

LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN STRUKTUR DATA

JOBSHEET 9



NAMA :MAULIDYAAFRIANI

NIM: 2441070200559


KELAS : 1E

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG

2025

PERCOBAAN 1

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

 Mahasiswa15.java >

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

```
String nama; 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


```
Mahasiswa15(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;  
}
```

A. Class StackTugasMahasiswa

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

 StackTugasMahasiswa15.java >

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

```
Mahasiswa15[] stack;  
int top;  
int size;
```

7. 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 StackTugasMahasiswa15(int size) {  
    this.size = size;  
    stack = new Mahasiswa15[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 {

```

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(Mahasiswa15 mhs) {

        if (!isFull()) {

            top++;

            stack[top] = mhs;

        }else {

            System.out.println("Stack penuh! tidak bisa
menambahkan 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 Mahasiswa15 pop() {
    if (!isEmpty()) {
        Mahasiswa15 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

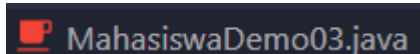
```
} public Mahasiswa15 peek() {  
    if (!isEmpty()) {  
        return stack[top];  
    } else {  
        System.out.println("Stack kosong! tidak ada tugas  
yang dikumpulkan");  
        return null;  
    }  
}
```

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("");  
}
```

B. Class Utama

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



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

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

```
import java.util.Scanner; public class MahasiswaDemo03 {  
  
    public static void main(String[] args) {  
        StackTugasMahasiswa15 stack = new  
StackTugasMahasiswa15(5);  
        Scanner scan = new Scanner(System.in);          int  
pilih;
```

16. Deklarasikan Scanner dengan nama variabel **scan** dan variabel **pilih** bertipe int

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

17. 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.println("0. Keluar");
        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();
                Mahasiswa15 mhs = new Mahasiswa15(nama, nim,
kelas);
                stack.push(mhs);
                System.out.printf("Tugas      %s      berhasil
dikumpulkan\n", mhs.nama);
                break;
            case 2:
                Mahasiswa15 dinilai = stack.pop();
                if (!stack.isEmpty()) {
                    System.out.println("Menilai tugas dari " +
dinilai.nama);
                    System.out.print("Masukkan 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:
                Mahasiswa15 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);
}
}

```

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 1
Nama: litya
NIM: 009
Kelas: 1e
Tugas litya berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 1
Nama: lala
NIM: 789
Kelas: 1q
Tugas lala berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 3
Tugas terakhir dikumpulkan oleh lala

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 1
Nama: yiala
NIM: 6366
Kelas: 1d
Tugas yiala berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 4
Daftar semua tugas:

| Nama | NIM | Kelas |
|-------|------|-------|
| litya | 009 | 1e |
| lala | 789 | 1q |
| yiala | 6366 | 1d |

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 2
Menilai tugas dari yiala
Masukkan nilai (0-100): 89
Nilai Tugas yiala adalah 89

```

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
0. Keluar
Pilih: 4
Daftar semua tugas:
Nama      NIM      Kelas
lidy      009      1e
lala      789      1q

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

```

Pertanyaan

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

Mahasiswa15.java

```

public class Mahasiswa15 {
    String nim, nama, kelas;
    int nilai;
    Mahasiswa15() {
        this.nim = "";
        this.nama = "";
        this.kelas = "";
        this.nilai = -1;
    }
    Mahasiswa15(String nim, String nama, String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.nilai = -1;
    }
    void tugasDinilai(int nilai) {
        this.nilai = nilai;
    }
}

