# LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

# STACK


**AHMAD SOFYAN BADAWI**

**244107020073**

**KELAS TI-1B**

**PRODI D-IV TEKNIK INFORMATIKA**

**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2025**

## Percobaan 1:

### A. Class Mahasiswa04

1. Buat folder baru bernama Jobsheet9 di dalam repository Praktikum ASD. Buat file baru, beri nama Mahasiswa04.java
2. Lengkapi class Mahasiswa dengan atribut yang telah digambarkan di dalam class diagram Mahasiswa, yang terdiri dari atribut nama, nim, kelas, dan nilai

```
public class Mahasiswa04 {
    String nim, nama, kelas;
    int nilai;
```

3. Tambahkan konstruktor berparameter pada class Mahasiswa sesuai dengan class diagram Mahasiswa. Berikan nilai default nilai = -1 sebagai nilai awal ketika tugas belum dinilai

```
    Mahasiswa04(String nama, String nim, String kelas) {
        this.nama = nama;
        this.nim = nim;
        this.kelas = kelas;
        nilai = -1;
    }
```

4. Tambahkan method tugasDinilai() yang digunakan untuk mengeset nilai ketika dilakukan penilaian tugas mahasiswa

```
    void tugasDinilai(int nilai) {
        this.nilai = nilai;
    }
}
```

### B. Class StackTugasMahasiswa04

1. Setelah membuat class Mahasiswa04, selanjutnya perlu dibuat class StackTugasMahasiswa04.java sebagai tempat untuk mengelola tumpukan tugas. Class StackTugasMahasiswa merupakan penerapan dari struktur data Stack
2. Lengkapi class StackTugasMahasiswa dengan atribut yang telah digambarkan di dalam class diagram StackTugasMahasiswa, yang terdiri dari atribut stack, size, dan top

```
public class StackTugasMahasiswa04 {
    Mahasiswa04[] stack;
    int size, top;
```

3. Tambahkan konstruktor berparameter pada class StackTugasMahasiswa untuk melakukan inisialisasi kapasitas maksimum data tugas mahasiswa yang dapat disimpan di dalam Stack, serta mengeset indeks awal dari pointer top

```
StackTugasMahasiswa04(int size) {
    this.size = size;
    stack = new Mahasiswa04[size];
    top = -1;
}
```

4. Selanjutnya, buat method isFull bertipe boolean untuk mengecek apakah tumpukan tugas mahasiswa sudah terisi penuh sesuai kapasitas

```
public boolean IsFull(){
    if (top == size - 1) {
        return true;
    } else {
        return false;
    }
}
```

5. Pada class StackTugasMahasiswa, buat method isEmpty bertipe boolean untuk mengecek apakah tumpukan tugas masih kosong

```
public boolean IsEmpty() {
    if (top == -1) {
        return true;
    } else {
        return false;
    }
}
```

6. Untuk dapat menambahkan berkas tugas ke dalam tumpukan Stack, maka buat method push. Method ini menerima parameter mhs yang berupa object dari class Mahasiswa

```
public void push(Mahasiswa04 mhs) {
    if (!IsFull()) {
        top++;
        stack[top] = mhs;
    } else {
        System.out.println(x:"Isi Stack Penuh! Tidak bisa menambahkan tugas lagi");
    }
}
```

7. Penilaian tugas mahasiswa yang dilakukan oleh dosen dilakukan dengan menggunakan method pop untuk mengeluarkan tugas yang akan dinilai. Method ini tidak menerima parameter apapun namun mempunyai nilai kembalian berupa object dari class Mahasiswa

```
public Mahasiswa04 pop() {
    if (!IsEmpty()) {
        Mahasiswa04 m = stack[top];
        top--;
        return m;
    } else {
        System.out.println(x:"Stack masih kosong! Tidak ada tugas untuk dinilai");
        return null;
    }
}
```

8. Buat method peek untuk dapat mengecek tumpukan tugas mahasiswa yang berada di posisi paling atas

```
    public Mahasiswa04 peek() {
        if (!IsEmpty()) {
            return stack[top];
        } else {
            System.out.println(x:"Stack masih kosong! Tidak ada tugas yang dikumpulkan");
            return null;
        }
    }
```

9. Tambahkan method print untuk dapat menampilkan semua daftar tugas mahasiswa pada Stack

```
public void print() {
    for (int i = 0; i <= top; i--) {
        System.out.println(stack[i].nama + "\t" + stack[i].nim + "\t" + stack[i].kelas);
    }
    System.out.println(x:"");
}
```

## C. Class Utama

1. Buat file baru, beri nama MahasiswaDemo04.java
2. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi main
3. Di dalam fungsi main, lakukan instansiasi object StackTugasMahasiswa bernama stack dengan nilai parameternya adalah 5.

```
import java.util.Scanner;
public class MahasiswaDemo04 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        StackTugasMahasiswa04 stack = new StackTugasMahasiswa04(size:5);
        int pilihan;
```

