

LAPORAN HASIL PRAKTIKUM  
ALGORITMA DAN STRUKTUR DATA  
JOBSHEET 9

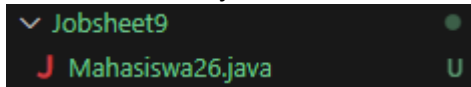


RIFO ANGGI BARBARA DANUARTA  
244107020063  
TI\_1E  
PROGRAM STUDI D\_IV TEKNIK INFORMATIKA  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI MALANG

## Percobaan 1

### A. Class Mahasiswa

1. Buat folder baru bernama Jobsheet9 di dalam repository Praktikum ASD. Buat file baru, beri nama Mahasiswa.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 Mahasiswa26 {  
    String nama;  
    String nim;  
    String 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

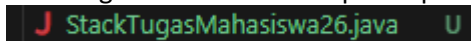
```
Mahasiswa26(String nama, String nim, String kelas) {  
    this.nama = nama;  
    this.nim = nim;  
    this.kelas = kelas;  
    nilai = -1;  
}
```

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

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

### B. Class StackTugasMahasiswa

5. Setelah membuat class Mahasiswa, selanjutnya perlu dibuat class StackTugasMahasiswa.java sebagai tempat untuk mengelola tumpukan tugas. Class StackTugasMahasiswa merupakan penerapan dari struktur data Stack



6. Lengkapi class StackTugasMahasiswa dengan atribut yang telah digambarkan di dalam class diagram StackTugasMahasiswa, yang terdiri dari atribut stack, size, dan top

```
public class StackTugasMahasiswa26 {  
    Mahasiswa26[] stack;  
    int top;  
    int size;  
}
```

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

```
public StackTugasMahasiswa26(int size) {  
    this.size = size;  
    stack = new Mahasiswa26[size];  
    top = -1;  
}
```

8. 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;  
    }  
}
```

9. 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;  
    }  
}
```

10. 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(Mahasiswa26 mhs) {  
    if (!isFull()) {  
        top++;  
        stack[top] = mhs;  
    } else {  
        System.out.println("Stack penuh! Tidak bisa menambah  
tugas lagi.");  
    }  
}
```

11. 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 Mahasiswa26 pop() {  
    if (!isEmpty()) {  
        Mahasiswa26 m = stack[top];  
        top--;  
        return m;  
    } else {  
        System.out.println("Stack kosong! Tidak ada tugas  
untuk dinilai.");  
        return null;  
    }  
}
```

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

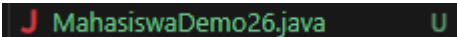
```
public Mahasiswa26 peek() {  
    if (!isEmpty()) {  
        return stack[top];  
    } else {  
        System.out.println("Stack kosong! Tidak ada tugas yang  
dikumpulkan");  
        return null;  
    }  
}
```

13. 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("");  
}  
}
```

### C. Class Utama

14. Buat file baru, beri nama MahasiswaDemo.java

 **MahasiswaDemo26.java**

15. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi main

```
public static void main(String[] args) {
```

16. Di dalam fungsi main, lakukan instansiasi object StackTugasMahasiswa bernama stack dengan nilai parameternya adalah 5.

```
StackTugasMahasiswa26 stack = new  
StackTugasMahasiswa26(5);
```

17. Deklarasikan Scanner dengan nama variabel scan dan variabel pilih bertipe int

```
Scanner scan = new Scanner(System.in);  
int pilih;
```

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