```

StuckTugasMahasiswa15.java

```
public class StackTugasMahasiswa15 {
    Mahasiswa15[] stack;
    int size, top;

    StackTugasMahasiswa15(int size) {
        this.size = size;
        this.stack = new Mahasiswa15[size];
        this.top = -1;
    }

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

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

    void push(Mahasiswa15 mhs) {
        if (!isFull()) {
            stack[++top] = mhs;
        } else {
            System.out.println("Stack penuh. Tidak dapat menambahkan tugas.");
        }
    }

    Mahasiswa15 pop() {
        if (!isEmpty()) {
            return stack[top--];
        } else {
            System.out.println("Stack kosong. Tidak ada tugas yang bisa dinilai.");
            return null;
        }
    }

    Mahasiswa15 peek() {
        if (!isEmpty()) {
            return stack[top];
        } else {
            System.out.println("Stack kosong.");
            return null;
        }
    }
}
```



```

    }

    void print() {
        if (!isEmpty()) {
            for (int i = top; i >= 0; i--) {
                System.out.println(stack[i].nama + "\t" +
stack[i].nim + "\t" + stack[i].kelas);
            }
        } else {
            System.out.println("Stack kosong.");
        }
    }
}

```

MahasiswaDemo15.java

```

import java.util.Scanner;

public class MahasiswaDemo15 {
    public static void main(String[] args) {
        StackTugasMahasiswa15 stack = new
StackTugasMahasiswa15(5);

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

        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.println("0. Keluar");
            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();
        Mahasiswa15 mhs = new Mahasiswa15(nim,
nama, kelas);

        stack.push(mhs);

        System.out.printf("Tugas %s berhasil
dikumpulkan\n", mhs.nama);
        break;
    case 2:
        Mahasiswa15 dinilai = stack.pop();
        if (dinilai != null) {
            System.out.println("Menilai tugas dari
" + dinilai.nama);

            System.out.print("Masukkan 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:
        Mahasiswa15 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;
    case 0:
        System.out.println("Terima kasih.");
        break;
    default:
        System.out.println("Pilihan tidak valid.");
    }
} while (pilih != 0);
}
}

```

3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 4

Daftar semua tugas:

| Nama | NIM | Kelas |
|------|------|-------|
| loi | 3020 | 1e |
| lia | 1001 | 1e |

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 2

Menilai tugas dari loi

Masukkan nilai (0-100): 66

Nilai Tugas loi adalah 66

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 4

Daftar semua tugas:

| Nama | NIM | Kelas |
|------|------|-------|
| loi | 3020 | 1e |
| lia | 1001 | 1e |

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 1

Nama: lia

NIM: 1001

Kelas: 1e

Tugas lia berhasil dikumpulkan

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 3

Tugas terakhir dikumpulkan oleh lia

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
0. Keluar

Pilih: 1

Nama: loi

NIM: 3020

2. Berapa banyak data tugas mahasiswa yang dapat ditampung di dalam Stack? Tunjukkan potongan kode programnya!

Jawab: Ada 5 data

3. Mengapa perlu pengecekan kondisi **!isFull()** pada method **push**? Kalau kondisi if-else tersebut dihapus, apa dampaknya?

Jawab: Kalau stack udah penuh tapi kamu masih maksa masukin data tanpa dicek dulu, program bisa error. Error-nya biasanya **ArrayIndexOutOfBoundsException**, yang artinya kamu nyoba taruh data di tempat yang nggak ada di dalam array. Nah, biar kejadian kayak gitu nggak terjadi, kita harus cek dulu pakai **!isFull()**. Itu fungsinya buat memastikan masih ada ruang kosong sebelum nambahin data ke stack. Kalau dicek dulu, aman. Tapi kalau langsung asal push, ya siap-siap programnya bisa berhenti mendadak. Jadi intinya, pengecekan itu penting banget supaya program tetap jalan lancar dan nggak nabrak batas yang udah 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:

Mahasiswa.java

```
public class Mahasiswa15 {
    String nim, nama, kelas;
    int nilai;

    Mahasiswa15() {
        this.nim = "";
        this.nama = "";
        this.kelas = "";
        this.nilai = -1;
    }

    Mahasiswa15(String nim, String nama, String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        this.nilai = -1;
    }

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

StackTugasMahasiswa15.java

```
public class StackTugasMahasiswa15 {
    Mahasiswa15[] stack;
    int size, top;

