LAPORAN HASIL PRAKTIKUM ALGORITMA DAN STRUKTUR DATA
JOBSHEET 9

NAMA :MOHAMAT FAUZI ROHMAN

NIM: 244107020067

KELAS : TI_1E

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2024

JOBSHEET IX

STACK

9. Praktikum

9.1 Percobaan 1: Mahasiswa Mengumpulkan Tugas

9.1.1 Langlah-langkah Percobaan

A. Class Mahasiswa

1. Buat folder baru bernama Jobsheet9 di dalam repository Praktikum ASD. Buat file baru, beri nama Mahasiswa16.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.

```
public class Mahasiswa16 {

    String nim;
    String nama;
    String kelas;
    int nilai;

Mahasiswa16(String nama, String nim, String kelas){
    this.nama = nama;
    this.nim = nim;
    this.kelas = kelas;
    nilai = -1;
}
```

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

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

B. Class StackTugasMahasiswa

5. Setelah membuat class Mahasiswa, selanjutnya perlu dibuat class StackTugasMahasiswa16.java sebagai tempat untuk mengelola tumpukan tugas. Class StackTugasMahasiswa merupakan penerapan dari struktur data Stack

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

```
public class StackTugasMahasiswa16 {

    Mahasiswa16[] stack;
    int size;
    int top;
```

7. Tambahkan konstruktor berparameter pada class StackTugasMahasiswa16 untuk melakukan inisialisasi kapasitas maksimum data tugas mahasiswa yang dapat disimpan di dalam Stack, serta mengeset indeks awal dari pointer top

```
public StackTugasMahasiswa16(int size){
    this.size = size;
    stack = new Mahasiswa16[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 StackTugasMahasiswa16, 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 Mahasiswa16

```
public void push(Mahasiswa16 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 Mahasiswa16

```
public Mahasiswa16 pop() {
    if (!isEmpty()) {
        Mahasiswa16 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 Mahasiswa16 peek() {
    if (!isEmpty()) {
        return stack[top];
    } else{
        System.out.println("Stack kosong! Tidak ada tugas yang dikumpulkan.");
        return null;
    }
}
```

13. Tambahkan method print untuk dapat menampilkan semua daftar tugas mahasiswa pada Stack

```
public void print() {
    for (int i = 0; i <= top; i++){
        System.out.println(stack[i].nama + "\t" + stack[i].nim + "\t" + stack[i].kelas);
    }
    System.out.println("");
}
}
```

C. Class Utama

14. Buat file baru, beri nama MahasiswaDemo16.java
15. Tuliskan struktur dasar bahasa pemrograman Java yang terdiri dari fungsi main dan lakukan instansiasi object StackTugasMahasiswa bernama stack dengan nilai parameternya adalah 5. Deklarasikan juga Scanner dengan nama variabel scan dan variabel pilih bertipe int

```
import java.util.Scanner;
public class MahasiswaDemo16 {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        StackTugasMahasiswa16 stack = new StackTugasMahasiswa16(5);
int pilih;
```

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

```
do {
        System.out.println("\nMenu:");
        System.out.println("1. Mengumpulkan Tugas");
        System.out.println("2. Menilai Tugas");
        System.out.println("3. Melihat Tugas Teratas");
        System.out.println("4. Melihat Daftar Tugas");
        System.out.print("Pilih: ");
        pilih = scan.nextInt();
        scan.nextLine();

        switch (pilih) {
```

```
            case 1:
                System.out.print("Nama: ");
                String nama = scan.nextLine();
                System.out.print("NIM: ");
                String nim = scan.nextLine();
                System.out.print("Kelas: ");
                String kelas = scan.nextLine();
                Mahasiswa16 mhs = new Mahasiswa16(nama, nim, kelas);
                stack.push(mhs);
                System.out.printf("Tugas %s berhasil dikumpulkan\n", mhs.nama);
                break;

            case 2:
                Mahasiswa16 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:
                Mahasiswa16 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);
        }
    }
```

