MODUL 10
ARRAY

PJ

Ferrincia Avril

Contact

230712368@students.uajy.ac.id

MODUL 10
ARRAY

TUJUAN

1. Memahami konsep dasar array.
2. Menggunakan array untuk menyimpan dan mengakses data secara efisien.
3. Mengimplementasikan operasi dasar pada array.
4. Menyelesaikan permasalahan pemrograman menggunakan array.

Array gak sesusah yang kalian bayangkan, percayalah dengan array hidup kalian sebagai anak informatika lebih mudah (di kasus tertentu). So, hilangkan mindset pusing dan malas sebelum belajar modul ini ^^

ILUSTRASI

- Bayangkan sebuah kompleks perumahan dengan beberapa rumah yang berjajar rapi di sepanjang jalan.
- Setiap rumah memiliki nomor rumah yang unik dan setiap rumah dapat ditempati oleh satu keluarga.

Alex	Chaterine	Jeli	Gab	Axel
0	1	2	3	4

- Kompleks perumahan adalah array
- Nomor rumah adalah indeks
- Penghuninya adalah nilai dalam array.
- Kalau kita ingin tahu siapa yang tinggal di rumah nomor 2, kita cukup cek array pada indeks 2, dan jawabannya Jeli. Ingat, array mulai dari angka 0! Alias dari rumah ke 0 sampai rumah ke 4. Totalnya tetap ada 5 rumah kan?
- Nah, itu dia ilustrasi array! 🎉

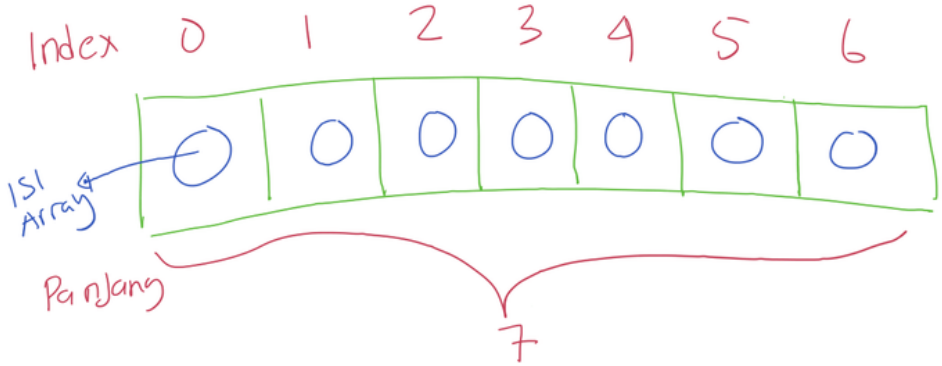
Array itu kumpulan rumah (wadah) yang bisa nyimpan penghuni (data) dalam urutan tertentu. Dan buat cari penghuninya, kamu harus tahu nomor rumahnya (indeks).

Jadi, kalau mau ke rumahnya Gab, berarti ke indeks 3.

Kenapa indeks ketiga kak? bukannya dia urutan nomor 4 yah kalo dari kiri? Sesuai yang sudah aku jelaskan, hitung indeks array mulai dari 0 yaa!!

Gampang kan? Array itu intinya seperti barisan wadah yang menyimpan banyak hal dengan urutan rapi! 🚀

APA ITU ARRAY?



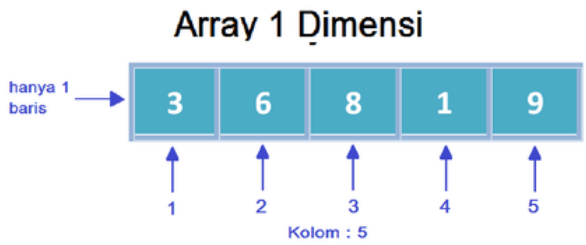
Array/larik/baris adalah kumpulan-kumpulan variabel yang menyimpan data dengan **tipe yang sama** atau data-data yang tersusun secara **linear** dimana di dalamnya terdapat elemen dengan **tipe yang sama**.