    StackTugasMahasiswa15(int size) {
        this.size = size;
        this.stack = new Mahasiswa15[size];
        this.top = -1;
    }

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

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

    void push(Mahasiswa15 mhs) {
        if (!isFull()) {
            stack[++top] = mhs;
        } else {
            System.out.println("Stack penuh. Tidak dapat menambahkan tugas.");
        }
    }
}
```

```

    }
}

Mahasiswa15 pop() {
    if (!isEmpty()) {
        return stack[top--];
    } else {
        System.out.println("Stack kosong. Tidak ada tugas
yang bisa dinilai.");
        return null;
    }
}

Mahasiswa15 peek() {
    if (!isEmpty()) {
        return stack[top];
    } else {
        System.out.println("Stack kosong.");
        return null;
    }
}

void print() {
    if (!isEmpty()) {
        for (int i = top; i >= 0; i--) {
            System.out.println(stack[i].nama + "\t" +
stack[i].nim + "\t" + stack[i].kelas);
        }
    } else {
        System.out.println("Stack kosong.");
    }
}

Mahasiswa15 bottom() {
    if (!isEmpty()) {
        return stack[0];
    } else {
        System.out.println("Stack kosong.");
        return null;
    }
}
}

```

MahasiswaDemo15.java

```

import java.util.Scanner;

public class MahasiswaDemo15 {
    public static void main(String[] args) {
        StackTugasMahasiswa15 stack = new
StackTugasMahasiswa15(5);
        Scanner scan = new Scanner(System.in);
        int pilih;

        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.println("5. Melihat Tugas Pertama yang
Dikumpulkan");
        System.out.println("0. Keluar");
        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();
                Mahasiswa15 mhs = new Mahasiswa15(nim,
nama, kelas);
                stack.push(mhs);
                System.out.printf("Tugas %s berhasil
dikumpulkan\n", mhs.nama);
                break;
            case 2:
                Mahasiswa15 dinilai = stack.pop();
                if (dinilai != null) {
                    System.out.println("Menilai tugas dari
" + dinilai.nama);
                    System.out.print("Masukkan 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:
                Mahasiswa15 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;
            case 5:
                Mahasiswa15 terbawah = stack.bottom();
                if (terbawah != null) {
                    System.out.println("Tugas pertama
dikumpulkan oleh " + terbawah.nama);
                }
                break;
            case 0:
                System.out.println("Terima kasih.");
                break;
            default:
                System.out.println("Pilihan tidak valid.");
        }
    } while (pilih != 0);
}

```

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

Jawab:

Menambahkan method `jumlahTugas` pada class `stacktugasmahasiswa`

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

Menambahkan menu operasi di `classdemo`

```
System.out.println(x:"6. Lihat jumlah tugas yang sudah dikumpulkan");
```

Menambahkan class handling

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

PERCOBAAN 2

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

```
public void konversiDesimalKeBiner(int kode) {  
    StackKonversi15 biner = new StackKonversi15(size:10);  
    while (kode > 0) {  
        int sisa = kode % 2;  
        biner.push(sisa);  
        kode = kode / 2;  
    }  
    System.out.print(s:"Nilai dalam biner: ");  
    while (!biner.isEmpty()) {  
        System.out.print(biner.pull());  
    }  
    System.out.println();  
}
```

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

```
public class StackKonversi15 {  
    int[] stack;  
    int size, top;  
  
    StackKonversi15(int size) {  
        this.size = size;  
        this.stack = new int[size];  
        this.top = -1;  
    }  
  
    boolean isEmpty() {  
        return top == -1;  
    }  
  
    boolean isFull() {  
        return top == size - 1;  
    }  
  
    void push(int data) {  
        if (!isFull()) {
```

```

        stack[++top] = data;
    } else {
        System.out.println("Stack konversi penuh.");
    }
}

int pull() {
    if (!isEmpty()) {
        return stack[top--];
    } else {
        System.out.println("Stack konversi kosong.");
        return -1;
    }
}
}

```

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:
    Mahasiswa15 dinilai = stack.pop();
    if (dinilai != null) {
        System.out.println("Menilai tugas dari " + 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);
        stack.konversiDesimalKeBiner(nilai);
    }
    break;

