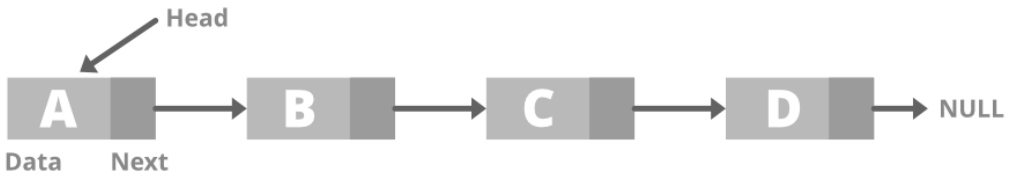
LINKED LIST




```c
struct Anime {
    char judul[20];
    char studio[20];
    int tahun;

    struct Anime *next;
};
```

Pertama deklarasi struct Anime yang akan digunakan sebagai node dalam linked list. Struct ini memiliki tiga member: judul (maksimal 20 karakter), studio (maksimal 20 karakter), tahun (sebuah integer), dan pointer ke node berikutnya dalam linked list.

```c
// Fungsi untuk menambahkan anime di depan linked list
void tambahAnime (struct Anime **head_ref, const char *judul,
const char *studio, int tahun) {

    // Alokasi memori untuk node baru
    struct Anime *new_node = (struct Anime*)malloc(sizeof(struct
Anime));

    // Memasukkan data ke dalam node baru
    strcpy(new_node->judul, judul);
    strcpy(new_node->studio, studio);
    new_node->tahun = tahun;

    // Menjadikan node baru sebagai head
    new_node->next = *head_ref;
    *head_ref = new_node;
```

```
}
```

Fungsi tambahAnime digunakan untuk menambahkan anime baru di depan linked list. Fungsi ini menerima alamat dari head linked list, judul, studio, dan tahun anime sebagai argumen. Prosesnya adalah sebagai berikut:

1. Mengalokasikan memori untuk node baru menggunakan malloc.
2. Memasukkan data anime ke dalam node baru dengan menggunakan fungsi strcpy untuk menduplikat string judul dan studio.
3. Memberi nilai tahun anime.
4. Menjadikan node baru sebagai head linked list.
5. Mengganti alamat head linked list dengan alamat node baru.

```
// Fungsi untuk menampilkan isi linked list
void printList(struct Anime *node) {
    while(node != NULL) {
        printf("%s | %s | %d\n-> ", node->judul, node->studio,
node->tahun);
        node = node->next;
    }
    printf("NULL\n");
}
```

Fungsi printList digunakan untuk menampilkan isi linked list. Fungsi ini menerima head linked list sebagai argumen dan menggunakan loop while untuk mencetak informasi setiap node sampai mencapai node terakhir.

```
int main() {
    // Inisialisasi linked list kosong
    struct Anime *head = NULL;

    // Menambahkan anime di depan linked list
    tambahAnime(&head, "Kimi no Na wa", "Comix Wave", 2016);
    tambahAnime(&head, "One Piece", "Toei Animation", 1998);
    tambahAnime(&head, "Naruto Shippuden", "Pierrot", 2007);
```

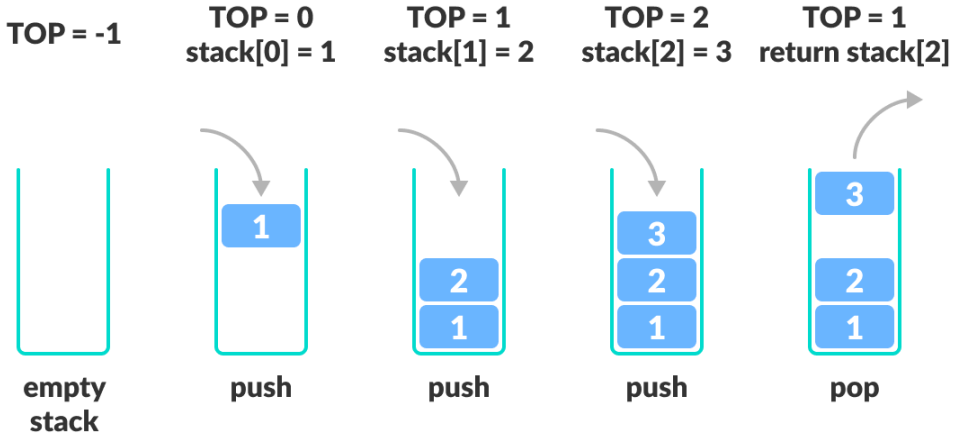
```
    // Menampilkan isi linked list
    printf("Linked list: \n");
    printList(head);

    return 0;
}
```