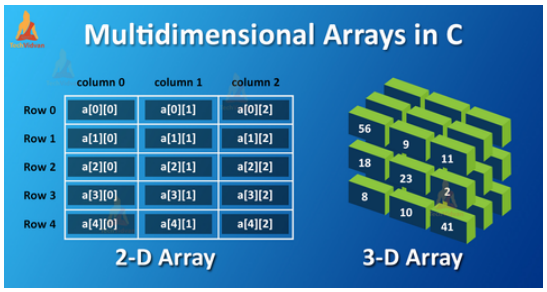
Indeks dalam array menyatakan elemen yang disimpan dan panjang atau length menyatakan total elemen yang tersimpan. Indeks dari elemen array dimulai dari 0, bukan 1.

INGAT!

1. Array dapat menyimpan lebih dari satu data dengan tipe data yang sama.
2. Ukuran dari array harus ditentukan saat pendeklarasiannya, sehingga perlu diketahui terlebih dahulu berapa maksimal data yang ingin ditampung di dalam variabel array.
3. Array bersifat statis, artinya ukuran dari array tidak bisa berubah seiring berjalannya program. Maksudnya statis adalah tidak berubah saat running/eksekusi program. Apa yang tidak berubah? **Ukuran array!** Artinya sekali pesan 5 memori integer (array isi 5 integer), maka kompiler akan menyediakan 5 tempat (memori) untuk 5 integer, sampai program selesai. Tidak peduli 5 memori itu dipakai atau tidak, tetapi kompiler sudah terlanjur mengkampling 5 tempat memori untuk integer.
4. Data pada array diakses menggunakan index, index adalah angka bilangan bulat dan index pada array dimulai dari 0 kemudian 1, 2 , 3 dan seterusnya.

JENIS ARRAY





- **Array satu dimensi**
  - Array satu dimensi merupakan array yang hanya dideklarasikan satu dimensi saja. Array satu dimensi adalah array yang menyimpan satu deret nilai yang bertipe data sama. Contoh penggunaan array satu dimensi yaitu untuk menyimpan daftar nama mahasiswa, daftar npm, daftar harga, dll.

```
int rumah[5] = {10, 20, 30, 40, 50}; // Deretan rumah berisi angka
```

- A**rray multidimensi**
  - Array multidimensi adalah array yang memiliki ukuran dimensi lebih dari satu. Array ini bisa berukuran 2 dimesi, 3 dimensi dan seterusnya. Biasanya array multidimensi yang sering dipakai adalah array yang berukuran 2. Untuk array yang berukuran selebihnya jarang digunakan. Contoh penggunaan array berukuran 2 dimensi pada C adalah matriks atau tabel.

```
int rumah[2][3] = {
    {10, 20, 30}, // Blok A
    {40, 50, 60}  // Blok B
};
```

DEKLARASI ARRAY

```
int angka[5]; // Array dengan 5 elemen bertipe integer
char simbol[10]; //Array dengan 10 elemen bertipe char
string nama[3]; //Array dengan 3 elemen bertipe string
int tabelAngka[5][5]; //Array 2 dimensi dengan 5 baris dan 5 kolom
```

Berikut struktur yang digunakan untuk mendeklarasikan variabel array :

tipeData namaVariabel[ukuran array];

Saat mendeklarasikan variabel array kita harus menentukan tipe datanya seperti saat mendeklarasikan variabel biasa. Namun, pada variabel array kita menambahkan kurung siku yang di dalamnya terdapat bilangan bulat. Bilangan bulat tersebut menandakan ukuran atau banyak data yang ingin ditampung pada variabel array tersebut.

Ingat, array bersifat **statis**!
Sekali dilakukan deklarasi, maka ukuran array **tidak akan bisa berubah** saat program dijalankan.

INISIALISASI ARRAY

```
// Inisialisasi array satu dimensi
int angka1[5] = {1, 2, 3, 4, 5};

// Inisialisasi array dua dimensi (2x3)
int angka2[2][3] = {
    {1, 2, 3},
    {4, 5, 6}
};

// Inisialisasi array tiga dimensi (2x2x2)
int angka3[2][2][2] = {
    {{1, 2}, {3, 4}},
    {{5, 6}, {7, 8}}
};
```

```
for(i = 0; i < 5; i++) {
        angka[i] = i * 10; // Mengisi array dengan kelipatan
10  }
```

Terdapat 2 cara untuk menginisialisasikan array.

