %.2f | nama: %s | komikus: %s | populer: %d | tahun: %d\n",
                       comics[i].rating, comics[i].name, comics[i].komikus, comics[i].populer, comics[i].tahun);
            }
            break;
```

```
        case 5:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].tahun > comics[j].tahun) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("tahun: %d | nama: %s | komikus: %s | populer: %d | rating: %.2f\n",
                       comics[i].tahun, comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating);
            }
            break;
        default:
            printf("pilihan tidak valid.\n");
            return;
    }
}
```

Gambar 3. Fungsi tampilkan komik

Fungsi tampilkan komik ini kami gunakan untuk menampilkan komik berdasarkan nama komik, nama komikus, rating, popularitas, dan tahun dimana akan ditampilkan berdasarkan komik yang sudah disimpan pada file komik.txt.

Pada fungsi ini File komik.txt dibuka dalam mode r (read) untuk membaca data. Jika file tidak bisa dibuka, fungsi akan mencetak pesan error dan keluar Data dibaca dari file menggunakan fscanf dan Setiap data yang berhasil dibaca disimpan dalam elemen array comics, dan variabel hitung ditingkatkan untuk melacak jumlah komik yang berhasil dibaca. Jika variabel hitung tetap nol setelah pembacaan, artinya tidak ada data komik yang tersimpan di file. Program akan mencetak pesan bahwa tidak ada komik untuk ditampilkan, lalu keluar dari

fungsi.lalu user akan memilih menampilkan komik berdasarkan nama komik, nama komikus, rating, popularitas atau tahun lalu program mencetak semua data dalam array comics dengan format sesuai kriteria pengurutan

3. Fungsi cari komik

```
void cari_komik() {
    if (hitung == 0) {
        printf("tidak ada komik untuk dicari.\n");
        return;
    }

    int pilih;
    printf("cari komik berdasarkan:\n");
    printf("1. nama komik\n");
    printf("2. nama komikus\n");
    printf("3. popularitas\n");
    printf("4. rating\n");
    printf("5. tahun\n");
    printf("pilih menu: ");
    scanf("%d", &pilih);

    switch (pilih) {
        case 1: {
            char cari_nama[MAX_NAME_LENGTH];
            printf("masukkan nama komik yang ingin dicari: ");
            getchar();
            fgets(cari_nama, MAX_NAME_LENGTH, stdin);
            cari_nama[strcspn(cari_nama, "\n")] = '\0';

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (strcasecmp(comics[i].name, cari_nama) == 0) {
                    printf("komik ditemukan: nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                    break;
                }
            }
            if (!found) {
                printf("komik dengan nama '%s' tidak ditemukan.\n", cari_nama);
            }
            break;
        }
```

```
        case 2: {
            char cari_komikus[MAX_KOMIKUS_LENGTH];
            printf("masukkan nama komikus yang ingin dicari: ");
            getchar();
            fgets(cari_komikus, MAX_KOMIKUS_LENGTH, stdin);
            cari_komikus[strcspn(cari_komikus, "\n")] = '\0';

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (strcasecmp(comics[i].komikus, cari_komikus) == 0) {
                    printf("komik ditemukan: nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                }
            }
            if (!found) {
                printf("komikus '%s' tidak ditemukan.\n", cari_komikus);
            }
            break;
        }
```

```c
case 3: {
    int cari_populer;
    printf("masukkan popularitas yang ingin dicari: ");
    scanf("%d", &cari_populer);

    int found = 0;
    for (int i = 0; i < hitung; i++) {
        if (comics[i].populer == cari_populer) {
            printf("komik ditemukan: nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
                   comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
            found = 1;
        }
    }
    if (!found) {
        printf("tidak ada komik dengan popularitas %d.\n", cari_populer);
    }
    break;
}
```

