

## JOBSHEET IX

### STACK

#### 1. Tujuan Praktikum

Setelah melakukan materi praktikum ini, mahasiswa mampu:

1. Membuat struktur data Stack
2. Menerapkan algoritma Stack ke dalam program Java

#### 2. Praktikum

##### 2.1 Percobaan 1: Mahasiswa Mengumpulkan Tugas

**Waktu Percobaan : 90 Menit**

Sejumlah mahasiswa mengumpulkan berkas tugas di meja dosen secara ditumpuk dengan menerapkan prinsip stack. Dosen melakukan penilaian secara terurut mulai dari berkas tugas teratas. Perhatikan Class Diagram Mahasiswa berikut.

Mahasiswa<NoAbsen>
nim: String nama: String kelas: String nilai: int
Mahasiswa<NoAbsen>() Mahasiswa<NoAbsen>(nim: String, nama: String, kelas: String) tugasDinilai(nilai: int)

Selanjutnya, untuk mengumpulkan berkas tugas, diperlukan class StackTugasMahasiswa yang berperan sebagai Stack tempat menyimpan data tugas mahasiswa. Atribut dan method yang terdapat di dalam class StackTugasMahasiswa merepresentasikan pengolahan data menggunakan struktur Stack. Perhatikan Class Diagram StackTugasMahasiswa berikut.

StackTugasMahasiswa<NoAbsen>
stack: Mahasiswa[] size: int top: int
StackTugasMahasiswa<NoAbsen>(size: int) isFull(): boolean isEmpty(): boolean push(mhs): void pop(): Mahasiswa peek(): Mahasiswa print(): void



---

**Catatan:** Tipe data pada variabel **stack** menyesuaikan dengan data yang akan disimpan di dalam Stack. Pada percobaan ini, data yang akan disimpan merupakan array of object dari **Mahasiswa**, sehingga tipe data yang digunakan adalah **Mahasiswa**.

Berdasarkan dua class diagram tersebut, program menggunakan bahasa

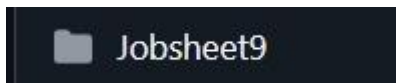
#### A. Class StackTugasMahasiswa

1. Setelah membuat class Mahasiswa, selanjutnya perlu dibuat class **StackTugasMahasiswa<NoAbsen>.java** sebagai tempat untuk mengelola t Java.

### 2.1.2 Langkah-langkah Percobaan

#### A. Class Mahasiswa

2. Buat folder baru bernama **Jobsheet9** di dalam repository **Praktikum ASD**. Buat file baru, beri nama **Mahasiswa<NoAbsen>.java**



3. Lengkapi class **Mahasiswa** dengan atribut yang telah digambarkan di dalam class diagram Mahasiswa, yang terdiri dari atribut **nama**, **nim**, **kelas**, dan **nilai**

```
public Mahasiswa14(String nim, String nama, String kelas) {  
    this.nim = nim;  
    this.nama = nama;  
    this.kelas = kelas;  
    this.nilai = -1;  
}
```

4. Tambahkan konstruktor berparameter pada class Mahasiswa sesuai dengan class diagram Mahasiswa. Berikan nilai default nilai = -1 sebagai nilai awal ketika tugas belum dinilai
5. Tambahkan method **tugasDinilai()** yang digunakan untuk mengeset nilai ketika dilakukan penilaian tugas mahasiswa

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

6. umpukan tugas. Class StackTugasMahasiswa merupakan penerapan dari struktur data Stack
7. Lengkapi class **StackTugasMahasiswa** dengan atribut yang telah digambarkan di dalam class diagram StackTugasMahasiswa, yang terdiri dari atribut **stack**, **size**, dan **top**

```
Mahasiswa14[] stack;  
int size;  
int top;
```

8. 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**

```

public StackTugasMahasiswa14(int size) {
    this.size = size;
    stack = new Mahasiswa14[size];
    top = -1;
}

```

9. 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;
    }
}

```

10. 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;
    }
}

```

11. 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(Mahasiswa14 mhs) {
    if (!isFull()) {
        top++;
        stack[top] = mhs;
    } else {
        System.out.println("Stack penuh! Tidak dapat menambahkan data");
    }
}

```

12. 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 Mahasiswa14 pop() {
    if (!isEmpty()) {
        Mahasiswa14 mhs = stack[top];
        top--;
        return mhs;
    } else {
        System.out.println(x:"Stack kosong! Tidak ada data yang dapat
        return null;
    }
}

```

*Catatan: Apabila diperlukan informasi mengenai data mahasiswa yang diambil, maka tipe kembalian harus berupa object Mahasiswa. Sebaliknya, tipe kembalian void dapat digunakan jika data mahasiswa yang dikeluarkan tidak akan diolah atau digunakan lagi*

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

```

public Mahasiswa14 peek() {
    if (!isEmpty()) {
        return stack[top];
    } else {
        System.out.println(x:"Stack kosong! Tidak ada tugas yang dikump
        return null;
    }
}

```

14. 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 + "\
    }
    System.out.println(x:"");
}

```

## B. Class Utama

15. Buat file baru, beri nama **MahasiswaDemo<NoAbsen>.java**

 MahasiswaDemo14.java

16. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi **main**
17. Di dalam fungsi **main**, lakukan instansiasi object StackTugasMahasiswa bernama **stack** dengan

nilai parameternya adalah 5.

