BINUS UNIVERSITY ONLINE

People
Innovation
Excellence

Tugas Kelompok ke-1

Week 7

Soal 1: Desain dan Analisis Struct dan Union

Deskripsi Soal:

Anda diminta untuk merancang sebuah sistem informasi sederhana untuk sebuah perpustakaan menggunakan structures dan union. Sistem tersebut harus dapat menyimpan informasi tentang buku dan majalah. Informasi yang perlu disimpan adalah judul, tahun terbit, dan jenis media (buku atau majalah). Gunakan union untuk menyimpan jenis media karena setiap item (buku atau majalah) hanya akan menggunakan satu jenis media dalam satu waktu.

Instruksi Pengerjaan

1. Gambarkan struktur structures dan union yang akan digunakan untuk memodelkan data tersebut.
2. Buatlah sebuah flowchart yang menggambarkan bagaimana data dimasukkan ke dalam struktur ini dan bagaimana jenis media ditentukan dan ditampilkan.
3. Analisis kelebihan menggunakan union dalam kasus ini.
4. Struktur struct harus memiliki field untuk judul, tahun terbit, dan union untuk jenis media.
5. Union harus memiliki dua field: satu untuk buku dan satu untuk majalah.
6. Flowchart harus menunjukkan proses input data, pengaturan dan pengambilan data dari union.
7. Kelebihan menggunakan union termasuk efisiensi penyimpanan memori.

Jawaban:

1. Gambar struktur structures dan union yang akan digunakan adalah pada data tersebut adalah :

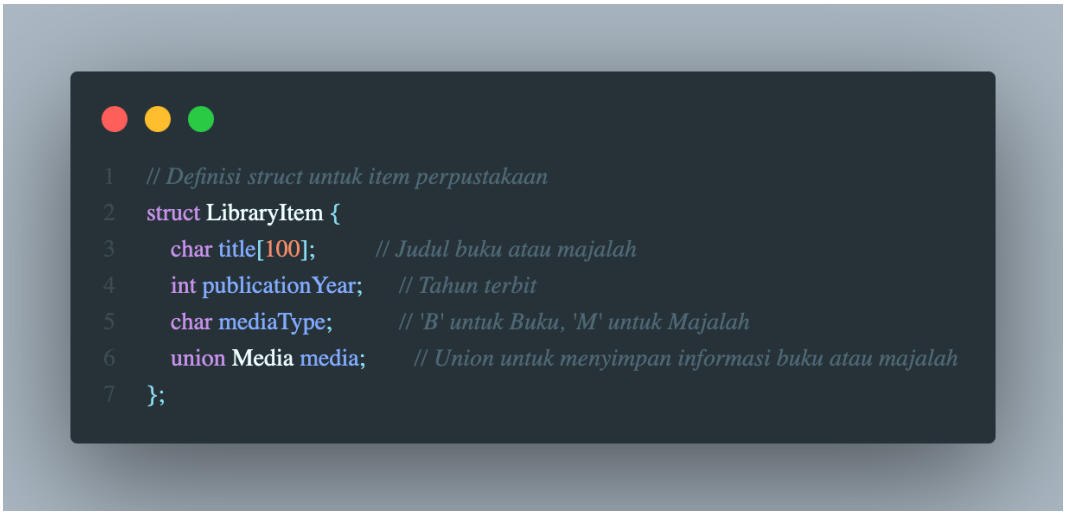
   - Union

Union akan digunakan untuk menyimpan informasi spesifik tentang jenis media (buku atau majalah). Hanya satu tipe data yang akan aktif pada satu waktu. Contoh code yang akan digunakan adalah :

```
union Media
{
    struct
    {
        char author[100]; // Nama penulis untuk buku
    } book;
    struct
    {
        char editor[100]; // Nama editor untuk majalah
    } magazine;
};
```