```c
case 4: {
    float cari_rating;
    printf("masukkan rating yang ingin dicari: ");
    scanf("%f", &cari_rating);

    int found = 0;
    for (int i = 0; i < hitung; i++) {
        if (comics[i].rating == cari_rating) {
            printf("komik ditemukan: nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
                   comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
            found = 1;
        }
    }
    if (!found) {
        printf("tidak ada komik dengan rating %.2f.\n", cari_rating);
    }
    break;
}
```

Gambar 4. Fungsi cari komik

Pada fungsi ini, pengguna dapat mencari komik berdasarkan nama komik, nama komikus, rating, popularitas, atau tahun. Fungsi ini mencakup langkah-langkah validasi data, pencarian data sesuai kriteria yang dipilih, serta penampilan hasil pencarian.Pertama, fungsi memeriksa apakah array comics kosong dengan mengecek nilai variabel hitung. Jika hitung bernilai nol, program mencetak pesan "tidak ada komik untuk dicari" dan keluar dari fungsi, karena tidak ada data yang dapat dicari. Namun, jika data tersedia, pengguna diminta memilih kriteria pencarian melalui menu yang mencakup opsi pencarian berdasarkan nama komik, nama komikus, rating, popularitas, atau tahun.Setelah pengguna memilih kriteria, program meminta input sesuai dengan kriteria yang dipilih. Misalnya, untuk pencarian berdasarkan nama komik, program meminta pengguna memasukkan nama komik yang ingin dicari. Nama yang dimasukkan pengguna kemudian dibandingkan dengan data dalam array comics menggunakan fungsi strcasecmp, yang tidak membedakan huruf besar dan kecil. Jika ditemukan kecocokan, data komik yang sesuai ditampilkan, dan variabel found disetel ke 1 untuk menandai bahwa data telah ditemukan. Jika setelah iterasi tidak ada kecocokan yang

ditemukan, program mencetak pesan bahwa data yang dicari tidak ada.Proses serupa dilakukan untuk kriteria pencarian lainnya, seperti nama komikus, popularitas, rating, dan tahun. Untuk masing-masing kriteria, program membandingkan input pengguna dengan atribut yang relevan dari setiap elemen dalam array comics. Jika data ditemukan, program mencetak informasi lengkap tentang komik, yang meliputi nama komik, nama komikus, popularitas, rating, dan tahun terbit.Jika semua data telah selesai diproses tetapi tidak ditemukan kecocokan, program mencetak pesan bahwa data yang dicari tidak ditemukan

4. **Fungsi main**

```c
int main() {
    int pilih;
    do {
        printf("\nmenu:\n");
        printf("1. tambah komik\n");
        printf("2. tampilkan komik\n");
        printf("3. cari komik\n");
        printf("4. keluar\n");
        printf("pilih menu: ");
        scanf("%d", &pilih);

        switch (pilih) {
            case 1:
                tambah_komik();
                break;
            case 2:
                tampilkan_komik();
                break;
            case 3:
                cari_komik();
                break;
            case 4:
                printf("keluar program...\n");
                break;
            default:
                printf("pilihan tidak valid.\n");
        }
    } while (pilih != 4);
    return 0;
}
```

Gambar 5. Fungsi main

Pada fungsi main digunakan untuk menjalankan menu utama yang memungkinkan pengguna untuk mengakses berbagai menu aplikasi. Fungsi ini menggunakan struktur do-while loop untuk menampilkan menu secara terus-menerus hingga pengguna memilih untuk keluar dari program. Di awal, variabel pilih dideklarasikan untuk menyimpan pilihan menu yang dimasukkan oleh pengguna. Selama loop berjalan, menu utama ditampilkan dengan opsi

Tambah Komik, Tampilkan Komik, Cari Komik , Keluar. Setiap kali menu ditampilkan, pengguna diminta untuk memasukkan angka pilihan menu. Nilai yang dimasukkan disimpan di variabel pilih. Kemudian, program memproses input pengguna menggunakan struktur switch-case dan memanggil fungsi yang sudah dibuat sebelumnya. Jika pengguna memasukkan angka selain 1-4, program menampilkan pesan "pilihan tidak valid" sebagai bentuk validasi input dan meminta pengguna untuk memilih ulang. Loop ini akan terus berjalan hingga pengguna memilih opsi Keluar (4). Pada akhirnya, program mengembalikan nilai 0 untuk menandakan bahwa eksekusi telah selesai

C. MATERI SORTING

Sorting (pengurutan) adalah proses untuk menyusun data dalam urutan tertentu, baik dalam urutan menaik (ascending) maupun menurun (descending). Salah satu algoritma sorting yang umum digunakan adalah **Bubble Sort**. Algoritma ini bekerja dengan cara membandingkan elemen-elemen yang berdekatan dalam array atau daftar, kemudian menukarnya jika elemen tersebut tidak dalam urutan yang diinginkan. Bubble Sort mengulang proses perbandingan ini berulang kali hingga seluruh elemen tersusun dengan benar. Proses ini dinamakan "bubble" karena elemen yang lebih besar (atau lebih kecil, tergantung pada urutan yang diinginkan) akan "menggelembung" ke posisi yang benar setelah setiap iterasi.

1. Fitur mengurutkan komik yang telah diinputkan berdasarkan judul komik