1. Langsung assign dengan nilai. Cara ini digunakan ketika kita sudah mengetahui nilai/data yang akan diisi. Sehingga kita dapat mengisi data yang kita inginkan sebelum program dijalankan. Dan data yang diisi harus sesuai dengan tipe datanya.
2. Menggunakan perulangan. Cara seperti ini kita gunakan ketika kita belum mengetahui dengan pasti data apa yang akan diisi ke dalam variabel array atau data berasal dari inputan pengguna. Pada gambar sebelah kanan, kita mengisi variabel array dengan nilai kelipatan 10.

Biasanya kalo bikin program, kita langsung inisialisasikan array lewat perulangan dengan diisi nilai 0. Kenapa? Karena biasanya di soal kita disuruh bikin arraynya dulu, lalu inputan pengguna yang akan jadi data “real”nya. Jadi, biar array kita terbentuk jelas, dan tidak berisi nilai random (alamat) maka kita inisialisasikan dengan 0 ^^.

MENGAKSES ARRAY

- Array satu dimensi

```c
int main() {
    int angka[5] = {10, 20, 30, 40, 50};

    // Mengakses dan menampilkan elemen array
    printf("Elemen pertama: %d\n", angka[0]);
    printf("Elemen ketiga: %d\n", angka[2]);
    printf("Elemen terakhir: %d\n",
angka[4]);
    return 0;
}
```

- Array satu dimensi (dengan perulangan)

```c
int main() {
    int angka[5] = {10, 20, 30, 40, 50};
    int i;

    // Mengakses array dengan perulangan
    printf("Isi array:\n");
    for(i = 0; i < 5; i++) {
        printf("angka[%d] = %d\n", i,
angka[i]);
    }

    return 0;
}
```

- Array dua dimensi

```c
int main() {
    int angka[2][3] = {
        {1, 2, 3},
        {4, 5, 6}
    };

    int i, j;

    // Menampilkan isi array 2D
    printf("Isi array 2D:\n");
    for(i = 0; i < 2; i++) {
        for(j = 0; j < 3; j++) {
            printf("angka[%d][%d] = %d\n", i, j, angka[i]
[j]);   }
    }

    return 0;
}
```

Perlu diperhatikan bahwa saat mengakses data pada variabel array, angka di dalam kurung siku menyatakan index. Sedangkan saat deklarasi variabel array, angka tersebut menyatakan ukuran variabel array.

ARRAY PADA PROSEDUR ATAU FUNGSI

Ketika sebuah array dipassing ke dalam sebuah prosedur atau fungsi, array akan secara otomatis dipassing by reference. Ini berarti bahwa perubahan yang dilakukan pada array di dalam fungsi otomatis juga akan mempengaruhi nilai array di dalam fungsi main().

Hal ini terjadi karena array sudah otomatis merujuk ke alamat pertama elemen, sehingga tidak perlu menggunakan tanda * seperti pada pointer biasa. JADI ARRAY GAPERLU POINTER YAAAA. Gaperlu bingung kalo parameter pakai variabel array.

PRINT ARRAY DENGAN FUNGSI

```c
void printArray(int arr[], int ukuran) {
  int i;
    for (i = 0; i < ukuran; i++) {
        printf("arr[%d] = %d\n", i,
arr[i]);
}

int main() {
    int angka[5] = {10, 20, 30, 40, 50};

    // Memanggil fungsi untuk mencetak array
    printArray(angka, 5);

    return 0;
}
```

CONTOH CODE

```
int main(int argc, char *argv[]) {
    String nama_mhs[5];
    initNama(nama_mhs);
    show(nama_mhs);
}

void initNama(String nama[]) {
    int i;
    for(i = 0; i < 5; i++) {
        strcpy(nama[i], "-");
    }
}

void show(String nama[]) {
    int i;
    for(i = 0; i < 5; i++) {
        printf("%s\n", nama[i]);
    }
}
```

Pada code di atas, kita membuat prosedur yang bernama initNama(String nama[]) . Pada parameter prosedur ata fungsi, ukuran array tidak perlu diisi nilainnya (Kecuali untuk array 2 dimensi atau lebih).

Pada prosedur ini kita menginisialisasi array string yang diisi “-“ dengan menggunakan perulangan for yang dimulai dari 0 sampai 4. Dalam perulangan tersebut, setiap elemen/data array diakses menggunakan index ke-i, Mengapa pakai ke-i? Karena nilai i ini akan dimulai dari 0 s/d 4 seperti yang dituliskan pada perulangan tersebut. Sehingga sesuai dengan nilai index yang dimiliki oleh variabel array tersebut.