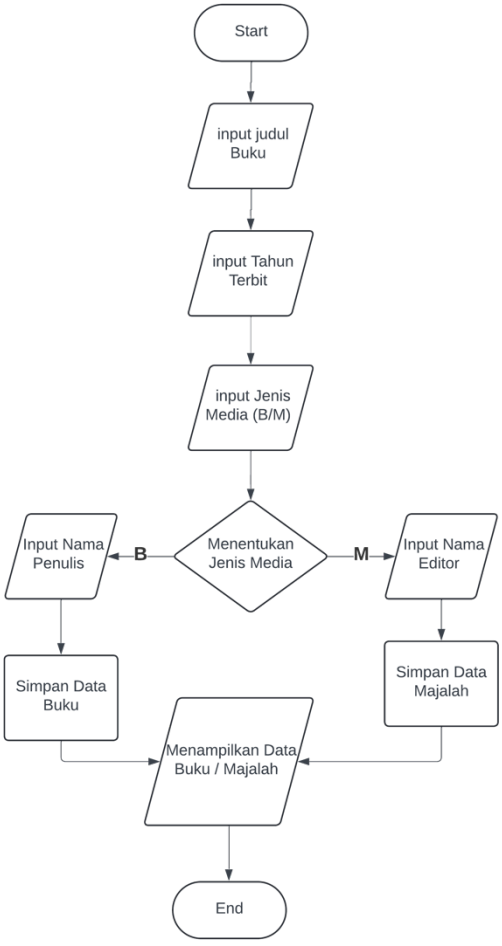
- Struktur

Struktur ini akan menyimpan informasi umum tentang item perpustakaan, termasuk judul, tahun terbit, dan jenis media.



```
// Definisi struct untuk item perpustakaan
struct LibraryItem {
    char title[100];        // Judul buku atau majalah
    int publicationYear;    // Tahun terbit
    char mediaType;         // 'B' untuk Buku, 'M' untuk Majalah
    union Media media;      // Union untuk menyimpan informasi buku atau majalah
};
```

2. Flowchart



3. Kelebihan dalam penggunaan union pada kasus ini adalah :

- Penggunaan Memori yang Optimal: Karena union hanya mengalokasikan ruang untuk satu tipe data pada satu waktu, ini menghemat memori. Dalam kasus ini, jika menyimpan banyak item, penggunaan union memungkinkan penggunaan memori yang lebih efisien dibandingkan dengan struct yang harus mengalokasikan ruang untuk semua field.
- Kesederhanaan dalam Struktur Data Dengan menggunakan union, dapat menyimpan informasi berbeda dalam satu field. Ini menyederhanakan logika pemrograman karena Anda tidak perlu membuat banyak struktur untuk berbagai jenis media.
- Mudah Dikelola dapat dengan mudah mengatur dan mengakses informasi berdasarkan jenis media yang sedang digunakan. Ketika mengetahui bahwa hanya satu jenis media yang digunakan, tidak perlu khawatir tentang field yang tidak terpakai dalam memori.
- Meningkatkan Kejelasan Dengan memiliki satu union untuk menyimpan informasi media, kode menjadi lebih jelas dan mudah dipahami. sehhingga dapat secara eksplisit menunjukkan bahwa informasi penulis atau editor tidak akan digunakan bersamaan.
- Meningkatkan Kejelasan Dengan memiliki satu union untuk menyimpan informasi media, kode menjadi lebih jelas dan mudah dipahami. sehingga dapat secara eksplisit menunjukkan bahwa informasi penulis atau editor tidak akan digunakan bersamaan.

4. Code yang akan digunakan dalam struct adalah sebagai berikut :

```
struct LibraryItem {
    char title[100];          // Judul buku atau majalah
    int publicationYear;      // Tahun terbit
    char mediaType;           // Jenis media: 'B' untuk Buku, 'M' untuk Majalah
    union Media media;        // Union untuk menyimpan informasi buku atau majalah
};
```

5. Code yang akan digunakan dalam Union adalah sebagai berikut :

```
union Media {
    struct {
        char author[100]; // Nama penulis untuk buku
    } book;
    struct {
        char editor[100]; // Nama editor untuk majalah
    } magazine;
};
```

6. Sudah digambarkan pada jawaban no 2
7. Berikut adalah beberapa kelebihan menggunakan union, terutama dalam konteks efisiensi penyimpanan memori dan manfaat lainnya:
   - Efisiensi Penyimpanan Memori : Penggunaan Memori yang Minimal: union hanya mengalokasikan ruang untuk satu anggota (field) pada satu waktu. Dengan kata lain, meskipun ada beberapa field dalam union, ukuran memori yang digunakan hanya sebesar field terbesar. Ini sangat bermanfaat jika Anda memiliki banyak data yang hanya akan digunakan satu per satu.
   - Menghemat Ruang dalam Struktur Data: Ideal untuk Data yang Tidak Dikenal Tipe-nya: Ketika Anda tahu bahwa hanya satu dari beberapa nilai yang akan digunakan pada waktu tertentu,

menggunakan union menghindari pemborosan ruang memori. Ini memungkinkan penyimpanan data yang lebih banyak dalam memori yang terbatas.