```

Hasil run program

```

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Pertama yang Dikumpulkan
6. Lihat jumlah tugas yang sudah dikumpulkan
0. Keluar
Pilih: 2
Menilai tugas dari tika
Masukkan nilai (0-100): 87
Nilai Tugas tika adalah 87
Nilai dalam biner: 1010111

```

Pertanyaan

1. Jelaskan alur kerja dari method **konversiDesimalKeBiner**!

Jawab: Method **konversiDesimalKeBiner** bekerja dengan cara mengubah angka desimal menjadi biner menggunakan prinsip pembagian dengan dua. Saat method ini dipanggil, angka desimal yang dimasukkan akan dibagi dengan 2 secara berulang-ulang, dan setiap sisa pembagian (antara 0 atau 1) akan disimpan ke dalam struktur data stack. Proses ini terus dilakukan sampai angka desimal menjadi 0. Setelah itu, isi stack akan dikeluarkan satu per satu dari atas ke bawah (menggunakan operasi **pull**) dan ditampilkan, sehingga membentuk angka biner yang benar. Penggunaan stack di sini penting karena urutan biner harus dibalik dari hasil pembagian biasa agar sesuai dengan bentuk biner yang benar.

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

Jawab: Jika kondisi perulangan diubah menjadi **while (kode != 0)**, hasil konversi tetap akan benar selama nilai awal dari kode lebih dari 0. Hal ini karena sebenarnya pengecekan kode > 0 dan kode

!= 0 akan menghasilkan efek yang sama untuk angka-angka positif, yaitu terus mengulang selama kode belum menjadi nol. Namun, menggunakan kode != 0 terdengar lebih umum dan fleksibel karena tidak membatasi pada angka positif saja. Dengan kata lain, hasilnya tidak akan berubah secara fungsional, tetapi gaya penulisan kodenya jadi sedikit lebih universal dan mudah dibaca.

TUGAS

```
public class Surat15{
    String idSurat,namaMahasiswa,kelas;
    char jenisIzin;
    int durasi;
    boolean statusCek;

    Surat15() {
    }

    Surat15(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;
        this.statusCek = false;
    }

    void suratDicek() {
        this.statusCek = true;
    }
}
```

```
public class StackSurat15 {

    Surat15[] stack;

    int top;

    int size;

    public StackSurat15(int size) {

        this.size = size;

        stack = new Surat15[size];

        top = -1;

    }

    public boolean isFull() {

        return top == size - 1;

    }

    public boolean isEmpty() {

        return top == -1;

    }

}
```

```

    public void push(Surat15 srt) {
        if (!isFull()) {
            top++;
            stack[top] = srt;
        } else {
            System.out.println("Stack penuh! Tidak bisa
menambahkan surat lagi.");
        }
    }

    public Surat15 pop() {
        if (!isEmpty()) {
            Surat15 s = stack[top];
            top--;
            return s;
        } else {
            System.out.println("Stack kosong! Tidak ada surat
untuk dicek.");
            return null;
        }
    }

    public Surat15 peek() {
        if (!isEmpty()) {
            return stack[top];
        } else {
            System.out.println("Stack kosong! Tidak ada surat
yang dikumpulkan.");
            return null;
        }
    }
}

```

```

import java.util.Scanner;

public class MainSurat15 {

```

```

public static void main(String[] args) {
    StackSurat15 stack = new StackSurat15(10);
    Scanner input = new Scanner(System.in);
    int menu;
    Surat15[] srtTerverif = new Surat15[10];
    int jmlVerif = 0;

    do {
        System.out.println("\n===== MENU
=====");
        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");
        System.out.println("5. Keluar Program");
        System.out.print("Pilih : ");
        menu = input.nextInt();
        input.nextLine();

        switch (menu) {
            case 1:
                System.out.println();
                System.out.print("ID Surat : ");
                String idSurat = input.nextLine();
                System.out.print("Nama Mahasiswa : ");
                String namaMahasiswa = input.nextLine();
                System.out.print("Kelas : ");
                String kelas = input.nextLine();
                System.out.print("Jenis perizinan (I/S) :
");
                char jenisIzin = input.next().charAt(0);
                input.nextLine();
                System.out.print("Lama waktu izin : ");
                int durasi = input.nextInt();