17. Commit dan push kode program ke Github
18. Compile dan run program.

9.1.2 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 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
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
Dila    1001    1A
Erik    1002    1B
```

2.1.3 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

Jawab:

1. Untuk yang diubah ialah pada bagian method push, ubah menjadi seperti ini

```
public void push(Mahasiswa16 mhs) {
        if (!isFull()) {

            for (int i = top + 1; i > 0; i--) {
                stack[i] = stack[i - 1];
            }
            stack[0] = mhs;
            top++;
        } else {
            System.out.println("Stack penuh! Tidak bisa menambahkan tugas lagi.");
        }
    }
```

Hasil :

```
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: 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
```

2. Banyak data yang ditampung oleh Stack ialah 5

```
StackTugasMahasiswa16 stack = new StackTugasMahasiswa16(5);
```

3. Berfungsi untuk memberikan umpan balik kepada pengguna dengan mencetak pesan tersebut. Agar pengguna paham bahwa operasi tidak dapat dilakukan karena Stack penuh

   Dan jika if-else dihapus dapat menyebabkan maka tidak dapat melakukan pengecekan apakah Stack sudah penuh atau belum

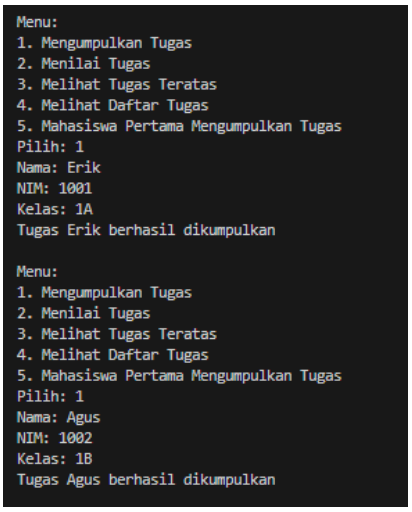
4. Tambahkan method lihatTugasTerbawah pada class StackTugasMahasiswa16

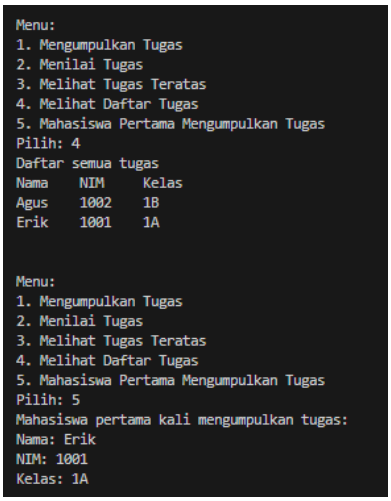
```
public Mahasiswa16 lihatTugasTerbawah() {
    if (!isEmpty()) {
        return stack[top];
    } else{
        return null;
    }
}
```

Tambahkan juga case 5 pada MahasiswaDemo16 untuk melihat siapa yang mengumpulkan tugas pertama kali

```
 case 5:
                Mahasiswa16 mhsBawah = stack.lihatTugasTerbawah();
                   if (mhsBawah != null) {
                       System.out.println("Mahasiswa pertama kali mengumpulkan tugas:");
                       System.out.println("Nama: " + mhsBawah.nama);
                       System.out.println("NIM: " + mhsBawah.nim);
                       System.out.println("Kelas: " + mhsBawah.kelas);
                   }
                   break;
```

Hasil:





5. Tambahkan method jmlTugas untuk menghitung banyaknya jumlah tugas pada class StackTugasMahasiswa16

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

Tambahkan case 6 yang berisi untuk menampilkan jumlah tugas pada MahasiswaDemo16

