LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR STACK

ABIM MUSTAWA

244107020078

KELAS TI-1B

PRODI D-IV TEKNIK INFORMATIKA JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

1. Percobaan Praktikum

1.1 Praktikum 1 – Mahasiswa Mengumpulkan Tugas

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 nim:
- String nama:
- String kelas:
- String nilai: int
- Mahasiswa()
- Mahasiswa(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 stack:

- Mahasiswa[]
- size: int
- top: int
- StackTugasMahasiswa(size: int)
- isFull(): boolean
- isEmpty(): boolean
- push(mhs): void
- pop(): Mahasiswa
- peek(): Mahasiswa
- print(): void Pada

Langkah-langkah Percobaan

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

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

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

```
public class StackTugasMahasiswa01 {
    Mahasiswa01[] stack;
    int top, size;
```

6. Lengkapi class StackTugasMahasiswa dengan atribut yang telah digambarkan di dalam class diagram StackTugasMahasiswa, yang terdiri dari atribut stack, size, dan top
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 StackTugasMahasiswa01(int size){
        this.size = size;
        stack = new Mahasiswa01[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(Mahasiswa01 mhs){
    if (!isFull()) {
        top++;
        stack[top] = mhs;
    }else {
        System.out.println(x:"Stack penuh! Tidak bisa menambahkan tuga
    }
}
```

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. 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

```
public Mahasiswa01 pop(){
    if (!isEmpty()) {
        Mahasiswa01 m = stack[top];
        top--;
        return m;
    }else {
        System.out.println(x:"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 Mahasiswa01 peek() {
    if (!isEmpty()) {
        return stack[top];
    }else {
        System.out.println(x:"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();
}
```

Class Utama :

14. Buat file baru, beri nama MahasiswaDemo.java
15. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi main
16. Di dalam fungsi main, lakukan instansiasi object StackTugasMahasiswa bernama stack dengan nilai parameternya adalah 5.
17. Deklarasikan Scanner dengan nama variabel scan dan variabel pilih bertipe int
18. Tambahkan menu untuk memfasilitasi pengguna dalam memilih operasi Stack dalam mengelola data tugas mahasiswa menggunakan struktur perulangan do-while

```
import java.util.Scanner;
public class MahasiswaDemo01 {
    Run | Debug
    public static void main(String[] args) {
        StackTugasMahasiswa01 stack = new StackTugasMahasiswa01(size:5);
        Scanner sc = new Scanner(System.in);
        int pilih;
        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:"Pilih: ");
            pilih = sc.nextInt();
            sc.nextLine();
            switch (pilih) {
                case 1:
                    System.out.println(x:"Nama : ");
                    String nama = sc.nextLine();
                    System.out.println(x:"NIM : ");
                    String nim = sc.nextLine();
                    System.out.println(x:"Kelas : ");
                    String kelas = sc.nextLine();
                    Mahasiswa01 mhs = new Mahasiswa01(nama, nim, kelas);
                    stack.push(mhs);
                    System.out.println("Tugas %s berhasil dikumpulkan\n" + mhs.nama);
                    break;
                case 2:
                    Mahasiswa01 dinilai = stack.pop();
                    if (dinilai != null) {
                        System.out.println("Menilai tugas dari " + dinilai.nama);
                        System.out.println(x:"Masukkan nilai (0-100): ");
                        int nilai = sc.nextInt();
                        dinilai.tugasDinilai(nilai);
                        System.out.println(x:"Nilai Tugas %s adalh %d\n");
                    }
```

```
                case 3:
                    Mahasiswa01 lihat = stack.pop();
                    if (lihat != null) {
                        System.out.println("Tugas terkhir dikumpulkan oleh " + lihat.nama);
                    }
                    break;
                case 4:
                    System.out.println(x:"Daftar semua tugas");
                    System.out.println(x:"Nama\tNIM\tKelas");
                    stack.print();
                    break;
                default:
                    System.out.println(x:"Pilihan tidak valid");
            }
        } while (pilih >= 1 && pilih <= 4);
    }
}
```

19. Commit dan push kode program ke Github
20. Compile dan run program.

Verifikasi Hasil Percobaan

```
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 terkhir 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
5. Melihat Tugas Terbawah
6. Jumlah Tugas yang Telah Dikumpulkan
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
5. Melihat Tugas Terbawah
6. Jumlah Tugas yang Telah Dikumpulkan
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
5. Melihat Tugas Terbawah
6. Jumlah Tugas yang Telah Dikumpulkan
Pilih: 4
Daftar semua tugas
Nama    NIM     Kelas
Erik    1002    1B
Dila    1001    1A
```

Pertanyaan

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

   Jawab:

```java
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();
}
```

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

   Jawab :

   Banyak data yang dapat ditampung yaitu 5.

