

**LAPORAN TUGAS
STRUKTUR DATA
MODUL 6**
(Dosen Pengampu : Toni Khalimi S.SI.,M.Kom.)



Disusun oleh : Arie Muhamad Syahrial
(20240810091) TINFC-2024-02

**PRODI TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER UNIVERSITAS
KUNINGAN
2025**

LATIHAN TUGAS

Modifikasi program `m6-2.cpp` menjadi seperti berikut:

1. Stack bisa ditentukan ukurannya secara **dinamis oleh pengguna** saat program dijalankan (bukan ukuran tetap seperti sebelumnya).
2. Jika pengguna melakukan `push` dan **stack dalam kondisi penuh**, maka program harus:
 - o Menampilkan pesan:
"Stack penuh! Apakah akan menghapus elemen terakhir? [y/t]:"
 - o Jika pengguna menjawab `y`, maka elemen paling atas dihapus (`pop`), lalu elemen baru bisa dimasukkan (`push`).
 - o Jika pengguna menjawab `t`, maka program hanya akan **menampilkan isi stack** tanpa melakukan `push`.

```
tugas.cpp > ...
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 struct Stack {
5     int top;
6     int *arr;
7     int capacity;
8 };
9
10 Qodo Gen: Options | Test this function
11 void initStack(struct Stack *s, int size) {
12     s->top = -1;
13     s->capacity = size;
14     s->arr = (int *)malloc(size * sizeof(int));
15 }
16
17 Qodo Gen: Options | Test this function
18 int isEmpty(struct Stack *s) {
19     return s->top == -1;
20 }
21
22 Qodo Gen: Options | Test this function
23 int isFull(struct Stack *s) {
24     return s->top == s->capacity - 1;
25 }
26
27 Qodo Gen: Options | Test this function
28 void display(struct Stack *s) {
29     if (isEmpty(s)) {
30         printf("Stack kosong!\n");
31         return;
32     }
33     printf("Isi stack: ");
34     for (int i = 0; i <= s->top; i++) {
35         printf("%d ", s->arr[i]);
36     }
37     printf("\n");
38 }
39
40 Qodo Gen: Options | Test this function
41 void push(struct Stack *s, int value) {
42     if (isFull(s)) {
43         printf("Stack penuh! Apakah akan menghapus elemen terakhir? [y/t]: ");
44         char jawaban;
45         scanf(" %c", &jawaban);
46         if (jawaban == 'y' || jawaban == 'Y') {
47             printf("Menghapus elemen: %d\n", s->arr[s->top]);
48             s->top--;
49         } else {
50             display(s);
51             return;
52         }
53     }
54     s->arr[++(s->top)] = value;
55     printf("%d dimasukkan ke dalam stack.\n", value);
56 }
```

```
51  }
52
53  Qodo Gen: Options | Test this function
54  int pop(struct Stack *s) {
55      if (isEmpty(s)) {
56          printf("Stack kosong! Tidak ada elemen untuk dihapus.\n");
57          return -1;
58      }
59      return s->arr[(s->top)--];
60  }
61
62  Qodo Gen: Options | Test this function
63  int peek(struct Stack *s) {
64      if (isEmpty(s)) {
65          printf("Stack kosong! Tidak ada elemen untuk dilihat.\n");
66          return -1;
67      }
68      return s->arr[s->top];
69
70  Qodo Gen: Options | Test this function
71  int main() {
72      struct Stack s;
73      int size;
74      printf("Masukkan ukuran stack: ");
75      scanf("%d", &size);
76      initStack(&s, size);
77
78      int choice, value;
79
80      while (1) {
81          printf("\nMenu:\n");
82          printf("1. Push\n");
83          printf("2. Pop\n");
84          printf("3. Peek\n");
85          printf("4. Display\n");
86          printf("5. Exit\n");
87          printf("Pilih operasi: ");
88          scanf("%d", &choice);
89
90          switch (choice) {
91              case 1:
92                  printf("Masukkan nilai: ");
93                  scanf("%d", &value);
94                  push(&s, value);
95                  break;
96              case 2:
97                  printf("Menghapus elemen: %d\n", pop(&s));
98                  break;
99              case 3:
100                 printf("Elemen teratas: %d\n", peek(&s));
101                 break;
102             case 4:
103                 display(&s);
104                 break;
105             case 5:
106                 free(s.arr); // Bebaskan memori
107                 printf("Keluar dari program.\n");
108                 exit(0);
109             default:
110                 printf("Pilihan tidak valid!\n");
111         }
112     }
113 }
```

Kode Asli: m6-2.cpp

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 10 // Ukuran maksimum stack
```

- Menggunakan `#define MAX 10` artinya ukuran stack **bersifat tetap** (fixed size).
- Ini akan menjadi **perbedaan utama** karena **TUGAS meminta stack berukuran dinamis.**

```
struct Stack {
    int top;
    int arr[MAX];
};
```

- Struct `Stack` menyimpan data stack:
 - `top`: penanda posisi elemen paling atas.
 - `arr[MAX]`: array untuk menampung data, **masih berukuran tetap (MAX)**.
- Di versi TUGAS, `arr` akan diganti menjadi pointer dinamis `int* arr` dan ukurannya akan ditentukan pengguna lewat `malloc/new`.