```
case 1:
    for (int i = 0; i < hitung - 1; i++) {
        for (int j = i + 1; j < hitung; j++) {
            if (strcmp(comics[i].name, comics[j].name) > 0) {
                Comic temp = comics[i];
                comics[i] = comics[j];
                comics[j] = temp;
            }
```

Gambar Fitur mengurutkan komik yang telah diinputkan berdasarkan judul komik

Pada bagian ini, komik diurutkan menggunakan metode **bubble short** berdasarkan nama komik secara urutan alfabet menggunakan fungsi **strcmp** dimana fungsi ini membandingkan dua string dan menentukan urutan abjad. Jika hasil perbandingan positif, maka elemen-elemen akan ditukar untuk memastikan elemen dengan nama yang lebih kecil berada di urutan sebelumnya. Sorting pada bagian ini berguna untuk memudahkan pengguna mencari komik yang diinginkan saat nama komik sudah terorganisasi dengan baik.

2. Fitur mengurutkan komik yang telah diinputkan berdasarkan nama komikus

```
126         case 2:
127             for (int i = 0; i < hitung - 1; i++) {
128                 for (int j = i + 1; j < hitung; j++) {
129                     if (strcmp(comics[i].komikus, comics[j].komikus) > 0) {
130                         Comic temp = comics[i];
131                         comics[i] = comics[j];
132                         comics[j] = temp;
```

Gambar mengurutkan komik yang telah diinputkan berdasarkan nama komikus

Pada bagian ini mengurutkan daftar komik berdasarkan nama komikus secara alfabetis. Prosesnya mirip dengan sorting nama komik, Fungsi strcmp membandingkan nama komikus dari dua komik, dan jika hasilnya lebih besar dari 0, maka urutan kedua komik tersebut perlu ditukar.. Dengan mengurutkan berdasarkan nama komikus, pengguna dapat melihat semua komik yang ditulis oleh komikus tertentu dalam urutan yang rapi.

3. Fitur mengurutkan komik yang telah diinputkan berdasarkan popularitas komik

```
142         case 3:
143             for (int i = 0; i < hitung - 1; i++) {
144                 for (int j = i + 1; j < hitung; j++) {
145                     if (comics[i].populer < comics[j].populer) {
146                         Comic temp = comics[i];
147                         comics[i] = comics[j];
148                         comics[j] = temp;
149                     }
```

Gambar mengurutkan komik yang telah diinputkan berdasarkan popularitas komik

Sorting dilakukan berdasarkan nilai popularitas dalam urutan menurun, artinya komik dengan popularitas tertinggi akan ditampilkan lebih dahulu. Nilai popularitas dibandingkan sebagai integer, jika comics[i].populer lebih kecil dari comics[j].populer, maka keduanya ditukar, hal ini untuk memastikan komik dengan popularitas lebih tinggi berada di atas.

4. Fitur mengurutkan komik yang telah diinputkan berdasarkan rating komik

