Nama : Siti Mutmainah

Kelas : TI-1B/21

NIM : 244107020143

PERCOBAAN 1

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

Langkah-langkah

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

```
String nama, nim, 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

```
Mahasiswa21(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;
}
```

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 StackTugasMahasiswa21 {
```

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

```
Mahasiswa21 [] 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 StackTugasMahasiswa21(int size){
    this.size = size;
    stack = new Mahasiswa21[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(Mahasiswa21 mhs) {
    if (!isFull()){
        top++;
        stack[top] = mhs;
    } else {
        System.out.println(x:"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 Mahasiswa21 pop() {
    if (!isEmpty()) {
        Mahasiswa21 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 Mahasiswa21 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(x:"");
}
```

Class Utama

14. Buat file baru, beri nama MahasiswaDemo.java

```
public class MahasiswaDemo21 {
    Run | Debug
```

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

```
StackTugasMahasiswa21 stack = new StackTugasMahasiswa21(size:5);
```

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

```
import java.util.Scanner;
public class MahasiswaDemo21 {
```

```
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(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 = 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();
        Mahasiswa21 mhs = new Mahasiswa21(nama, nim, kelas);
        stack.push(mhs);
        System.out.printf(format:"Tugas %s berhasil dikumpulkan\n", mhs.nama);
        break;
    case 2:
        Mahasiswa21 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:"Tugas %s adalah %d\n", dinilai.nama, nilai);
        }
        break;
```

```
        case 3:
            Mahasiswa21 lihat = stack.peek();
            if (lihat != null){
                System.out.println("Tugas Terakhir 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.

```
PS C:\Users\ASUS\OneDrive\Documents\Semester 2\Praktikum ASD\Jobsheet\Jobsh
9>  & 'C:\Program Files\Java\jdk-24\bin\java.exe' '--enable-preview' '-XX:+
wCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ASUS\AppData\Roaming\Code\
r\workspaceStorage\c056a271b213c96bf51f5aef79a55e21\redhat.java\jdt_ws\Jobs
t9_66eaaf68\bin' 'MahasiswaDemo21'

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
dila    1001    1A
erik    1002    1B
tika    1003    1C
```

```
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
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
dila    1001    1A
erik    1002    1B
```

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(x:"");
}
```

```
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
```

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

   Jawab:

   Ada 5 data yang dapat ditampung

```
StackTugasMahasiswa21 stack = new StackTugasMahasiswa21(size:5);
```

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

   Jawab:

   Jadi method push berfungsi untuk menambahkan data, sebelum menambahkan data wajib dicek terlebih dahulu ada indeks yang kosong atau tidak, jika ada yang kosong maka data bisa ditambahkan, jika tidak ada yang kosong maka akan masuk ke else dan keluar output stack penuh. Jika kondisi if-else dihapus maka dampaknya top akan terus berjalan, Ketika stack penuh top akan terus bertambah dan melebihi kapasitas array akan muncul error.

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 Mahasiswa21 TugasTerbawah(){
    if (!isEmpty()) {
        return stack[0];
    } else{
        System.out.println(x:"Stack kosong! Tidak ada tugas yang dikumpulkan");
        return null;
    }
}
```

```
System.out.println(x:"5. Lihat tugas terbawah");
```

```
case 5:
    Mahasiswa21 TugasTerbawah = stack.TugasTerbawah();
    if (TugasTerbawah != null) {
    System.out.println("Tugas terbawah adalah " + TugasTerbawah.nama);
    }
    break;
```

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Lihat tugas terbawah
6. Jumlah tugas yang dikumpulkan
Pilih: 4
Daftar semua tugas
Nama    NIM     Kelas
Adel    14      1A
adel    13      1A
iin     12      1A


Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Lihat tugas terbawah
6. Jumlah tugas yang dikumpulkan
Pilih: 5
Tugas terbawah adalah iin
```

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

   Jawab:

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

```
System.out.println(x:"6. Jumlah tugas yang dikumpulkan");
```

```
case 6:
int Jumlah = stack.JumlahTugas();
System.out.println("Jumlah tugas yang dikumpulkan " + Jumlah);
break;

default:
System.out.println(x:"Pilihan tidak valid");
}
```

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Lihat tugas terbawah
6. Jumlah tugas yang dikumpulkan
Pilih: 4
Daftar semua tugas
Nama    NIM     Kelas
adel    1235    1A
iin     1234    1A


Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Lihat tugas terbawah
6. Jumlah tugas yang dikumpulkan
Pilih: 6
Jumlah tugas yang dikumpulkan 2
```

6. Commit dan push kode program ke Github

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

```
public class StackKonversi {
```

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, top;

    public StackKonversi(){
        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

```
case 2:
    Mahasiswa21 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:"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

```
Pilih: 2
Menilai tugas dari iin
Masukkan nilai (0-100): 12
Tugas iin adalah 12
Nilai biner Tugas: 1100
```

6. Commit dan push kode program ke Github

PERTANYAAN

1. Jelaskan alur kerja dari method konversiDesimalKeBiner!

   Jawab:

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

   Stack digunakan untuk menyimpan sisa hasil pembagian nilai desimal dengan 2. Menggunakan stack karena dalam konversi desimal ke biner, kita butuh hasil sisa pembagian dari belakang ke depan (last in, first out)

```
while (nilai > 0) {
    int sisa = nilai % 2;
    stack.push (sisa);
    nilai = nilai/2;
}
```