```
case 6:
                int jumlahTugas = stack.jmlTugas();
                System.out.println("Jumlah Tugas yang Dikumpulkan: " + jumlahTugas);
                break;
```

Hasil:

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Mahasiswa Pertama Mengumpulkan Tugas
6. Jumlah Tugas
Pilih: 1
Nama: Agus
NIM: 1001
Kelas: 1A
Tugas Agus berhasil dikumpulkan

Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Mahasiswa Pertama Mengumpulkan Tugas
6. Jumlah 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
5. Mahasiswa Pertama Mengumpulkan Tugas
6. Jumlah Tugas
Pilih: 6
Jumlah Tugas yang Dikumpulkan: 2
```

9.2 Percobaan 2: Konversi Nilai Tugas ke Biner

9.2.1 Langkah-langkah Percobaan

1. Buka kembali file StackTugasMahasiswa16.java
2. Tambahkan method konversiDesimalKeBiner dengan menerima parameter nilai bertipe int

```
public String konversiDesimalKeBiner(int nilai){
    StackKonversi16 stack = new StackKonversi16();
    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 StackKonversi16

```
public class StackKonversi16 {
    int[] tumpukanBiner;
    int size;
    int top;

public StackKonversi16(){
    this.size = 32;
    tumpukanBiner = new int[size];
    top = -1;
}

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

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

public void push(int data){
    if (isFull()) {
        System.out.println("Stack Penuh");
    } else{
        top++;
        tumpukanBiner[top] = data;
    }
}

public int pop(){
    if (isEmpty()) {
        System.out.println("Stack kosong");
        return -1;
    } else{
        int data = tumpukanBiner[top];
        top--;
        return data;
    }
}
}
```

4. Agar nilai tugas mahasiswa dikonversi ke dalam bentuk biner setelah dilakukan penilaian, maka tambahkan baris kode program pada method pop di class MahasiswaDemo16

```
case 2:
                Mahasiswa16 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);
                    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

9.2.2 Verifikasi Hasil Percobaan

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Mahasiswa Pertama Mengumpulkan Tugas
6. Jumlah Tugas
Pilih: 2
Menilai tugas dari Agus
Masukkan nilai (0-100): 80
Nilai Tugas Agus adalah 80
Nilai Biner Tugas: 1010000
```

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!

Jawab:

1. Yaitu dengan menghitung sisa dari pembagian atribut nilai dengan 2 menggunakan modulus (nilai % 2). Kemudian sisa akan menjadi digit biner dan disimpan ke dalam stack
2.

```
public String konversiDesimalKeBiner(int nilai){
    StackKonversi16 stack = new StackKonversi16();
    while (nilai != 0){
        int sisa = nilai % 2;
        stack.push(sisa);
        nilai = nilai/2;
    }
```

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Mahasiswa Pertama Mengumpulkan Tugas
6. Jumlah Tugas
Pilih: 2
Menilai tugas dari Erik
Masukkan nilai (0-100): 80
Nilai Tugas Erik adalah 80
Nilai Biner Tugas: 1010000
```

Hasilnya tetap sama seperti sebelumnya menggunakan nilai > 0. Karena nilai != 0 sama halnya dengan menggunakan nilai > 0, karena perulangan akan berhenti ketika nilai menjadi 0, yang merupakan kondisi akhir dari proses konversi. Ketika kondisi nilai = 0 makan akan menjadi false, dan perulangan akan berhenti.

9.3 Latihan Praktikum

Class Surat16

```
public class Surat16 {
    String idSurat;
    String namaMahasiswa;
    String kelas;
    char jenisIzin;
    int durasi;

Surat16(){

}

Surat16(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;
}
}
```