- Kesederhanaan dalam Pemrograman : Kejelasan dan Organisasi: Dengan menggunakan union, Anda dapat menyimpan beberapa tipe data dalam satu field. Ini menyederhanakan kode karena Anda tidak perlu membuat beberapa struktur untuk berbagai tipe data yang saling eksklusif.
- Fleksibilitas : Penggunaan yang Fleksibel: Union memungkinkan Anda untuk menggunakan tipe data yang berbeda tanpa perlu mendefinisikan banyak struktur terpisah. Ini mempermudah penanganan berbagai tipe data di dalam program.
- Pengurangan Kompleksitas : Mengurangi Kompleksitas Kode: Dengan memanfaatkan union, Anda mengurangi jumlah kode yang perlu ditulis dan dikelola. Ini mempermudah pemeliharaan dan pengembangan aplikasi.

Soal 2: Analisis Kesalahan Program Sorting

Deskripsi Soal:

Diberikan sebuah cuplikan kode program C untuk sorting array yang mengandung kesalahan. Tugas Anda adalah menganalisis dan menentukan kesalahan dalam algoritma sorting tersebut dan memberikan solusi yang benar.

Cuplikan Kode:

```
#include <stdio.h>

void sort(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n-1; i++) {
        for (j = 0; j < n; j++) {
            if (arr[j] > arr[i]) {
                temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
    }
}
int main() {
    int arr[] = {64, 34, 25, 12, 22};
    int n = sizeof(arr)/sizeof(arr[0]);
    sort(arr, n);
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Instruksi Pengerjaan

1. Identifikasi kesalahan dalam kode sorting yang diberikan.
2. Jelaskan mengapa ini adalah kesalahan.
3. Berikan kode yang sudah diperbaiki.
4. Gambarkan flowchart untuk algoritma sorting yang benar.

Jawaban:

1. Terdapat beberapa kesalahan dalam kode sorting yang diberikan, yaitu:

- Penggunaan j<n pada for loop kedua
- Perbandingan if (arr[j] > arr[i])

2. Penyebab poin-poin tersebut menjadi kesalahan.

- **Penggunaan j<n pada for loop kedua**

  Dalam kode awal, loop kedua (for (j = 0; j < n; j++)) memeriksa setiap elemen dengan semua elemen lainnya, termasuk elemen-elemen yang sudah terurut. Ini sangat tidak efisien dan salah karena elemen yang sudah terurut seharusnya tidak lagi dibandingkan dalam iterasi berikutnya. Untuk memperbaikinya, seharusnya hanya membandingkan elemen-elemen yang belum terurut, yang mana bisa dilakukan dengan membatasi nilai j di loop kedua agar tidak memeriksa elemen yang sudah berada di posisi yang benar (seperti yang dilakukan pada Bubble Sort).

- **Perbandingan if (arr[j] > arr[i])**

  Pada code if (arr[j] > arr[i]), elemen yang dibandingkan adalah elemen yang tidak berdekatan. Ini adalah kesalahan karena dalam algoritma sorting yang seharusnya dibandingkan adalah elemen-elemen yang berdekatan. Contoh, pada Bubble Sort, perbandingan dilakukan antara elemen berdekatan (misalnya arr[j] dan arr[j+1]), dan jika urutannya salah, elemen-elemen tersebut ditukar. Selanjutnya pada Selection Sort, perbandingan dilakukan untuk menemukan elemen terkecil dari bagian array yang belum terurut, kemudian elemen terkecil tersebut ditukar dengan elemen di posisi yang sesuai.

3. Perbaikan code dalam bentuk Bubble Sort