```

```

        input.nextLine();

        Surat15 srt = new Surat15(idSurat,
namaMahasiswa, kelas, jenisIzin, durasi);

        stack.push(srt);

        System.out.printf("Surat %s berhasil
dikumpulkan\n", srt.namaMahasiswa);

        break;

    case 2:

        Surat15 dicek = stack.pop();

        if (dicek != null) {

            System.out.println();

            System.out.println("Memverifikasi surat
dari " + dicek.namaMahasiswa);

            dicek.suratDicek();

            Surat15 temp = dicek;

            srtTerverif[jmlVerif] = temp;

            System.out.printf("Surat dari mahasiswa
%s sudah di verifikasi.\n", dicek.namaMahasiswa);

            jmlVerif++;

        }

        break;

    case 3:

        Surat15 lihat = stack.peek();

        if (lihat != null) {

            System.out.println("\nSurat terakhir
dikumpulkan oleh : " + lihat.namaMahasiswa);

        } else {

            System.out.println("\nTidak ada surat
yang dikumpulkan.");

        }

        break;

    case 4:

        System.out.println();

```

```

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

        String namaCari = input.nextLine();

        boolean find = false;

        int posisi = -1;

        for (int j = 0; j < stack.size; j++) {

            if (stack.stack[j] != null &&
stack.stack[j].namaMahasiswa.equalsIgnoreCase(namaCari)) {

                find = true;

                posisi = j;

                break;

            }

        }

        if (!find) {

            for (int j = 0; j < srtTerverif.length;
j++) {

                if (srtTerverif[j] != null &&
srtTerverif[j].namaMahasiswa.equalsIgnoreCase(namaCari)) {

                    find = true;

                    posisi = j;

                    break;

                }

            }

        }

        if (find) {

            if (posisi >= 0 && posisi < stack.size
&& stack.stack[posisi] != null) {

                Surat15 suratDitemukan =
stack.stack[posisi];

                System.out.println("\nSurat
Ditemukan : ");

                System.out.println("ID Surat : " +
suratDitemukan.idSurat);

                System.out.println("Nama Mahasiswa
: " + suratDitemukan.namaMahasiswa);

                System.out.println("Kelas : " +
suratDitemukan.kelas);

```

```

                System.out.println("Jenis Perizinan
: " + suratDitemukan.jenisIzin);

                System.out.println("Lama Waktu Izin
: " + suratDitemukan.durasi);

                } else if (posisi >= 0 && posisi <
srtTerverif.length && srtTerverif[posisi] != null) {

                Surat15 suratDitemukan =
srtTerverif[posisi];

                System.out.println("\nSurat
Ditemukan : ");

                System.out.println("ID Surat : " +
suratDitemukan.idSurat);

                System.out.println("Nama Mahasiswa
: " + suratDitemukan.namaMahasiswa);

                System.out.println("Kelas : " +
suratDitemukan.kelas);

                System.out.println("Jenis Perizinan
: " + suratDitemukan.jenisIzin);

                System.out.println("Lama Waktu Izin
: " + suratDitemukan.durasi);

                }

                } else {

                System.out.println("\nSurat Mahasiswa "
+ namaCari + " tidak ditemukan dalam daftar surat.");

                }

                break;

        case 5:

                System.out.println("\nTerima kasih, program
selesai.");

                break;

        default:

                System.out.println("\nPilihan tidak valid,
silakan coba lagi.");

                }

        } while (menu != 5);

    }
}

```