TUGAS KELOMPOK 3

MATA KULIAH STRUKTUR DATA

“RESUME MATERI STACK”

(Dosen Pengampu : *Nunu Nugraha, S.Pd., MT*)

Disusun Oleh:

1. Bayu Imantoro (20230810089)
2. Dimas Nurdiansyah (20230810087)
3. Muhamad Hafizh Albar (20230810161)
4. Muhammad Rizal Nurfirdaus (20230810088)
5. Reza Saputra (20230810114)

TEKNIK INFORMATIKA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS KUNINGAN

2024

Resume tentang Stack

Pengertian Stack:

Stack adalah struktur data yang beroperasi berdasarkan prinsip LIFO (Last In First Out). Artinya, elemen yang terakhir dimasukkan ke dalam stack akan menjadi elemen pertama yang diambil. Stack dapat diilustrasikan sebagai tumpukan benda, di mana hanya elemen teratas yang dapat ditambah atau diambil.

Operasi-Operasi pada Stack:

1. Push: Menambah item ke bagian atas stack.
2. Pop: Mengambil item dari bagian atas stack.
3. Clear: Mengosongkan stack.
4. IsEmpty: Mengecek apakah stack kosong.
5. IsFull: Mengecek apakah stack penuh.

Implementasi Stack dengan Array:

1. Deklarasi dan Inisialisasi:
   - Mendefinisikan stack menggunakan `struct`.
   - Menetapkan konstanta `MAX_STACK` untuk menyimpan kapasitas maksimum stack.
   - Menggunakan array untuk menyimpan elemen stack.
   - Menambahkan variabel `top` untuk menunjukkan elemen teratas stack, diinisialisasi dengan -1.

```
#define MAX_STACK 10

typedef struct STACK {
    int top;
    char data[MAX_STACK][10];
} STACK;

STACK tumpuk;
```

2. Fungsi IsFull dan IsEmpty:
   - IsFull: Mengembalikan `true` jika `top` sudah mencapai `MAX_STACK - 1`.

- IsEmpty: Mengembalikan `true` jika `top` adalah -1.

```
bool IsFull() {
    return tumpuk.top == MAX_STACK - 1;
}

bool IsEmpty() {
    return tumpuk.top == -1;
}
```

3. Fungsi Push dan Pop:

- Push: Menambah elemen ke stack, mengincrement nilai `top` terlebih dahulu.
- Pop: Mengambil elemen teratas stack, mengurangi nilai `top` setelahnya.

```
void Push(char item) {
    if (!IsFull()) {
        tumpuk.top++;
        strcpy(tumpuk.data[tumpuk.top], item);
    } else {
        printf("Stack penuh!\n");
    }
}

char Pop() {
    if (!IsEmpty()) {
        char item[10];
        strcpy(item, tumpuk.data[tumpuk.top]);
        tumpuk.top--;
        return item;
    } else {
        printf("Stack kosong!\n");
        return '\0';
```

```
    }
  }
```

4. Fungsi Clear dan Print:

- Clear: Mengosongkan stack dengan mengatur `top` ke -1.
- Print: Menampilkan elemen-elemen stack dari atas ke bawah.

```
  void Clear() {
    tumpuk.top = -1;
  }

  void Print() {
    for (int i = tumpuk.top; i >= 0; i--) {
      printf("%s\n", tumpuk.data[i]);
    }
  }
```

Contoh Implementasi dengan Program Pembalikan Kalimat:

Program berikut menggunakan stack untuk membalikkan sebuah kalimat yang dimasukkan pengguna:

```
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>

#define MAXSTACK 200

struct STACK {
   int top;
   char data[MAXSTACK];
};
```

```
STACK stackbaru;

void inisialisasi() {
    stackbaru.top = -1;
}

bool isfull() {
    return stackbaru.top == MAXSTACK - 1;
}

bool isempty() {
    return stackbaru.top == -1;
}

void push(char dta) {
    if (!isfull()) {
        stackbaru.top++;
        stackbaru.data[stackbaru.top] = dta;
    } else {
        puts("Maaf Stack penuh");
    }
}

void pop() {
    while (!isempty()) {
        std::cout << stackbaru.data[stackbaru.top];
        stackbaru.top--;
    }
}

void print() {
    for (int i = 0; i <= stackbaru.top; i++) {
```

```
        std::cout << stackbaru.data[i];
    }
}

int main() {
    char kata[200];

    printf("--- PROGRAM PEMBALIK KALIMAT --- \n\n");
    printf("Masukkan kalimat yang Anda inginkan: \n");
    gets(kata);

    for (int i = 0; kata[i]; i++)
        push(kata[i]);

    printf("------------------------------------ \n\n");
    print();
    std::cout << " menjadi ";
    pop();
    std::cout << "\n";
    getche();

    return 0;
}
```

Kesimpulan:

Stack adalah struktur data yang sangat berguna dan sering digunakan dalam berbagai aplikasi pemrograman. Prinsip LIFO yang diadopsinya membuatnya ideal untuk pengelolaan data yang membutuhkan akses cepat ke elemen terakhir yang dimasukkan. Implementasi stack dapat dilakukan dengan menggunakan array atau struktur data lainnya, dan operasi-operasinya seperti push, pop, clear, isfull, dan isempty dapat diimplementasikan dengan cara yang efisien.