```
do {
    System.out.println("\nMenu:");
    System.out.println("1. Mengumpulkan Tugas");
    System.out.println("2. Menilai Tugas");
    System.out.println("3. Melihat Tugas Teratas");
    System.out.println("4. Melihat Daftar Tugas");
    System.out.print("Pilih: ");
    pilih = scan.nextInt();
    scan.nextLine();
    switch (pilih) {
        case 1:
            System.out.print("Nama: ");
            String nama = scan.nextLine();
            System.out.print("NIM: ");
            String nim = scan.nextLine();
            System.out.print("Kelas: ");
            String kelas = scan.nextLine();
            Mahasiswa26 mhs = new Mahasiswa26(nama, nim,
kelas);

            stack.push(mhs);
            System.out.printf("Tugas %s berhasil
dikumpulkan\n", mhs.nama);
            break;
        case 2:
            Mahasiswa26 dinilai = stack.pop();
            if (dinilai != null) {
                System.out.println("Menilai tugas dari " +
dinilai.nama);

                System.out.print("Masukan nilai (0-100):
");

                int nilai = scan.nextInt();
                dinilai.tugasDinilai(nilai);
                System.out.printf("Nilai Tugas %s adalah
%d\n", dinilai.nama, nilai);
            }
            break;
        case 3:
            Mahasiswa26 lihat = stack.peek();
            if (lihat != null) {
                System.out.println("Tugas terakhir
dikumpulkan oleh " + lihat.nama);
            }
            break;
        case 4:
            System.out.println("Daftar semua tugas");
            System.out.println("Nama\tNIM\tKelas");
            stack.print();
            break;
        default:
            System.out.println("Pilihan tidak valid.");
    }
} while (pilih >= 1 && pilih <= 4);
}
```

### Pertanyaan

1. Lakukan perbaikan pada kode program, sehingga keluaran yang dihasilkan sama dengan verifikasi hasil percobaan! Bagian mana yang perlu diperbaiki?
2. Berapa banyak data tugas mahasiswa yang dapat ditampung di dalam Stack? Tunjukkan potongan kode programnya!
3. Mengapa perlu pengecekan kondisi `!isFull()` pada method `push`? Kalau kondisi `if-else` tersebut dihapus, apa dampaknya?
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!
5. Tambahkan method untuk dapat menghitung berapa banyak tugas yang sudah dikumpulkan saat ini, serta tambahkan operasi `menunya`!
6. Commit dan push kode program ke Github

### Jawaban

1. Setelah saya amati sejauh ini belum ada yang perlu diperbaiki outputnya sesuai Tetapi ada yang kurang menurut saya yaitu menu keluar agar perulangan `do while` dapat berhenti walaupun bisa dihentikan dengan memasukan angka yang tidak ada di menu tetapi lebih rapi lagi apabila ada menu keluar.
2. Data yang dapat disimpan yaitu 5

```
StackTugasMahasiswa26 stack = new
StackTugasMahasiswa26(5);
```

3. Karena diperlukan untuk mencegah stack overflow  
Dan jika `if else` dihapus program akan mencoba menyimpan data dibatas luar array dan dapat menyebabkan program crash.
4. modifikasi yang di class `StackTugasMahasiswa`

```
public Mahasiswa26 bottom() {
    if (!isEmpty()) {
        for (int i = 0; i <= top; i++) {
            if (stack[i] != null) {
                return stack[i];
            }
        }
    }
    return null;
}
```

modifikasi yang di class `MahasiswaDemo`

```
case 5:
    Mahasiswa26 pertama = stack.bottom();
    if (pertama != null) {
        System.out.println("Tugas pertama
dikumpulkan oleh " + pertama.nama);
    }
    break;
```

## 5. modifikasi yang di class StackTugasMahasiswa

```
public int jumlahTugas() {  
    return top + 1;  
}
```

## modifikasi yang di class Mahasiswademmo

```
case 6:  
    System.out.println("Jumlah tugas yang  
dikumpulkan: " + (stack.top + 1));  
    break;
```

## 6.

Xyrf0 Delete Jobsheet9/p c568a31 · 8 minutes ago History		
Name	Last commit message	Last commit date
..		
Mahasiswa26.class	Add files via upload	8 minutes ago
Mahasiswa26.java	Add files via upload	8 minutes ago
MahasiswaDemo26.class	Add files via upload	8 minutes ago
MahasiswaDemo26.java	Add files via upload	8 minutes ago
StackTugasMahasiswa26.class	Add files via upload	8 minutes ago
StackTugasMahasiswa26.java	Add files via upload	8 minutes ago

## Percobaan 2

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