```
158         case 4:
159             for (int i = 0; i < hitung - 1; i++) {
160                 for (int j = i + 1; j < hitung; j++) {
161                     if (comics[i].rating < comics[j].rating) {
162                         Comic temp = comics[i];
163                         comics[i] = comics[j];
164                         comics[j] = temp;
165                     }
166                 }
167             }
```

Gambar Fitur mengurutkan komik yang telah diinputkan berdasarkan rating komik

Pada bagian ini komik diurutkan berdasarkan rating dalam urutan menurun, sehingga komik dengan rating tertinggi akan muncul di bagian atas daftar. Proses perbandingan dilakukan dengan tipe data float,jika comics[i].rating lebih rendah daripada comics[j].rating maka komik-komik tersebut ditukar,hal ini untuk memastikan komik dengan rating lebih tinggi muncul di atas. Sorting ini berguna bagi user yang yang ingin mencari komik dengan kualitas terbaik berdasarkan penilaian rating yang tinggi.

5. Fitur mengurutkan komik yang telah diinputkan berdasarkan tahun komik

```
174         case 5:
175             for (int i = 0; i < hitung - 1; i++) {
176                 for (int j = i + 1; j < hitung; j++) {
177                     if (comics[i].tahun > comics[j].tahun) {
178                         Comic temp = comics[i];
179                         comics[i] = comics[j];
180                         comics[j] = temp;
181                     }
182                 }
```

Gambar mengurutkan komik yang telah diinputkan berdasarkan tahun komik

Sorting dilakukan berdasarkan tahun rilis komik dalam urutan menaik, menampilkan komik yang dirilis lebih dahulu di bagian atas. Perbandingan dilakukan menggunakan integer pada atribut tahun. Jika comics[i].tahun lebih besar dari comics[j].tahun, maka kedua komik ini ditukar, memastikan komik yang lebih lama ditampilkan terlebih dahulu. Urutan ini membantu pengguna melihat kronologi rilis komik.

D. MATERI STRING

string adalah penggunaan data char dengan cara berkelompok. Bisa kita sebut juga dengan array of char karena string merupakan array dari type data char. Karena string merupakan array dari char, maka pendeklarasiannya sama dengan mendeklarasikan array dari char Bahasa pemrograman C memang tidak memiliki tipe data string, tapi kita tetap bisa membuat string dengan tipe data char. Berikut penggunaan string pada program kami

- **Deklarasi String dalam Struktur Comic**

```
typedef struct {
    char name[MAX_NAME_LENGTH];
    char komikus[MAX_KOMIKUS_LENGTH];
    int populer;
    float rating;
    int tahun;
} Comic;
```

Gambar Deklarasi String

Di dalam struktur Comic, terdapat dua anggota yang merupakan string char name[MAX_NAME_LENGTH] Ini adalah array karakter yang digunakan untuk menyimpan nama komik. MAX_NAME_LENGTH diatur ke 100, yang berarti nama komik dapat memiliki panjang maksimum 99 karakter ditambah satu untuk karakter null yang menandakan akhir string. char

komikus[MAX_KOMIKUS_LENGTH] Ini adalah array karakter yang digunakan untuk menyimpan nama komikus dengan cara yang sama seperti name.

- **Membaca Input String**

```
printf("masukkan nama komik: ");
getchar();
fgets(komik_baru.name, MAX_NAME_LENGTH, stdin);
komik_baru.name[strcspn(komik_baru.name, "\n")] = '\0';

printf("masukkan nama komikus: ");
fgets(komik_baru.komikus, MAX_KOMIKUS_LENGTH, stdin);
komik_baru.komikus[strcspn(komik_baru.komikus, "\n")] = '\0';
```

Gambar Membaca Input String

Di dalam fungsi tambah komik , string digunakan untuk membaca input dari pengguna fgets(): Fungsi ini digunakan untuk membaca string dari input standar biasanya keyboard. Ini lebih aman dibandingkan scanf karena fgets dapat membaca string dengan spasi dan membatasi jumlah karakter yang dibaca. Lalu getchar() Digunakan untuk mengkonsumsi karakter newline yang tersisa di buffer input setelah pemanggilan fungsi sebelumnya, sehingga tidak mengganggu pembacaan string berikutnya. strcspn() Fungsi ini digunakan untuk menemukan posisi karakter newline dalam string yang dibaca dan menggantinya dengan karakter null . Ini penting untuk memastikan bahwa string yang disimpan tidak mengandung karakter newline yang tidak diinginkan.

- **Menampilkan String**