4. Deklarasikan Scanner dengan nama variabel scan dan variabel pilih bertipe int 5

```
import java.util.Scanner;
public class MahasiswaDemo04 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        StackTugasMahasiswa04 stack = new StackTugasMahasiswa04(size:5);
        int pilihan;
```

5. Tambahkan menu untuk memfasilitasi pengguna dalam memilih operasi Stack dalam mengelola data tugas mahasiswa menggunakan struktur perulangan do-while

```java
import java.util.Scanner;
public class MahasiswaDemo04 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        StackTugasMahasiswa04 stack = new StackTugasMahasiswa04(size:5);
        int pilihan;
        do {
            System.out.println(x:"\nMenu: ");
            System.out.println(x:"1. Mengumpulkan Tugas");
            System.out.println(x:"2. Menilai Tugas");
            System.out.println(x:"3. Melihat Tugas Teratas");
            System.out.println(x:"4. Melihat Daftar Tugas");
            System.out.print(s:"Masukkan pilihan: ");
            pilihan = sc.nextInt();
            switch (pilihan) {
                case 1:
                    System.out.print(s:"Nama: ");
                    sc.nextLine();
                    String nama = sc.nextLine();
                    System.out.print(s:"NIM: ");
                    String nim = sc.nextLine();
                    System.out.print(s:"Kelas: ");
                    String kelas = sc.nextLine();
                    Mahasiswa04 mhs = new Mahasiswa04(nama, nim, kelas);
                    stack.push(mhs);
                    System.out.printf(format:"Tugas %s berhasil dikumpulkan\n", mhs.nama);
                    break;
                case 2:
                    Mahasiswa04 dinilai = stack.pop();
                    if (dinilai != null) {
                        System.out.println("Menilai tugas dari " + dinilai.nama);
                        System.out.print(s:"Masukkan nilai (0-100): ");
                        int nilai = sc.nextInt();
                        dinilai.tugasDinilai(nilai);
                        System.out.printf(format:"Nilai Tugas %s adalah %d\n", dinilai.nama, nilai);
                    }
                    break;
                case 3:
                    Mahasiswa04 lihat = stack.peek();
                    if (lihat != null) {
                        System.out.println("Tugas terakhir dikumpulkan oleh " + lihat.nama);
                    }
                    break;
                case 4:
                    System.out.println(x:"Daftar semua yang mengumpulkan");
                    System.out.println(x:"Nama\tNIM\tKelas");
                    stack.print();
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid!");
            }

        } while (pilihan >= 1 && pilihan <= 4);
    }
}
```

## D. Pertanyaan

1. Lakukan perbaikan pada kode program, sehingga keluaran yang dihasilkan sama dengan verifikasi hasil percobaan! Bagian mana yang perlu diperbaiki?
   **Jawab:**

```
    public void print() {
        for (int i = top; i >= 0; i--) {
            System.out.println(stack[i].nama + "\t" + stack[i].nim + "\t" + stack[i].kelas);
        }
        System.out.println(x:"");
    }
}
```

2. Berapa banyak data tugas mahasiswa yang dapat ditampung di dalam Stack? Tunjukkan potongan kode programnya!
   **Jawab:** Banyak data tugas yang dapat ditampung yaitu 5, berikut codenya:

```
StackTugasMahasiswa04 stack = new StackTugasMahasiswa04(size:5);
```

3. Mengapa perlu pengecekan kondisi !isFull() pada method push? Kalau kondisi if-else tersebut dihapus, apa dampaknya?
   **Jawab:** Karena jika tidak ada kondisi full maka akan terjadi output StackOverFlow karena data yang dimasukkan melebihi kapasitas yang sudah ditentukan
4. Modifikasi kode program pada class MahasiswaDemo dan StackTugasMahasiswa sehingga pengguna juga dapat melihat mahasiswa yang pertama kali mengumpulkan tugas melalui operasi lihat tugas terbawah!
   **Jawab:**

```
public Mahasiswa04 peekTugasPertama() {
    if (!IsEmpty()) {
        System.out.println(x:"Nama\tNIM\tKelas");
        System.out.println(stack[0].nama + "\t" + stack[0].nim + "\t" + stack[0].kelas);
        return stack[0];
    } else {
        System.out.println(x:"Stack masih kosong! Tidak ada tugas yang dikumpulkan");
        return null;
    }
}
```

```
case 5:
    System.out.println(x:"Mahasiswa yang mengumpulkan pertama: ");
    stack.peekTugasPertama();
    break;
```

5. Tambahkan method untuk dapat menghitung berapa banyak tugas yang sudah dikumpulkan saat ini, serta tambahkan operasi menunya!
   **Jawab:**

```
public void banyakTugas() {
    int banyakTerkumpul = top + 1;
    System.out.println("Banyak yang sudah mengumpulkan tugas: " + banyakTerkumpul);
}
```

```
case 6:
    stack.banyakTugas();
    break;
```