```
public String konversiDesimalKeBiner(int nilai) {  
    StackKonversi stack = new StackKonversi();  
    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 StackKonversi 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 StackKonversi

```
public class StackKoversi26 {
    int[] tumpukanBiner;
    int size;
    int top;

    public StackKoversi26() {
        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("Stack penuh!");
        } else {
            top++;
            tumpukanBiner[top] = data;
        }
    }
    public int pop() {
        if (isEmpty()) {
            System.out.println("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 MahasiswaDemo

```
case 2:
    Mahasiswa26 dinilai = stack.pop();
    if (dinilai != null) {
        System.out.println("Menilai tugas dari " +
dinilai.nama);
        System.out.print("Masukan nilai (0-100):
");
        int nilai = scan.nextInt();
        dinilai.tugasDinilai(nilai);
        System.out.printf("Nilai Tugas %s adalah
%d\n", dinilai.nama, nilai);
        String biner =
stack.konversiDesimalKeBiner(nilai);
        System.out.println("Nilai Biner Tugas: " +
biner);
    }
    break;
```



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

Praktikum-ASD / Jobsheet9 /

**Add file**

**Xyrf** Add files via upload 50fecbc · 3 minutes ago **History**

Name	Last commit message	Last commit date
..		
Mahasiswa26.class	Add files via upload	1 hour ago
Mahasiswa26.java	Add files via upload	1 hour ago
MahasiswaDemo26.class	Add files via upload	3 minutes ago
MahasiswaDemo26.java	Add files via upload	3 minutes ago
StackKonversi26.class	Add files via upload	3 minutes ago
StackKonversi26.java	Add files via upload	3 minutes ago
StackTugasMahasiswa26.class	Add files via upload	3 minutes ago
StackTugasMahasiswa26.java	Add files via upload	3 minutes ago

### Pertanyaan

1. Jelaskan alur kerja dari method konversiDesimalKeBiner!
2. Pada method konversiDesimalKeBiner, ubah kondisi perulangan menjadi while (kode != 0), bagaimana hasilnya? Jelaskan alasannya!

### Jawaban

1. Inisialisasi Stack : Membuat stack kosong untuk menampung digit biner (0/1).  
 Loop 1 : Hitung sisa pembagian (nilai % 2) → simpan di stack. Bagi nilai dengan 2 (nilai = nilai / 2). Ulangi hingga nilai = 0.  
 Loop 2 : Ambil digit dari stack (LIFO: digit terakhir masuk = digit paling signifikan). Gabungkan digit menjadi string biner.  
 Return Hasil : Kembalikan string biner yang telah dibentuk.
2. Hasilnya sama dengan > 0 tetapi jika mengimput bilangan negative loop tetap berjalan dan dapat menyebabkan infinite loop dengan nilai tertentu. Jadi lebih aman menggunakan > 0 daripada != 0.

## Tugas

```
package Jobsheet9;

public class Surat26 {
    String idSurat;
    String namaMahasiswa;
    String kelas;
    char jenisIzin;
    int durasi;

    public Surat26() {
        this.idSurat = "";
        this.namaMahasiswa = "";
        this.kelas = "";
        this.jenisIzin = ' ';
        this.durasi = 0;
    }
    public Surat26(String idSurat, String namaMahasiswa, String kelas,
char jenisIzin, int durasi) {
        this.idSurat = idSurat;
        this.namaMahasiswa = namaMahasiswa;
        this.kelas = kelas;
        this.jenisIzin = jenisIzin;
        this.durasi = durasi;
    }
    public String getNamaMahasiswa() {
        return namaMahasiswa;
    }
}
```