Class StackSurat16

```
public class StackSurat16 {
    Surat16[] stack;
    int size;
    int top;

public StackSurat16(int size){
    this.size = size;
    stack = new Surat16[size];
    top = -1;
}

public boolean isFull(){
    if (top == size - 1) {
        return true;
    } else{
        return false;
    }
}

public boolean isEmpty(){
    if (top == -1) {
        return true;
    } else{
        return false;
    }
}

public void push(Surat16 srt) {
        if (!isFull()) {
            top++;
            stack[top] = srt;
        } else {
            System.out.println("Stack penuh! Tidak bisa menambahkan surat lagi.");
        }
    }
```

```
public Surat16 pop() {
    if (!isEmpty()) {
        Surat16 s = stack[top];
        top--;
        return s;
    } else{
        System.out.println("Stack kosong! Tidak ada surat yang dapat diproses.");
        return null;
    }
}

public Surat16 peek() {
    if (!isEmpty()) {
        return stack[top];
    } else{
        System.out.println("Stack kosong! Tidak ada surat yang dikumpulkan.");
        return null;
    }
}

public void print() {
    for (int i = 0; i <= top; i++){
        System.out.println(stack[i].idSurat + "\t\t" + stack[i].namaMahasiswa +
"\t" + stack[i].kelas + "\t\s\t" + stack[i].jenisIzin + "\t" + stack[i].durasi);
    }
    System.out.println("");
}

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

Class SuratDemo16

```
import java.util.Scanner;
public class SuratDemo16 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

    StackSurat16 stack = new StackSurat16(3);
    int pilih;

     do {
        System.out.println("\nMenu:");
        System.out.println("1. Terima Surat Izin");
        System.out.println("2. Proses Surat Izin");
        System.out.println("3. Lihat Surat Izin Terakhir");
        System.out.println("4. Cari Surat");
        System.out.print("Pilih: ");
        pilih = sc.nextInt();
        sc.nextLine();
```

```
switch (pilih) {
            case 1:
                System.out.print("ID Surat\t: ");
                String idSurat = sc.nextLine();
                System.out.print("Nama Mahasiswa\t: ");
                String namaMahasiswa = sc.nextLine();
                System.out.print("Kelas\t\t: ");
                String kelas = sc.nextLine();
                System.out.print("Jenis Izin\t: ");
                char jenisIzin = sc.nextLine().charAt(0);
                System.out.print("Durasi\t\t: ");
                int durasi = sc.nextInt();
                Surat16 srt = new Surat16(idSurat, namaMahasiswa, kelas, jenisIzin, durasi );
                stack.push(srt);
                System.out.println("Surat izin dari " + srt.namaMahasiswa + " berhasil
dikumpulkan.");
                break;

            case 2:
                Surat16 proses = stack.pop();
                if (proses != null) {
                    System.out.println("Surat " + proses.jenisIzin + " dari " +
proses.namaMahasiswa + " berhasil diverifikasi.");
                }
                break;

            case 3:
                System.out.println("Daftar Semua Surat");
                System.out.println("ID Surat\tNama\tKelas\tJenis Izin\tDurasi");
                stack.print();
                break;

            case 4:
                System.out.print("Masukkan Nama Mahasiswa: ");
                String nama = sc.nextLine();
                if (stack.cari(nama)) {
                    System.out.println("Surat dengan nama " + nama + " ditemukan");
                } else{
                    System.out.println("Surat dengan nama " + nama + " tidak ditemukan");

                }
                break;

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

            }
        } while (pilih >= 1 && pilih <=4);
    }
}
```

Hasil:

```
Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 1
ID Surat        : 111
Nama Mahasiswa  : Agus
Kelas           : 1A
Jenis Izin      : S
Durasi          : 2
Surat izin dari Agus berhasil dikumpulkan.

Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 1
ID Surat        : 222
Nama Mahasiswa  : Roy
Kelas           : 1E
Jenis Izin      : I
Durasi          : 1
Surat izin dari Roy berhasil dikumpulkan.

Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 2
Surat I dari Roy berhasil diverifikasi.
```

```
Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 3
Daftar Semua Surat
ID Surat        Nama    Kelas   Jenis Izin      Durasi
111             Agus    1A              S       2


Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 4
Masukkan Nama Mahasiswa: Agus
Surat dengan nama Agus ditemukan

Menu:
1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat
Pilih: 4
Masukkan Nama Mahasiswa: Roy
Surat dengan nama Roy tidak ditemukan
```