```
#include <stdio.h>

void sort(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n - 1; i++) {
        for (j = 0; j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

int main() {
    int arr[] = {64, 34, 25, 12, 22};
    int n = sizeof(arr) / sizeof(arr[0]);
    sort(arr, n);
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

Perbaikan yang Telah Dilakukan:

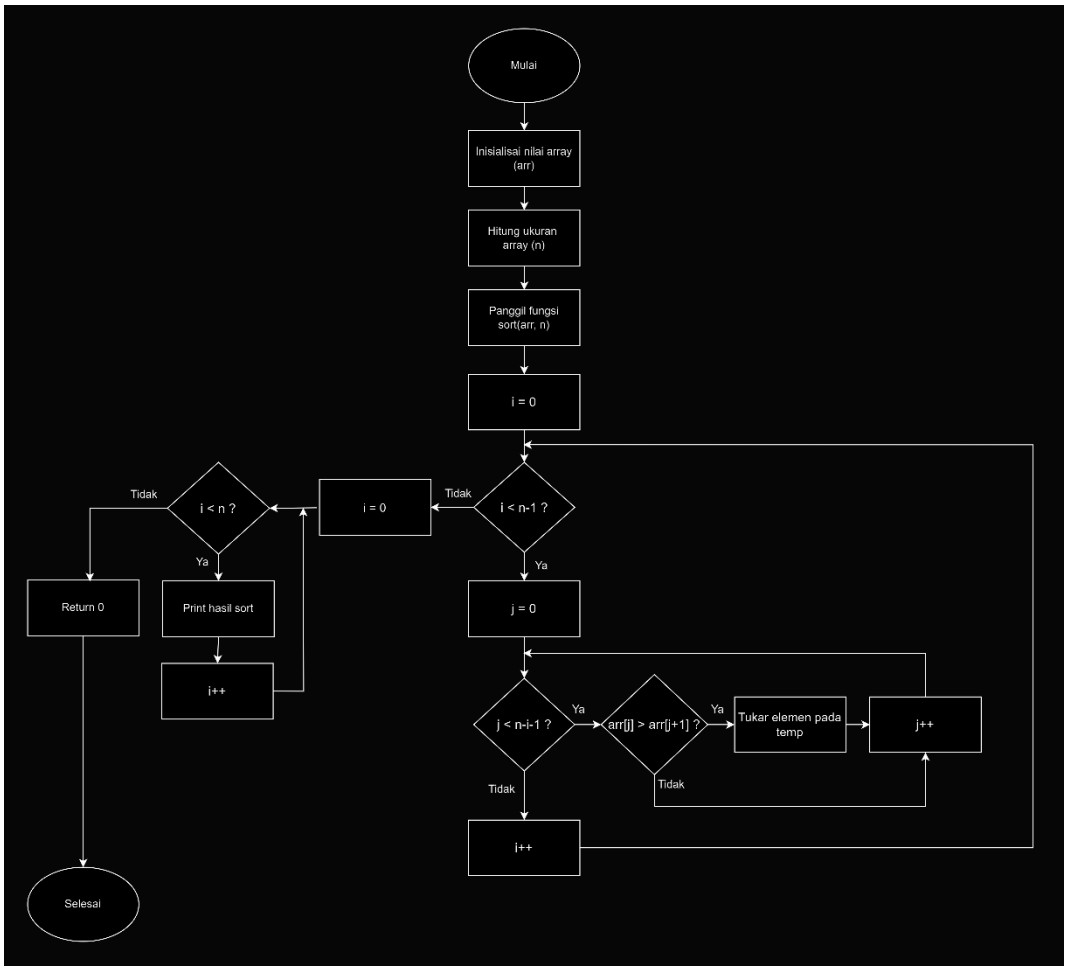
- Perubahan Batas Loop Kedua (j < n):

  Batas j diubah menjadi n - i - 1, yang artinya hanya elemen yang belum terurut yang akan dibandingkan. Setiap kali iterasi i selesai, elemen terbesar "menggelembung" ke akhir array, jadi perbandingan untuk elemen terakhir tidak lagi diperlukan. Perbaikan ini untuk mencegah

perbandingan elemen yang sudah terurut, sehingga lebih efisien dan tidak memeriksa elemen yang sudah berada di posisi yang benar.