```
public static void main(String[] args) {  
    StackTugasMahasiswa14 stack = new StackTugasMahasiswa14(size:5);  
    Scanner scan = new Scanner(System.in);  
    int pilih;
```

18. Deklarasikan Scanner dengan nama variabel **scan** dan variabel **pilih** bertipe int
19. Tambahkan menu untuk memfasilitasi pengguna dalam memilih operasi Stack dalam mengelola data tugas mahasiswa menggunakan struktur perulangan **do-while**



```

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. Lihat Tugas Teratas");
    System.out.println(x:"4. Melihat Daftar Tugas");
    System.out.print(s:"Pilih: ");
    pilih = scan.nextInt();
    scan.nextLine();

    switch (pilih) {
        case 1:
            System.out.print(s:"Nama: ");
            String nama = scan.nextLine();
            System.out.print(s:"NIM: ");
            String nim = scan.nextLine();
            System.out.print(s:"Kelas: ");
            String kelas = scan.nextLine();
            Mahasiswa mhs = new Mahasiswa(nama, nim, kelas);
            stack.push(mhs);
            System.out.printf(format:"Tugas %s berhasil dikumpulkan\n", mhs.nama);
            break;
        case 2:
            Mahasiswa dinilai = stack.pop();
            if (dinilai != null) {
                System.out.printf(format:"Menilai tugas dari %s\n", dinilai.nama);
                System.out.print(s:"Masukkan nilai (0-100): ");
                int nilai = scan.nextInt();
                dinilai.tugasDinilai(nilai);
                System.out.printf(format:"Nilai Tugas %s adalah %d\n", dinilai.nama, nilai);
            }
            break;
        case 3:
            Mahasiswa lihat = stack.peek();
            if (lihat != null) {
                System.out.printf(format:"Tugas terakhir dikumpulkan oleh %s\n", lihat.nama);
            }
            break;
        case 4:
            System.out.println(x:"Daftar semua tugas:");
            stack.print();
            break;
        default:
            System.out.println(x:"Pilihan tidak valid.");
    }
} while (pilih >= 1 && pilih <= 4);

```

20. Commit dan push kode program ke Github

21. Compile dan run program.

### 2.1.3 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 1

Nama: Dila

NIM: 1001

Kelas: 1A

Tugas Dila berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 1

Nama: Erik

NIM: 1002

Kelas: 1B

Tugas Erik berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 3

Tugas terakhir dikumpulkan oleh Erik

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 1

Nama: Tika

NIM: 1003

Kelas: 1C

Tugas Tika berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 4

Daftar semua tugas

Nama	NIM	Kelas
Tika	1003	1C
Erik	1002	1B
Dila	1001	1A

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 2

Menilai tugas dari Tika

Masukkan nilai (0-100): 87

Nilai Tugas Tika adalah 87

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas

Pilih: 4

Daftar semua tugas

Nama	NIM	Kelas
Erik	1002	1B
Dila	1001	1A

```

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 1
Nama: Dila
NIM: 1001
Kelas: 1A
Tugas Dila berhasil dikumpulkan

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 1
Nama: Erik
NIM: 1002
Kelas: 1B
Tugas Erik berhasil dikumpulkan

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 3
Tugas terakhir dikumpulkan oleh Erik

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 1
Nama: Tika
NIM: 1003
Kelas: 1C
Tugas Tika berhasil dikumpulkan

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 4
Daftar semua tugas:
Dila 1001 1A
Erik 1002 1B
Tika 1003 1C

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 2
Menilai tugas dari Tika
Masukkan nilai (0-100): 87
Nilai Tugas Tika adalah 87

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 4
Daftar semua tugas:
Dila 1001 1A
Erik 1002 1B

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Lihat Tugas Teratas
4. Melihat Daftar Tugas
Pilih: 

```

## 2.1.4 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 TUGAS

- 1.
2. Stack dapat menampung 5 data tugas mahasiswa seperti terlihat dari parameter di konstruktor new StackTugasMahasiswa14(5)
3. karena kondisi isfull adalah untuk mencegah penambahan elemen ketika  $top == size - 1$  yaitu ketika stack sudah penuh, jika dihapus maka program akan langsung menaikkan  $top++$  dan mengakses array di  $stack[top]$  tanpa peduli apakah array masih punya ruang atau tidak.