6. Commit dan push kode program ke Github

## Percobaan 2:

1. Buka kembali file StackTugasMahasiswa04.java
2. Tambahkan method konversiDesimalKeBiner dengan menerima parameter kode bertipe int

```
public String konversiDesimalKeBiner(int nilai) {
    StackKonversi04 stack = new StackKonversi04();
    while (nilai > 0) {
        int sisa = nilai % 2;
        stack.push(sisa);
        nilai = nilai / 2;
    }
    String biner = new String();
    while (!stack.isEmpty()) {
        biner += stack.pop();
    }
    return biner;
}
```

   Pada method ini, terdapat penggunaan StackKonversi04 yang merupakan penerapan Stack, sama halnya dengan class StackTugasMahasiswa. Hal ini bertujuan agar Stack untuk mahasiswa berbeda dengan Stack yang digunakan untuk biner karena tipe data yang digunakan berbeda. Oleh karena itu, buat file baru bernama StackKonversi.java
3. Tambahkan empat method yaitu isEmpty, isFull, push, dan pull sebagai operasi utama Stack pada class StackKonversi04

```java
public class StackKonversi04 {
    int[] tumpukanBiner;
    int size, top;

    StackKonversi04() {
        this.size = 32;
        tumpukanBiner = new int[size];
        top = -1;
    }

    public boolean isEmpty() {
        return top == -1;
    }

    public boolean IsFull() {
        return top == size - 1;
    }

    public void push(int data) {
        if (IsFull()) {
            System.out.println(x:"Stack Penuh");
        } else {
            top++;
            tumpukanBiner[top] = data;
        }
    }

    public int pop() {
        if (isEmpty()) {
            System.out.println(x:"Stack Kosong!");
            return -1;
        } else {
            int data = tumpukanBiner[top];
            top--;
            return data;
        }
    }
}
```

4. Agar nilai tugas mahasiswa dikonversi ke dalam bentuk biner setelah dilakukan penilaian, maka tambahkan baris kode program pada method pop di class MahasiswaDemo04

```
ase 2:
   Mahasiswa04 dinilai = stack.pop();
   if (dinilai != null) {
       System.out.println("Menilai tugas dari " + dinilai.nam
       System.out.print(s:"Masukkan nilai (0-100): ");
       int nilai = sc.nextInt();
       dinilai.tugasDinilai(nilai);
       System.out.printf(format:"NIlai Tugas %s adalah %d\n",
       String biner = stack.konversiDesimalKeBiner(nilai);
       System.out.println("Nilai Biner Tugas: " + biner);
   }
```

5. Compile dan run program.
6. Commit dan push kode program ke Github

**Pertanyaan**

1. Jelaskan alur kerja dari method konversiDesimalKeBiner!
   **Jawab:**
   - method konversiDesimalKeBiner menerima parameter berupa variabel nilai bertipe data integer kemudian di dalamnya melakukan instansiasi objek StackKonversi dengan nama variabel stack.
   - Kemudian ada perulangan while dengan melihat value variabel nilai apakah > 0, jika benar maka akan masuk kedalam while tersebut kemudian akan dicari hasil sisa modulo 2 dari value nilai kemudian akan disimpan dalam variabel sisa dan kemudian hasilnya akan dipush dengan menggunakan stack.push(sisa), selanjutnya value nilai dibagi 2 dan hasilnya menjadi value terbaru dari variabel nilai, terus dilakukan sampai value tidak mememuhi kondisi while.
   - Kemudian setelah while tersebut terdapat while kembali dimana disitu dicek apakah stacknya kosong atau tidak jika tidak maka akan masuk kedalam perulangan while tersebut dan kemudian di dalam tersebut terdapat variabel biner yang siap menerima value dari stack yang sudah dikeluarkan dan akan disimpan, terus berlanjut sampai kondisi while tersebut tidak memenuhi.
   - Setelah semua itu method tersebut akan me-return variabel biner

   Contoh:

   Nilai = 10

   Sisa = 0 (di push)

   Nilai terbaru = 5

   Nilai = 5

   Sisa = 1 (di push)

   Nilai terbaru = 2

Nilai = 2

Sisa = 0 (di push)

Nilai terbaru = 1

Nilai = 1

Sisa = 1 (di push)

Nilai terbaru = 0

Nilai = 0 (tidak memenuhi)

Sisa tersebut disimpan dalam variabel stack:

[0,1,0,1]

Kemudian dikeluarkan satu persatu dari paling atas dan simpan variabel Biner maka hasilnya menjadi berikut:

Hasil Biner: [1,0,1,0]

2. Pada method konversiDesimalKeBiner, ubah kondisi perulangan menjadi while (kode != 0), bagaimana hasilnya? Jelaskan alasannya!
   **Jawab:** Jika yang hanya diubah hanya pada whilenya akan error karena tidak ada variabel kode dan method tersebut juga menerimanya parameter nilai, namun jika semua variabel yang diterima dan variabel yang bernama nilai diubah menjadi kode maka tidak akan terjadi apa apa dia hanya mengubah nama variabel menjadi kode.

**Tugas**

**(ada pada git)**