- Kesalahan dalam Perbandingan Elemen (arr[j] > arr[i]):
  Perbandingan diubah menjadi if (arr[j] > arr[j + 1]) untuk membandingkan elemen yang berdekatan dalam array. Ini sesuai dengan prinsip dasar Bubble Sort, di mana elemen-elemen yang berdekatan saling dibandingkan dan ditukar jika perlu.

4. Flowchart untuk algoritma sorting yang benar



Dengan penjelasan sebagai berikut:

- **Mulai:** Flowchart dimulai dari proses start atau mulai.
- **Inisialisasi nilai array (arr):** Array yang akan diurutkan diinisialisasi di sini, misalnya arr[] = {64, 34, 25, 12, 22}.

- **Hitung ukuran array (n):** Ukuran array dihitung menggunakan rumus n = sizeof(arr) / sizeof(arr[0]), yang menentukan jumlah elemen dalam array.
- **Panggil fungsi sort(arr, n):** Fungsi bubble sort dipanggil untuk memulai proses pengurutan. Di sinilah langkah-langkah algoritma bubble sort dimulai.
- **Inisialisasi i = 0:** Loop luar dimulai dengan menginisialisasi nilai variabel i ke 0. Ini adalah indeks pertama yang mengontrol loop luar.
- **Apakah i < n - 1?:**
  Jika YA: Artinya, iterasi loop luar masih berlangsung, lanjut ke inisialisasi j = 0 (loop dalam).
  Jika TIDAK: Jika kondisi tidak terpenuhi (artinya, loop luar selesai), lanjut ke print hasil sort.
- **Inisialisasi j = 0:** Jika i < n - 1, nilai j diinisialisasi menjadi 0. Ini adalah indeks untuk loop dalam yang akan mengontrol pertukaran elemen.
- **Apakah j < n - i - 1?:**
  Jika YA: Loop dalam masih berlanjut. Lanjut ke proses pengecekan kondisi elemen array: arr[j] > arr[j + 1].
  Jika TIDAK: Loop dalam selesai untuk nilai i saat ini, kembali ke loop luar untuk meningkatkan nilai i dengan i++.
- **Apakah arr[j] > arr[j + 1]?:**
  Jika YA: Tukar elemen pada arr[j] dengan arr[j + 1]. Lanjut ke j++ untuk memeriksa elemen berikutnya.
  Jika TIDAK: Tidak ada pertukaran elemen. Langsung lanjut ke j++ untuk memeriksa elemen berikutnya.
- **Tukar elemen pada temp:** Jika elemen di arr[j] lebih besar dari elemen di arr[j + 1], maka pertukaran dilakukan menggunakan variabel sementara temp. Setelah pertukaran selesai, lanjutkan dengan j++.
- **j++:** Setelah pertukaran (atau jika tidak ada pertukaran), nilai j ditingkatkan, lalu kembali ke Decision 2 untuk memeriksa apakah j < n - i - 1.
- **i++:** Setelah loop dalam selesai (nilai j mencapai n - i - 1), nilai i ditingkatkan, dan kembali ke Decision 1 untuk memeriksa apakah i < n - 1. Jika YA, loop luar terus berlanjut; jika TIDAK, lanjutkan ke proses berikutnya.

- **Apakah i < n?:**

  Jika YA: Array yang sudah diurutkan dicetak satu per satu pada proses print hasil sort.

  Jika TIDAK: Jika proses pengurutan selesai, maka program akan dilanjutkan ke akhir (proses selesai).
- **Print hasil sort:** Array yang sudah diurutkan dicetak sebagai output, elemen demi elemen.
- **Return 0:** Program selesai dijalankan, dan kontrol dikembalikan (fungsi main() selesai).
- **Selesai:** Proses selesai, program berakhir.

Sumber Referensi:

ChatGPT, pada link https://chatgpt.com/?ref=dotcom, diakses pada 12 Oktober 2024.

C Program to Sort an Array in Ascending Order, https://www.geeksforgeeks.org/c-program-to-sort-an-array-in-ascending-order/, diakses pada 12 Oktober 2024.

Program Algoritma Bubble Sort Bahasa C, https://www.kopicoding.com/bubble-sort-bahasa-c/, diakses pada 13 Oktober 2024.