```
for (int i = 0; i < hitung; i++) {
    printf("nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
           comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
}
break;
```

Gambar Menampilkan String

Dalam fungsi tampilkan_komik(), string digunakan untuk menampilkan informasi komik %s: Format ini digunakan dalam printf() untuk mencetak string. Di sini, comics[i].name dan comics[i].komikus adalah string yang dicetak ke layar, menampilkan informasi tentang komik yang disimpan dalam array comics.

- **Mencari String**

```
char cari_nama[MAX_NAME_LENGTH];
printf("masukkan nama komik yang ingin dicari: ");
getchar();
fgets(cari_nama, MAX_NAME_LENGTH, stdin);
cari_nama[strcspn(cari_nama, "\n")] = '\0';

int found = 0;
for (int i = 0; i < hitung; i++) {
    if (strcasecmp(comics[i].name, cari_nama) == 0) {
        printf("komik ditemukan: nama: %s | komikus: %s | populer: %d | rating: %.2f | tahun: %d\n",
               comics[i].name, comics[i].komikus, comics[i].populer, comics[i].rating, comics[i].tahun);
        found = 1;
        break;
    }
}
if (!found) {
    printf("komik dengan nama '%s' tidak ditemukan.\n", cari_nama);
}
break;
}
```

Gambar Mencari String

Dalam fungsi cari_komik(), string digunakan untuk mencari komik berdasarkan nama komik atau nama komikus. char cari_nama[MAX_NAME_LENGTH] Ini adalah array karakter yang digunakan untuk menyimpan nama komik yang ingin dicari fgets() Digunakan untuk membaca nama komik yang ingin dicari dari input pengguna strcasecmp() Fungsi ini membandingkan dua string tanpa memperhatikan huruf besar atau kecil. Ini memungkinkan pencarian yang lebih fleksibel, sehingga pengguna tidak perlu

E. MATERI POINTER

Pointer (variabel penunjuk) adalah suatu variabel yang berisi alamat memori dari suatu variabel lain.Dengan kata lain, variabel pointer tidak memegang nilai, tetapi memegang alamat dari variabel lain. Pointer merupakan variabel level rendah yang dapat digunakan untuk menunjuk nilai integer, character, float, double, atau single, dan bahkan tipe-tipe data lain yang didukung oleh bahasa C.Karena pointer tidak memegang nilai tetapi memegang alamat, maka dalam pointer terdapat dua bagian, yaitu pointer itu sendiri yang memegang alamat dan alamat tersebut menunjuk ke suatu nilai. Berikut penggunaan pointer pada program aplikasi komik

- **Penggunaan Pointer dalam Pengelolaan File**

```
file = fopen("komik.txt", "w");
```

Gambar Penggunaan Pointer dalam Pengelolaan File

Salah satu contoh penggunaan pointer dalam kode ini adalah pada deklarasi dan penggunaan pointer file Di sini, file adalah pointer yang digunakan untuk merujuk ke objek file yang dibuka. Fungsi fopen() mengembalikan pointer ke file yang dibuka, yang kemudian digunakan untuk melakukan operasi baca atau tulis. Jika ,file tidak dapat dibuka, pointer file akan bernilai NULL, dan program memeriksa kondisi ini untuk menghindari kesalahan lebih lanjut

- **Penggunaan Pointer dalam Fungsi fscanf dan fprintf**

```
while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d", comics[hitung].name, comics[hitung].komikus,
              &comics[hitung].populer, &comics[hitung].rating, &comics[hitung].tahun) == 5) {
    hitung++;
}
```

Gambar Penggunaan Pointer dalam Fungsi fscanf dan fprintf

Pointer juga digunakan dalam fungsi fscanf() dan fprintf(), yang merupakan fungsi untuk membaca dari dan menulis ke file. Dalam fungsi tambah_komik(), data komik dibaca dari file menggunakan fscanf(). Di sini, operator & digunakan untuk mendapatkan alamat dari variabel populer, rating, dan tahun, yang merupakan anggota dari struktur Comic. Ini memungkinkan fscanf() untuk menulis nilai yang dibaca langsung ke dalam variabel tersebut

- **Penggunaan Pointer dalam Fungsi scanf**