Fungsi main merupakan fungsi utama program. Di dalamnya, dilakukan inisialisasi linked list anime kosong, menambahkan beberapa anime menggunakan fungsi tambahAnime, dan menampilkan isi linked list menggunakan fungsi printList.

Hasil output:

```
Linked list:
Naruto Shippuden | Pierrot | 2007
-> One Piece | Toei Animation | 1998
-> Kimi no Na wa | Comix Wave | 2016
-> NULL
```

STACK




```
// Struct untuk stack
struct Stack {
    int items[MAX_SIZE];
    int top;
};
```

Struktur data Stack memiliki dua anggota: array items untuk menyimpan elemen-elemen stack, dan top yang menyimpan indeks elemen teratas pada stack.

```c
// Fungsi untuk cek apakah stack kosong
int kosong (struct Stack *stack) {
    return stack->top == -1;
}

// Fungsi untuk cek apakah stack penuh
int penuh(struct Stack *stack) {
    return stack->top == MAX_SIZE - 1;
}
```

Fungsi kosong digunakan untuk memeriksa apakah stack kosong (tidak memiliki elemen), sedangkan penuh untuk memeriksa apakah stack penuh (sudah mencapai batas maksimum).

```c
// Fungsi untuk menambah item ke stack
void push(struct Stack *stack, int nilai) {
    if(penuh(stack)) {
        printf("Stack penuh!\n");
    } else {
        stack->top++;
        stack->items[stack->top] = nilai;
    }
}
```

Fungsi push digunakan untuk menambahkan elemen baru ke dalam stack. Jika stack sudah penuh, akan diberikan pesan "Stack penuh!" agar tidak terjadi overflow. Jika tidak, elemen baru akan ditambahkan dan indeks top diperbarui.

```c
// Fungsi untuk menghapus elemen dari stack
int pop(struct Stack *stack) {
    int nilai;

    if(kosong(stack)) {
        printf("Stack kosong!\n");
        return -1;
    } else {
        nilai = stack->items[stack->top];
        stack->top--;
        return nilai;
    }
}
```

Fungsi pop digunakan untuk menghapus elemen teratas dari stack. Jika stack kosong, akan diberikan pesan "Stack kosong!" dan nilai kembalian adalah -1. Jika tidak, nilai elemen teratas diambil, indeks top dikurangi satu, dan nilai elemen tersebut dikembalikan.

```c
// Fungsi untuk mengambil item teratas tanpa dihapus
int peek(struct Stack *stack) {
    if(kosong(stack)) {
        printf("Stack kosong!\n");
        return -1;
    } else {
        return stack->items[stack->top];
    }
}
```

Fungsi peek digunakan untuk melihat nilai elemen teratas stack tanpa menghapusnya. Jika stack kosong, akan diberikan pesan "Stack kosong!" dan nilai kembalian adalah -1. Jika tidak, nilai elemen teratas dikembalikan.

Fungsi main merupakan fungsi utama dari program. Pada awalnya, sebuah stack (myStack) dibuat dengan indeks top diatur ke nilai -1 untuk menandakan bahwa stack kosong.

Selanjutnya, beberapa operasi dilakukan:

1. Tiga elemen (10, 20, dan 30) dimasukkan ke dalam stack menggunakan fungsi push.
2. Nilai elemen teratas stack ditampilkan menggunakan fungsi peek.

3. Elemen teratas dihapus dari stack menggunakan fungsi pop, dan nilainya ditampilkan.
4. Nilai elemen teratas stack setelah penghapusan ditampilkan menggunakan fungsi peek.

Hasil output:

```
Elemen teratas: 30
Elemen yang dihapus: 30
Elemen teratas setelah penghapusan: 20
```