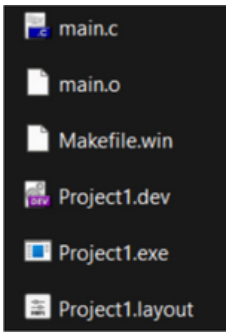
Kemudian pada pemanggilan prosedur di main(), parameternya hanya perlu nama variabel arraynya saja.

REFERENSI BELAJAR

- https://www.geeksforgeeks.org/introduction-to-arrays/
- https://www.petanikode.com/c-array/
- http://www.eecs.qmul.ac.uk/~pc/teaching/introprogramming/week7/exercises7.html
- https://www.hackerrank.com/domains/data-structures?filters%5Bsubdomains%5D%5B%5D=arrays

KETENTUAN GUIDED

- Tidak perlu membuat comment pada guided, buat code-nya saja
- Jangan hanya mengumpulkan file .c saja.



- Format penamaan folder : GD10_X_YYYYY lalu di-zip
- Keterangan:
  - X = Kelas
  - Y = 5 digit NPM terakhir

SEMANGAT MENGERJAKAN ^_^

GUIDED

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define N 5 //mendeklarasikan N = 5, supaya jika ingin mengubah ukuran array
            // tinggal ubah di header, maka semuanya akan otomatis berubah
            // misal ubah jadi N 10, maka ukuran array jadi 10

typedef char string[50];

void initDataFlorist (string namaFlorist[]);
void initDataBunga (string namaBunga[]);
int getEmptyIndex(string array[]);
int cekKosong(string array[]);
int checkNamaUnik(string namaBunga[], string namaB);

int main(int argc, char *argv[]) {
    string namaFlorist[N], namaBunga[N], namaF, namaB, namaBungaBaru;
    int menu, index1, index2, i;
    char yayNay;

    initDataFlorist(namaFlorist); //pemanggilan prosedur inisialisasi,
    initDataBunga(namaBunga);    //kalo ga dipanggil, nanti program crash

    do{
        system("cls");
        printf("\n\t\t--== PROJECT FLORIST ==--");
        printf("\n[1]. CREATE");
        printf("\n[2]. READ");
        printf("\n[3]. UPDATE");
        printf("\n[4]. DELETE");
        printf("\n[0]. EXIT");
        printf("\n>>> "); scanf("%d", &menu);

        switch(menu){
            case 1:
                if((getEmptyIndex(namaFlorist)!=-1)&&(getEmptyIndex(namaBunga)!=-1)){ //jika fungsi
getEmptyIndex tidak return -1
                    index1 = getEmptyIndex(namaFlorist); //nilai return fungsi disimpan dalam var index
                    printf("\n\t CREATE DATA FLORIST");
                    do{
                        printf("\n\t- Masukkan nama florist : "); fflush(stdin); gets(namaF);
                        if(strlen(namaF)==0){
                            printf("\n\t\t[!] NAMA TIDAK BOLEH KOSONG [!]");
                        }
                    }while(strlen(namaF)==0);

                    index2 = getEmptyIndex(namaBunga);
                    do{
                        printf("\n\t- Masukkan nama bunga : "); fflush(stdin); gets(namaB);
                        if(strlen(namaB)==0){
                            printf("\n\t\t[!] NAMA TIDAK BOLEH KOSONG [!]");
                            continue; //melanjutkan perulangan dan melewati program bawahnya
                        }

                        if(checkNamaUnik(namaBunga, namaB)!=-1){
                            printf("\n\t\t[!] NAMA TIDAK BOLEH SAMA [!]");
                        }
                    }while(strlen(namaB)==0||checkNamaUnik(namaBunga, namaB)!=-1);

                    strcpy(namaFlorist[index1], namaF);
                    strcpy(namaBunga[index2], namaB);

                    printf("\n\t[+] BERHASIL CREATE DATA [+]");
                }else{ //jika fungsi getEmptyIndex return -1, alias array penuh
                    printf("\n\t[!] ARRAY FLORIST PENUH [!]");
                }
            break;

            case 2:
                if(cekKosong(namaFlorist)!=-1&&cekKosong(namaBunga)!=-1){ //jika array tdk kosong
                    printf("\n\t\t DATA FLORIST ");
                    for(i=0;i<N;i++){
                        if(strcmp(namaFlorist[i], "-")!=0){
                            printf("\n\t[%d]. Nama florist    : %s", i+1, namaFlorist[i]);
                            printf("\n\t     Nama bunga unik : %s", namaBunga[i]);
                        }
                    }
                }else{
                    printf("\n\t[!] DATA MASIH KOSONG [!]");
                }
            break;
```

GUIDED