```
while (scanf("%d", &komik_baru.populer) != 1) {
    printf("input invalid! masukkan angka: ");
    while (getchar() != '\n');
}
```

Gambar Penggunaan Pointer dalam Fungsi scanf

Dalam fungsi tambah_komik(), pointer juga digunakan saat membaca input dari pengguna. Di sini, &komik_baru.populer memberikan alamat dari variabel populer dalam struktur komik_baru, sehingga scanf() dapat menyimpan nilai yang dimasukkan pengguna ke dalam variabel tersebut

F. MATERI FILE

Dalam bahasa C, file adalah suatu koleksi data yang disimpan secara permanen di perangkat penyimpanan seperti hard disk atau flash drive. File dapat berisi berbagai jenis informasi, termasuk teks, gambar, audio, video, dan data biner lainnya

1. Menulis Data Komik ke File (fungsi tambah_komik)

```
64      file = fopen("komik.txt", "w");
65      if (!file) {
66          printf("file tidak dapat dibuka untuk menulis.\n");
67          return;
68      }
69
70      for (int i = 0; i < hitung; i++) {
71          fprintf(file, "%s\n%s\n%d\n%.2f\n%d\n", comics[i].name, comics[i ].komikus, comics[i].populer,
72                  comics[i].rating, comics[i].tahun);
73      }
74
75      printf("komik berhasil ditambahkan.\n");
76      fclose(file);
77  }
```

Setelah data komik baru ditambahkan ke dalam array comics[ ], program menulis kembali semua data komik ke dalam file "komik.txt" dengan membuka file menggunakan mode "w" (write). Fungsi fprintf digunakan untuk menulis setiap atribut komik (nama, komikus, popularitas, rating, dan tahun) ke dalam file, dengan format yang sesuai. Dengan menulis kembali data ke file, program memastikan bahwa data komik yang baru saja ditambahkan atau diperbarui tetap tersimpan di dalam file dan dapat diakses pada eksekusi berikutnya.

2. Membaca Data Komik dari File untuk Menampilkan Komik (fungsi tampilkan_komik)