```java
StackTugasMahasiswa01 stack = new StackTugasMahasiswa01(size:5);
```

3. Mengapa perlu pengecekan kondisi !isFull() pada method push? Kalau kondisi if-else tersebut dihapus, apa dampaknya?
   Jawab :
   Supaya program menolak menambah data apabila stack sudah penuh.

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 :

```
case 5:
    Mahasiswa01 terbawah = stack.peekBottom();
    if (terbawah != null) {
        System.out.println("Tugas pertama dikumpulkan oleh " + terbawah.nama);
    }
    break;
```

```
public Mahasiswa01 peekBottom() {
    if (!isEmpty()) {
        return stack[0];
    } else {
        System.out.println(x:"Stack kosong! Tidak ada tugas yang dikumpulkan.");
        return null;
    }
}
```

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

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

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

6. Commit dan push kode program ke Github

1.2 Praktikum 2 – Konversi Nilai Tugas ke Biner

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.

Langkah-langkah Percobaan

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

bertipe int 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

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.

```
public String konversiDesimalKeBiner(int nilai){
    StackKonversi01 stack = new StackKonversi01(nilai);
    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;
}
```

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

```java
public class StackKonversi01 {
    int[] tumpukanBiner;
    int size;
    int top;

    public StackKonversi01(int size) {
        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 MahasiswaDemo

```
    break;
case 2:
    Mahasiswa01 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.nam
        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

Verifikasi Hasil Percobaan

```
Menu
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat daftar Tugas
5. Melihat Tugas Terbawah
6. Jumlah Tugas yang Telah Dikumpulkan
Pilih: 2
Menilai tugas dari Tika
Masukkan nilai (0-100): 87
Nilai Tugas Tika adalah 87
Nilai Biner Tugas: 1010111
```

Pertanyaan

1. Jelaskan alur kerja dari method konversiDesimalKeBiner!

   Jawab:

   Pertama membuat stack kosong untuk menyimpan nilai sisa pembagian. Perulangan while (nilai > 0), kode pertama pada while berfungsi untuk membagi nilai dengan 2, kode kedua berfungsi untuk menyimpan sisanya ke dalam stack, kode ketiga berfungsi untuk mengganti nilai dengan hasil pembagian nilai = nilai / 2, ulangi sampai nilai menjadi 0. Lalu mengambil sisa pembagian dari stack dan menggabungkannya menjadi string biner. Yang terakhir mengembalikan hasil konversi dalam bentuk string biner.

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

   Jawab:

   Jika kondisi perulangan diubah menjadi while (kode != 0), hasilnya akan tetap benar asalkan variabel kode bernilai sama dengan niilai yang positif, karena loop akan berhenti saat kode mencapai 0, namun jika kode bernilai nigatif atau tidak terkait dengan nilai, hasinya bisa salah karena loop tidak akan berhenti.

2. Latihan Praktikum

2.1 Latihan 1

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 idSurat: String
- namaMahasiswa: String
- kelas: String
- jenisIzin: char
- durasi: int
- Surat()
- Surat(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:

a. Terima Surat Izin untuk memasukkan data surat
b. Proses Surat Izin untuk memproses atau memverifikasi surat
c. Lihat Surat Izin Terakhir untuk melihat surat teratas
d. Cari Surat untuk mencari ada atau tidaknya surat izin berdasarkan nama mahasiswa

Jawab:

```
Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 1
ID Surat : 112
Nama Mahasiswa : Harisun
Kelas : 1A
jenis Izin (S/I) : S
Durasi (jam) : 2
Surat berhasil ditambahkan!

Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 1
ID Surat : 223
Nama Mahasiswa : Abim
Kelas : 1B
jenis Izin (S/I) : I
Durasi (jam) : 4
Surat berhasil ditambahkan!
```

```
Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 3
Surat Terakhir Masuk:
ID Surat : 223
Nama Mahasiswa : Abim
Kelas : 1B
Jenis Izin : Izin
Durasi : 4 jam

Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 2

Memproses Surat Izin dari Abim
Detail Surat:
ID Surat : 223
Nama Mahasiswa : Abim
Kelas : 1B
Jenis Izin : Izin
Durasi : 4 jam
```

```
Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 3
Surat Terakhir Masuk:
ID Surat : 112
Nama Mahasiswa : Harisun
Kelas : 1A
Jenis Izin : sakit
Durasi : 2 jam

Menu Pengelolaan Surat Izin
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
5. Keluar
Pilih Menu : 4
Masukkan Nama Mahasiswa : harisun
Surat ditemukan:
ID: 112 | Nama: Harisun | Kelas: 1A | Izin: Sakit | Durasi: 2 hari
```