```
void initStack(struct Stack *s) {
    s->top = -1;
}
```

- Inisialisasi awal, `top = -1` artinya stack kosong.

```
int isEmpty(struct Stack *s) {
    return s->top == -1;
}
```

- Mengecek apakah stack kosong.

```
int isFull(struct Stack *s) {
    return s->top == MAX - 1;
}
```

- Mengecek apakah stack penuh berdasarkan nilai `MAX`.
- **Di tugas nanti**, fungsi ini harus menerima **kapasitas stack** sebagai parameter karena `MAX` sudah tidak relevan (diganti ukuran dinamis).

```
void push(struct Stack *s, int value) {
    if (isFull(s)) {
        printf("Stack penuh! Tidak dapat menambahkan elemen.\n");
        return;
    }
    s->arr[++(s->top)] = value;
    printf("%d dimasukkan ke dalam stack.\n", value);
```

```
}
```

- Menambahkan elemen ke stack.
- Jika penuh, hanya mencetak pesan.
- **Di tugas**, ketika penuh:
 - Muncul pertanyaan: "*Apakah akan menghapus elemen terakhir?*"
 - Jika y: pop dilakukan, lalu push.
 - Jika t: tampilkan isi stack.

```
int pop(struct Stack *s) {
    if (isEmpty(s)) {
        printf("Stack kosong! Tidak ada elemen untuk dihapus.\n");
        return -1;
    }
    return s->arr[(s->top)--];
}
```

- Menghapus elemen teratas dan mengembalikannya.
- Aman dari error jika kosong.

```
int peek(struct Stack *s) {
    if (isEmpty(s)) {
        printf("Stack kosong! Tidak ada elemen untuk dilihat.\n");
        return -1;
    }
    return s->arr[s->top];
}
```

- Melihat nilai elemen teratas tanpa menghapusnya.

```
void display(struct Stack *s) {
    if (isEmpty(s)) {
        printf("Stack kosong!\n");
        return;
    }
    printf("Isi stack: ");
    for (int i = 0; i <= s->top; i++) {
        printf("%d ", s->arr[i]);
    }
    printf("\n");
}
```

- Menampilkan semua isi stack.

```
int main() {
    struct Stack s;
    initStack(&s);
    int choice, value;
```

- Membuat objek stack dan memulai program.

```

while (1) {
    printf("\nMenu:\n");
    printf("1. Push\n");
    printf("2. Pop\n");
    printf("3. Peek\n");
    printf("4. Display\n");
    printf("5. Exit\n");
    printf("Pilih operasi: ");
    scanf("%d", &choice);
}

```

- Menu interaktif untuk memilih operasi stack.
-

```

switch (choice) {
    case 1:
        printf("Masukkan nilai: ");
        scanf("%d", &value);
        push(&s, value);
        break;
    case 2:
        printf("Menghapus elemen: %d\n", pop(&s));
        break;
    case 3:
        printf("Elemen teratas: %d\n", peek(&s));
        break;
    case 4:
        display(&s);
        break;
    case 5:
        exit(0);
    default:
        printf("Pilihan tidak valid!\n");
}
return 0;
}

```

- Menjalankan fungsi-fungsi berdasarkan pilihan user.
 - Program akan terus berjalan sampai user memilih opsi keluar.
-

Perbedaan Penting antara m6-2.cpp dan Tugas (G.1)

Aspek	Versi m6-2 .cpp	Versi TUGAS G.1
Ukuran Stack	Tetap, menggunakan #define MAX 10	Dinamis, ditentukan oleh user saat program dijalankan
Array Stack	int arr[MAX]	int* arr (pointer dinamis menggunakan malloc atau new)
Deteksi Penuh	Langsung ditolak	Diberi opsi: Hapus elemen terakhir [y/t] atau tampilkan isi

Aspek	Versi m6-2 .cpp	Versi TUGAS G .1
Interaksi saat penuh	Menampilkan pesan "Stack penuh" saja	Menampilkan pertanyaan, menunggu input user, melakukan aksi

```
Masukkan ukuran stack: 2

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 1
Masukkan nilai: 12
12 dimasukkan ke dalam stack.

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 1
Masukkan nilai: 33
33 dimasukkan ke dalam stack.

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 3
Elemen teratas: 33

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 4
Isi stack: 12 33

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 2
Menghapus elemen: 33

Menu:
1. Push
2. Pop
3. Peek
4. Display
5. Exit
Pilih operasi: 5
Keluar dari program.
PS C:\Users\Hype\Documents\struktur data\modul6> █
```