```
package Jobsheet9;

public class StackSurat26 {
    Surat26[] stack;
    int top;
    int size;

    public StackSurat26(int size) {
        this.size = size;
        stack = new Surat26[size];
        top = -1;
    }

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

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

```

    public void push(Surat26 surat) {
        if (!isFull()) {
            top++;
            stack[top] = surat;
        } else {
            System.out.println("Stack penuh! Tidak bisa menambah surat
izin lagi.");
        }
    }

    public Surat26 pop() {
        if (!isEmpty()) {
            Surat26 surat = stack[top];
            top--;
            return surat;
        } else {
            System.out.println("Stack kosong! Tidak ada surat izin untuk
diambil.");
            return null;
        }
    }

    public Surat26 peek() {
        if (!isEmpty()) {
            return stack[top];
        } else {
            return null;
        }
    }

    public Surat26 CariSurat(String namaMahasiswa) {
        for (int i = top; i >= 0; i--) {
            if
(stack[i].getNamaMahasiswa().equalsIgnoreCase(namaMahasiswa)) {
                return stack[i];
            }
        }
        return null;
    }
}

```

```

package Jobsheet9;
import java.util.Scanner;

public class SuratMain26 {
    public static void main(String[] args) {
        StackSurat26 stack = new StackSurat26(10);
        Scanner scan = new Scanner(System.in);
        int pilih;

        do {
            System.out.println("\nMenu:");
            System.out.println("1. Terima Surat Izin");
            System.out.println("2. Proses Surat Izin");
            System.out.println("3. Lihat Surat Izin Teratas");
            System.out.println("4. Cari Surat");
            System.out.println("5. Keluar");
            System.out.println("=====");
            System.out.print("Pilih: ");
            pilih = scan.nextInt();
            scan.nextLine();
            switch (pilih) {
                case 1:
                    System.out.print("ID Surat: ");
                    String idSurat = scan.nextLine();
                    System.out.print("Nama Mahasiswa: ");
                    String namaMahasiswa = scan.nextLine();
                    System.out.print("Kelas: ");
                    String kelas = scan.nextLine();
                    System.out.print("Jenis Izin (C untuk Cuti, S untuk Sakit, L untuk
Lainnya): ");
                    char jenisIzin = scan.nextLine().charAt(0);
                    System.out.print("Durasi (hari): ");
                    int durasi = scan.nextInt();
                    scan.nextLine();
                    Surat26 surat = new Surat26(idSurat, namaMahasiswa, kelas, jenisIzin,
durasi);
                    stack.push(surat);
                    System.out.printf("Surat izin dari %s berhasil diterima.\n",
namaMahasiswa);
                    break;
                case 2:
                    Surat26 diproses = stack.pop();
                    if (diproses != null) {
                        System.out.println("Surat izin dari " + diproses.namaMahasiswa + "
telah diproses.");
                    } else {
                        System.out.println("Tidak ada surat izin untuk diproses.");
                    }
                    break;
                case 3:
                    Surat26 terakhir = stack.peek();
                    if (terakhir != null) {
                        System.out.println("Surat izin terakhir diterima dari " +
terakhir.namaMahasiswa);
                    } else {
                        System.out.println("Tidak ada surat izin yang diterima.");
                    }
                    break;
                case 4:
                    System.out.print("Masukkan nama mahasiswa untuk mencari surat izin:
");
                    String namaCari = scan.nextLine();
                    Surat26 ditemukan = stack.CariSurat(namaCari);
                    if (ditemukan != null) {
                        System.out.println("Surat izin ditemukan: " + ditemukan.idSurat +
" dari " + ditemukan.namaMahasiswa);
                    } else {
                        System.out.println("Surat izin tidak ditemukan untuk nama
tersebut.");
                    }
                    break;
                case 5:
                    System.out.println("Keluar dari program.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid.");
            }
        } while (pilih != 5);
    }
}

```