```
            case 3:
                if(cekKosong(namaFlorist)!=-1&&cekKosong(namaBunga)!=-1){ //jika array tdk kosong
                    printf("\n\t\t UPDATE DATA BUNGA FLORIST ");
                    printf("\n\tMasukkan nama bunga : "); fflush(stdin); gets(namaB);
                    if(checkNamaUnik(namaBunga, namaB)!=-1){ //cek apakah nama bunga ada di array
                        printf("\n\tMasukkan nama bunga baru: "); fflush(stdin); gets(namaBungaBaru);
                        if(checkNamaUnik(namaBunga, namaBungaBaru)!=-1){ //apakah nama bunga baru
unik
                            printf("\n\t\t[!] NAMA BUNGA HARUS UNIK [!]");
                        }else{
                            index1 = checkNamaUnik(namaBunga, namaB);
                            strcpy(namaBunga[index1], namaBungaBaru);
                            printf("\n\t\t[+] BERHASIL UPDATE DATA [+]");
                        }
                    }else{
                        printf("\n\t[!] BUNGA TIDAK DITEMUKAN [!]");
                    }
                }else{
                    printf("\n\t[!] DATA MASIH KOSONG [!]");
                }
            break;

            case 4:
                if(cekKosong(namaFlorist)!=-1&&cekKosong(namaBunga)!=-1){ //jika array tdk kosong
                    printf("\n\t\t HAPUS DATA FLORIST ");
                    printf("\n\tMasukkan nama florist : "); fflush(stdin); gets(namaF);
                    if(checkNamaUnik(namaFlorist, namaF)!=-1){ //cek apakah nama florist ada di array
                        printf("\n\tApakah anda yakin ingin menghapus florist ini ? [Y/N]");
                        yayNay = getch();
                        if(yayNay=='Y'){
                            index1 = checkNamaUnik(namaFlorist, namaF);
                            strcpy(namaFlorist[index1], "-");
                            strcpy(namaBunga[index1], "-");
                            printf("\n\t[+] DATA BERHASIL DIHAPUS [+]");
                        }else{
                            printf("\n\t\t[!] BATAL MENGHAPUS DATA [!]");
                        }
                    }else{
                        printf("\n\t[!] FLORIST TIDAK DITEMUKAN [!]");
                    }
                }else{
                    printf("\n\t[!] DATA MASIH KOSONG [!]");
                }
            break;

            case 0:
                printf("\n\t NAMA - KELAS - NPM");
                printf("\n\t ... ARRAY EZ ...");
            break;

            default:
                printf("\n\t\t[!] MENU TIDAK ADA [!]");
            break;
        }getch();
    }while(menu!=0);



    return 0;
}
```

GUIDED

```
void initDataFlorist (string namaFlorist[]){
    int i;

    for(i=0;i<N;i++){
        strcpy(namaFlorist[i], "-"); //mengisi array namaFlorist dengan - (inisialisasi)
                                     // jika namaFlorist berisi "-" maka artinya indeks itu kosong
    }
}

void initDataBunga (string namaBunga[]){
    int i;

    for(i=0;i<N;i++){
        strcpy(namaBunga[i], "-"); //sama seperti initDataFlorist
    }
}

int getEmptyIndex(string array[]){
    int i;

    for(i=0;i<N;i++){
        if(strcmp(array[i], "-")==0){
            return i; //jika indeks i kosong, maka akan
                    // return nilai i,sebagai indeks yang
                    // masih kosong
        }
    }

    return -1; //jika array penuh, maka akan return -1
}

int cekKosong(string array[]){
    int i;

    for(i=0;i<N;i++){
        if(strcmp(array[i], "-")!=0){ //jika array i nilainya tidak kosong
            return i;
        }
    }

    return -1; //jika array kosong
}

int checkNamaUnik(string array[], string nama){
    int i;

    for(i=0;i<N;i++){
        if(strcmp(array[i], nama)==0){
            return i;//jika nama sama dgn namaBunga di array, maka return indeks
        }
    }

    return -1; //jika tidak ada nama yg sama, return -1, artinya unik
}
```