```
public Mahasiswa14 lihatTugasTerbawah() {
    if (!isEmpty()) {
        return stack[0];
    } else {
        System.out.println(x: "Stack kosong!");
        return null;
    }
}
```

4.

```
System.out.println(x: "4. Melihat Daftar Tugas");
System.out.println(x: "5. lihat Tugas pertama");
System.out.println(x: "6. lihat jmlh tugas");
```

```
case 5:
    Mahasiswa14 bawah = stack.lihatTugasTerbawah();
    if (bawah != null) {
        System.out.printf(format: " pertama dikumpulkan oleh %s\n", bawah.nama);
    }
    break;
case 6:
```

5. `System.out.println(x:"6. lihat jmlh tugas");`

```
break;  
case 6:  
    System.out.printf(format:"Jumlah tugas yang telah dikumpulkan: %d\n", stack.jumlahTugas());  
    break;
```

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

## 2.2 Percobaan 2: Konversi Nilai Tugas ke Biner

**Waktu Percobaan: 60 Menit**

Sampai tahap ini, proses pengelolaan data tugas mahasiswa menggunakan konsep Stack telah berhasil dibuat pada Percobaan 1. Selanjutnya, pada Percobaan 2 ini ditambahkan method baru yang berfungsi untuk mengonversi nilai tugas bertipe int ke dalam bentuk biner setelah tugas tersebut diberi nilai dan dikeluarkan dari Stack.

### 2.2.1 Langkah-langkah Percobaan

1. Buka kembali file **StackTugasMahasiswa<NoAbsen>.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<NoAbsen>.java**

*Catatan: Perlu diingat bahwa pada dasarnya semua class Stack mempunyai operasi (method) yang sama. Hal yang membedakan adalah aktivitas spesifik yang perlu dilakukan, misalnya setelah menambah atau mengeluarkan data.*

3. Tambahkan empat method yaitu **isEmpty**, **isFull**, **push**, dan **pull** sebagai operasi utama Stack pada class **StackKonversi**

```
int[] tumpukanBiner;
int size;
int top;

public StackKonversi() {
    this.size = 32; // asumsi 32 bit
    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 **MahasiswaDemo**

```
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

### 2.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

Menu:

```
1. Mengumpulkan Tugas  
2. Menilai Tugas  
3. Melihat Tugas Teratas  
4. Melihat Daftar Tugas  
Pilih: 2  
Menilai tugas dari Tika  
Masukkan nilai (0-100): 87  
Nilai Tugas Tika adalah 87  
Nilai Biner Tugas: 1010111
```

```
Menu:  
1. Mengumpulkan Tugas  
2. Menilai Tugas  
3. Lihat Tugas Teratas  
4. Melihat Daftar Tugas  
5. lihat Tugas pertama  
6. lihat jmlh tugas  
Pilih: 2  
Menilai tugas dari Tika  
Masukkan nilai (0-100): 87  
Nilai Tugas Tika adalah 87  
Nilai Biner Tugas: 1010111
```

---

### 2.2.3 Pertanyaan

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

#### Jawaban Pertanyaan

1. Dibuat method **konversiDesimalKeBiner** yang menerima input nilai bertipe data int.

Selanjutnya, dibuat objek stack dari kelas **StackKonversi14** yang digunakan untuk menyimpan sisa hasil pembagian nilai dengan 2.

Lalu, selama nilai lebih dari 0, dilakukan perulangan:

Nilai akan dibagi dengan 2 dan sisanya disimpan ke dalam stack menggunakan method **push**.

Nilai kemudian dibagi 2 untuk proses selanjutnya (menggunakan pembagian bilangan bulat)

Setelah seluruh proses pembagian selesai (nilai menjadi 0), dibuat sebuah variabel bertipe **String** bernama **biner** untuk menampung hasil akhir

Selanjutnya, dilakukan perulangan untuk mengambil semua isi stack menggunakan method **pop**, dan tiap hasilnya ditambahkan (+) ke string **biner**

string **biner** yang berisi representasi biner dari nilai desimal dikembalikan sebagai hasil dari method.

2. maka akan terjadi error kompilasi (compilation error) karena variabel **kode** tidak dikenal atau tidak dideklarasikan di dalam method tersebut

### 2.4 Latihan Praktikum

**Waktu : 90 Menit**

Mahasiswa mengajukan surat izin (karena sakit atau keperluan lain) setiap kali tidak mengikuti perkuliahan. Surat terakhir yang masuk akan diproses atau divalidasi lebih dulu oleh admin Prodi. Perhatikan class diagram berikut.

Surat<NoAbsen>
idSurat: String namaMahasiswa: String kelas: String jenisIzin: char durasi: int

Surat<NoAbsen>() Surat<NoAbsen>(idSurat: String, namaMahasiswa: String, kelas: String, jenisIzin: char, durasi: int)
--

Atribut **jenisIzin** digunakan untuk menyimpan keterangan ijin mahasiswa (S: sakit atau I: izin keperluan lain) dan **durasi** untuk menyimpan lama waktu izin.

Berdasarkan class diagram tersebut, implementasikan class **Surat** dan tambahkan class **StackSurat** untuk mengelola data Surat. Pada class yang memuat method main, buat pilihan menu berikut:

1. **Terima Surat Izin** untuk memasukkan data surat
2. **Proses Surat Izin** untuk memproses atau memverifikasi surat
3. **Lihat Surat Izin Terakhir** untuk melihat surat teratas
4. **Cari Surat** untuk mencari ada atau tidaknya surat izin berdasarkan **nama mahasiswa**

JAWABAN TUGAS

//FILE SURATDEMO