```
82      FILE *file = fopen("komik.txt", "r");
83      if (!file) {
84          printf("File tidak dapat dibuka untuk membaca.\n");
85          return;
86      }
87
88      while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d", comics[hitung].name, comics[hitung].komikus,
89                    &comics[hitung].populer, &comics[hitung].rating, &comics[hitung].tahun) == 5) {
90          hitung++;
91      }
92      fclose(file);
```

Pada bagian ini, file "komik.txt" dibuka dengan mode "r" untuk membaca data komik yang sudah ada. Fungsi fscanf digunakan untuk mengekstrak informasi tentang setiap komik dari file dan menyimpannya ke dalam array comics[ ] yang digunakan oleh program. Ini memungkinkan program untuk menampilkan daftar komik kepada

pengguna, berdasarkan kriteria tertentu yang dipilih oleh pengguna, seperti nama, komikus, popularitas, rating, atau tahun.

BAB 2

KODE PROGRAM

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

#define MAX_COMICS 100
#define MAX_NAME_LENGTH 100
#define MAX_KOMIKUS_LENGTH 100

typedef struct {
    char name[MAX_NAME_LENGTH];
    char komikus[MAX_KOMIKUS_LENGTH];
    int populer;
    float rating;
    int tahun;
} Comic;

Comic comics[MAX_COMICS];
int hitung = 0;

void tambah_komik() {
    Comic komik_baru;
    FILE *file = fopen("komik.txt", "w");
    if (!file) {
        printf("file tidak dapat dibuka untuk membaca.\n");
        return;
    }

    while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d",
comics[hitung].name, comics[hitung].komikus,
                  &comics[hitung].populer, &comics[hitung].rating,
&comics[hitung].tahun) == 5) {
        hitung++;
    }
    fclose(file);

    printf("masukkan nama komik: ");
    getchar();
```

```
    fgets(komik_baru.name, MAX_NAME_LENGTH, stdin);
    komik_baru.name[strcspn(komik_baru.name, "\n")] = '\0';

    printf("masukkan nama komikus: ");
    fgets(komik_baru.komikus, MAX_KOMIKUS_LENGTH, stdin);
    komik_baru.komikus[strcspn(komik_baru.komikus, "\n")] = '\0';

    printf("masukkan popularitas (angka): ");
    while (scanf("%d", &komik_baru.populer) != 1) {
        printf("input invalid! masukkan angka: ");
        while (getchar() != '\n');
    }

    printf("masukkan rating (0-10): ");
    while (scanf("%f", &komik_baru.rating) != 1 || komik_baru.rating < 0
|| komik_baru.rating > 10) {
        printf("input invalid! masukkan rating antara 0-10: ");
        while (getchar() != '\n');
    }

    printf("masukkan tahun: ");
    while (scanf("%d", &komik_baru.tahun) != 1) {
        printf("input invalid! masukkan angka: ");
        while (getchar() != '\n');
    }

    comics[hitung] = komik_baru;
    hitung++;

    file = fopen("komik.txt", "w");
    if (!file) {
        printf("file tidak dapat dibuka untuk menulis.\n");
        return;
    }

    for (int i = 0; i < hitung; i++) {
        fprintf(file, "%s\n%s\n%d\n%.2f\n%d\n", comics[i].name, comics[i
].komikus, comics[i].populer,
                comics[i].rating, comics[i].tahun);
    }
```

```
    printf("komik berhasil ditambahkan.\n");
    fclose(file);
}

void tampilkan_komik() {
    hitung = 0;

    FILE *file = fopen("komik.txt", "r");
    if (!file) {
        printf("File tidak dapat dibuka untuk membaca.\n");
        return;
    }

    while (fscanf(file, " %[^\n]\n%[^\n]\n%d\n%f\n%d",
comics[hitung].name, comics[hitung].komikus,
                  &comics[hitung].populer, &comics[hitung].rating,
&comics[hitung].tahun) == 5) {
        hitung++;
    }
    fclose(file);

    if (hitung == 0) {
        printf("Tidak ada komik untuk ditampilkan.\n");
        return;
    }

    int pilih;
    printf("\ntampilkan komik berdasarkan:\n");
    printf("1. nama komik\n");
    printf("2. nama komikus\n");
    printf("3. popularitas\n");
    printf("4. rating\n");
    printf("5. tahun\n");
    printf("pilih menu: ");
    scanf("%d", &pilih);

    switch (pilih) {
        case 1:
            for (int i = 0; i < hitung - 1; i++) {
```

```
                for (int j = i + 1; j < hitung; j++) {
                    if (strcmp(comics[i].name, comics[j].name) > 0) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("nama: %s | komikus: %s | populer: %d | rating:
%.2f | tahun: %d\n",
                        comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
            }
            break;
        case 2:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (strcmp(comics[i].komikus, comics[j].komikus) > 0)
{
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("komikus: %s | nama: %s | populer: %d | rating:
%.2f | tahun: %d\n",
                        comics[i].komikus, comics[i].name,
comics[i].populer, comics[i].rating, comics[i].tahun);
            }
            break;
        case 3:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].populer < comics[j].populer) {
                        Comic temp = comics[i];
```

```
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("populer: %d | nama: %s | komikus: %s | rating:
%.2f | tahun: %d\n",
                        comics[i].populer, comics[i].name,
comics[i].komikus, comics[i].rating, comics[i].tahun);
            }
            break;
        case 4:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].rating < comics[j].rating) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("rating: %.2f | nama: %s | komikus: %s | populer:
%d | tahun: %d\n",
                        comics[i].rating, comics[i].name,