   Disini nilai dibagi 2, simpan sisanya ke stack ullangi proses ini sampai nilai jadi 0

```
String biner = new String();
while (!stack.isEmpty()){
    biner += stack.pop();
}
```

   Setelah pembagian selesai, kita ambil sisa-sisa dari stack satu per satu (dengan .pop()) dan susun jadi string biner. Karena stack bekerja seperti tumpukan (yang terakhir masuk akan pertama keluar), maka hasilnya sudah dalam urutan biner yang benar

```
    return biner;
}
```

Kita kembalikan hasil konversi sebagai string

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

   Jawab:

```
public String konversiDesimalKeBiner(int nilai){
    StackKonversi stack = new StackKonversi();
    while (kode != 0) {
```

Hasilnya error karena tidak ada variable bernama kode yang pernah dibuat sebelumnya

TUGAS

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

1. Surat21

```
public class Surat21 {
    String idSurat, namaMahasiswa, kelas;
    char jenisIzin;
    int durasi;

    public Surat21(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 void tampil() {
    System.out.println("ID surat        : " + idSurat);
    System.out.println("Nama Mahasiswa  : " + namaMahasiswa);
    System.out.println("Kelas           : " + kelas);
    System.out.println("Jenis Izin      : " + jenisIzin);
    System.out.println("Durasi          : " + durasi + " Hari");
}

}
```

2. StackSurat21

```
public class StackSurat21 {
    int size;
    int top;
    Surat21[] stack;

    public StackSurat21(int size) {
        this.size = size;
        stack = new Surat21[size];
        top = -1;
    }

    public void pushSurat21(Surat21 surat) {
        if (top == size - 1) {
            System.out.println(x:"Stack penuh!");
        } else {
            stack[++top] = surat;
        }
    }

    public Surat21 popSurat21() {
        if (top == -1) {
            System.out.println(x:"Stack kosong!");
            return null;
        } else {
            return stack[top--];
        }
    }
```

```
    public Surat21 peekSurat21() {
        if (top == -1) {
            System.out.println(x:"Stack kosong!");
            return null;
        } else {
            return stack[top];
        }
    }

    public boolean cariSurat21(String nama) {
    for (int i = 0; i <= top; i++) {
        if (stack[i].namaMahasiswa.equalsIgnoreCase(nama)) {
            return true;
        }
    }
    return false;
}
}
```

3. SuratDemo21

```java
import java.util.Scanner;
public class SuratDemo21 {
    Run | Debug
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        StackSurat21 stack = new StackSurat21(size:5);

        int pilih;
        do {
            System.out.println(x:"\nMenu:");
            System.out.println(x:"1. Terima Surat Izin");
            System.out.println(x:"2. Proses Surat Izin");
            System.out.println(x:"3. Lihat Surat Izin Terakhir");
            System.out.println(x:"4. Cari surat berdasarkan Nama Mahasiswa");
            System.out.println(x:"5. Keluar");
            System.out.print(s:"Pilih: ");
            pilih = input.nextInt();
            input.nextLine();

            switch (pilih) {
                case 1:
                    System.out.print(s:"ID Surat: ");
                    String idSurat = input.nextLine();
                    System.out.print(s:"Nama Mahasiswa: ");
                    String namaMahasiswa = input.nextLine();
                    System.out.print(s:"Kelas: ");
                    String kelas = input.nextLine();
                    System.out.print(s:"Jenis Izin (S/I): ");
                    char jenisIzin = input.next().charAt(index:0);
                    System.out.print(s:"Durasi Izin (hari): ");
                    int durasi = input.nextInt();
                    input.nextLine();
```

```java
                    Surat21 suratBaru = new Surat21(idSurat, namaMahasiswa, kelas, jenisIzin, durasi);
                    stack.pushSurat21(suratBaru);
                    break;

                case 2:
                    Surat21 suratProses = stack.popSurat21();
                    if (suratProses != null) {
                        System.out.println("Memproses surat milik: " + suratProses.namaMahasiswa);
                    }
                    break;

                case 3:
                    Surat21 suratTop = stack.peekSurat21();
                    if (suratTop != null) {
                        System.out.println("Surat Terakhir dari: " + suratTop.namaMahasiswa);
                    }
                    break;
```

```java
                case 4:
                    System.out.print(s:"Masukkan nama mahasiswa: ");
                    String cariNama = input.nextLine();
                    boolean ditemukan = stack.cariSurat21(cariNama);
                    if (ditemukan) {
                        System.out.println("Surat ditemukan untuk " + cariNama);
                    } else {
                        System.out.println("Surat tidak ditemukan untuk " + cariNama);
                    }
                    break;

                case 5:
                    System.out.println(x:"Keluar...");
                    break;

                default:
                    System.out.println(x:"Pilihan tidak valid!");
            }

        } while (pilih != 5);
    }
}
```

4. Hasil Run

```
Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari surat berdasarkan Nama Mahasiswa
5. Keluar
Pilih: 1
ID Surat: 1234
Nama Mahasiswa: iin
Kelas: 1B
Jenis Izin (S/I): s
Durasi Izin (hari): 2

Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari surat berdasarkan Nama Mahasiswa
5. Keluar
Pilih: 1
ID Surat: 0212
Nama Mahasiswa: adel
Kelas: 1B
Jenis Izin (S/I): i
Durasi Izin (hari): 1
```

```
Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari surat berdasarkan Nama Mahasiswa
5. Keluar
Pilih: 2
Memproses surat milik: adel

Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari surat berdasarkan Nama Mahasiswa
5. Keluar
5. Keluar
Pilih: 3
Surat Terakhir dari: iin
```