```
import java.util.Scanner;

public class SuratDemo14 {
    public static void main(String[] args) {
        StackSurat14 stack14 = new StackSurat14(5);
        Scanner scan14 = new Scanner(System.in);
        int pilih14;
        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 Terakhir");
            System.out.println("4. Cari Surat berdasarkan nama mahasiswa");
            System.out.println("5. Keluar");
            System.out.print("Pilih: ");
            pilih14 = scan14.nextInt();
            scan14.nextLine();
            switch (pilih14) {
```

```

        case 1:

            System.out.print("ID Surat: ");

            String id14 = scan14.nextLine();

            System.out.print("Nama Mahasiswa: ");

            String nama14 = scan14.nextLine();

            System.out.print("Kelas: ");

            String kelas14 = scan14.nextLine();

            System.out.print("Jenis Izin (S/I): ");

            char jenis14 =
scan14.nextLine().toUpperCase().charAt(0);

            System.out.print("Durasi (hari): ");

            int durasi14 = scan14.nextInt();

            scan14.nextLine();

            Surat14 surat14 = new Surat14(id14, nama14, kelas14,
jenis14, durasi14);

            stack14.push(surat14);

            System.out.println("Surat berhasil diterima.");

            break;

        case 2:

            Surat14 proses14 = stack14.pop();

            if (proses14 != null) {

                System.out.println("Surat diproses: " +
proses14);

            }

            break;

        case 3:

            Surat14 lihat14 = stack14.peek();

            if (lihat14 != null) {

                System.out.println("Surat terakhir: " + lihat14);

            }

            break;

```

```

        case 4:

            System.out.print("Masukkan nama mahasiswa yang
dicari: ");

            String cari14 = scan14.nextLine();

            boolean ditemukan14 = stack14.cariSurat(cari14);

            if (ditemukan14) {

                System.out.println("Surat izin ditemukan.");

            } else {

                System.out.println("Surat izin tidak
ditemukan.");

            }

            break;

        case 5:

            System.out.println("Keluar dari program.");

            break;

        default:

            System.out.println("Pilihan tidak valid!");

    }

    } while (pilih14 != 5);

}
}

```

//FILE STACKSURAT

```

public class StackSurat14 {

    Surat14[] stack14;

    int top14;

    int size14;

    public StackSurat14(int size14) {

        this.size14 = size14;

        this.stack14 = new Surat14[size14];

    }

}

```

```
        this.top14 = -1;
    }

    public boolean isFull() {
        return top14 == size14 - 1;
    }

    public boolean isEmpty() {
        return top14 == -1;
    }

    public void push(Surat14 surat14) {
        if (!isFull()) {
            stack14[++top14] = surat14;
        } else {
            System.out.println("Stack penuh! Tidak bisa menambahkan sura");
        }
    }

    public Surat14 pop() {
        if (!isEmpty()) {
            return stack14[top14--];
        } else {
            System.out.println("Stack kosong! Tidak ada surat yang iproses");
            return null;
        }
    }

    public Surat14 peek() {
        if (!isEmpty()) {
```

```

        return stack14[top14];
    } else {
        System.out.println("Stack kosong!");
        return null;
    }
}

public boolean cariSurat(String nama14) {
    for (int i = 0; i <= top14; i++) {
        if (stack14[i].namaMahasiswa.equalsIgnoreCase(nama14)) {
            return true;
        }
    }
    return false;
}
}

```

//FILE SURAT

```

public class Surat14 {
    String idSurat, namaMahasiswa, kelas;
    char jenisIzin;
    int durasi;

    public Surat14(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 toString() {  
    return "ID: " + idSurat + ", Nama: " + namaMahasiswa + ", Kelas:  
" + kelas +  
        ", Jenis Izin: " + jenisIzin + ", Durasi: " + durasi + "  
hari";  
    }  
}
```