comics[i].komikus, comics[i].populer, comics[i].tahun);
            }
            break;
        case 5:
            for (int i = 0; i < hitung - 1; i++) {
                for (int j = i + 1; j < hitung; j++) {
                    if (comics[i].tahun > comics[j].tahun) {
                        Comic temp = comics[i];
                        comics[i] = comics[j];
                        comics[j] = temp;
                    }
                }
```

```
            }
            printf("\ndaftar komik:\n");
            for (int i = 0; i < hitung; i++) {
                printf("tahun: %d | nama: %s | komikus: %s | populer: %d |
rating: %.2f\n",
                       comics[i].tahun, comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating);
            }
            break;
        default:
            printf("pilihan tidak valid.\n");
            return;
    }
}

void cari_komik() {
    if (hitung == 0) {
        printf("tidak ada komik untuk dicari.\n");
        return;
    }

    int pilih;
    printf("cari komik berdasarkan:\n");
    printf("1. nama komik\n");
    printf("2. nama komikus\n");
    printf("3. popularitas\n");
    printf("4. rating\n");
    printf("5. tahun\n");
    printf("pilih menu: ");
    scanf("%d", &pilih);

    switch (pilih) {
        case 1: {
            char cari_nama[MAX_NAME_LENGTH];
            printf("masukkan nama komik yang ingin dicari: ");
            getchar();
            fgets(cari_nama, MAX_NAME_LENGTH, stdin);
            cari_nama[strcspn(cari_nama, "\n")] = '\0';

            int found = 0;
```

```
            for (int i = 0; i < hitung; i++) {
                if (strcasecmp(comics[i].name, cari_nama) == 0) {
                    printf("komik ditemukan: nama: %s | komikus: %s |
populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                    break;
                }
            }
            if (!found) {
                printf("komik dengan nama '%s' tidak ditemukan.\n",
cari_nama);
            }
            break;
        }
        case 2: {
            char cari_komikus[MAX_KOMIKUS_LENGTH];
            printf("masukkan nama komikus yang ingin dicari: ");
            getchar();
            fgets(cari_komikus, MAX_KOMIKUS_LENGTH, stdin);
            cari_komikus[strcspn(cari_komikus, "\n")] = '\0';

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (strcasecmp(comics[i].komikus, cari_komikus) == 0) {
                    printf("komik ditemukan: nama: %s | komikus: %s |
populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                }
            }
            if (!found) {
                printf("komikus '%s' tidak ditemukan.\n", cari_komikus);
            }
            break;
        }
        case 3: {
            int cari_populer;
```

```
            printf("masukkan popularitas yang ingin dicari: ");
            scanf("%d", &cari_populer);

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (comics[i].populer == cari_populer) {
                    printf("komik ditemukan: nama: %s | komikus: %s |
populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                }
            }
            if (!found) {
                printf("tidak ada komik dengan popularitas %d.\n",
cari_populer);
            }
            break;
        }
        case 4: {
            float cari_rating;
            printf("masukkan rating yang ingin dicari: ");
            scanf("%f", &cari_rating);

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (comics[i].rating == cari_rating) {
                    printf("komik ditemukan: nama: %s | komikus: %s |
populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                }
            }
            if (!found) {
                 printf("tidak ada komik dengan rating %.2f.\n",
cari_rating);
            }
            break;
        }
```

```
        case 5: {
            int cari_tahun;
            printf("masukkan tahun yang ingin dicari: ");
            scanf("%d", &cari_tahun);

            int found = 0;
            for (int i = 0; i < hitung; i++) {
                if (comics[i].tahun == cari_tahun) {
                    printf("komik ditemukan: nama: %s | komikus: %s |
populer: %d | rating: %.2f | tahun: %d\n",
                           comics[i].name, comics[i].komikus,
comics[i].populer, comics[i].rating, comics[i].tahun);
                    found = 1;
                }
            }
            if (!found) {
                printf("tidak ada komik dengan tahun %d.\n", cari_tahun);
            }
            break;
        }
        default:
            printf("pilihan tidak valid.\n");
            return;
    }
}

int main() {
    int pilih;
    do {
        printf("\nmenu:\n");
        printf("1. tambah komik\n");
        printf("2. tampilkan komik\n");
        printf("3. cari komik\n");
        printf("4. keluar\n");
        printf("pilih menu: ");
        scanf("%d", &pilih);

        switch (pilih) {
            case 1:
                tambah_komik();
```

```
                break;
            case 2:
                tampilkan_komik();
                break;
            case 3:
                cari_komik();
                break;
            case 4:
                printf("keluar program...\n");
                break;
            default:
                printf("pilihan tidak valid.\n");
        }
    } while (pilih != 4);
    return 0;
}
```