

ALGORITMA DAN STRUKTUR DATA

“Laporan hasil Praktikum pada Jobsheet 9 “STACK” ”

Oleh:

Hafiz Rizqi Hernanda

NIM (244107020154)



Jurusan Teknologi informasi

Teknik Informatika

Politeknik Negeri Malang

2. Praktikum

2.1 Percobaan 1: Mahasiswa Mengumpulkan Tugas

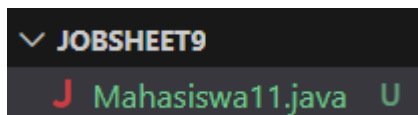
Mahasiswa<NoAbsen>
nim: String nama: String kelas: String nilai: int
Mahasiswa<NoAbsen>() Mahasiswa<NoAbsen>(nim: String, nama: String, kelas: String) tugasDinilai(nilai: int)

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

2.1.1 Langkah-langkah Percobaan

A. Class Mahasiswa

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



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

```
public class Mahasiswa11 {  
    String nim;  
    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

```
Mahasiswa11(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 StackTugasMahasiswa<noabsen>.java sebagai tempat untuk mengelola tumpukan tugas. Class StackTugasMahasiswa merupakan penerapan dari struktur data Stack

```
StackTugasMahasiswa11.java  
1 public class StackTugasMahasiswa11 {  
36  
37     public Mahasiswa11 pop() {  
38         if (isEmpty()) {
```

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

```
public class StackTugasMahasiswa11 {  
    Mahasiswa11[] 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 StackTugasMahasiswa11(int size) {  
    this.size = size;  
    stack = new Mahasiswa11[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(Mahasiswa11 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 Mahasiswa11 pop() {
    if (!isEmpty()) {
        Mahasiswa11 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 Mahasiswa11 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:"");
    }
}

```

C. Class Utama

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

 MahasiswaDemo11.java >  MahasiswaDemo11 >  main(String[])

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

```

Run | Debug
public static void main(String[] args) {

```

16. Di dalam fungsi main, lakukan instansiasi object StackTugasMahasiswa bernama stack dengan nilai parameter nya adalah 5.

```

StackTugasMahasiswa11 stack = new StackTugasMahasiswa11(size:5);

```

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

```

1  import java.util.Scanner;
2  public class MahasiswaDemo11 {
    Run | Debug
3      public static void main(String[] args) {
4          StackTugasMahasiswa11 stack = new StackTugasMahasiswa11(size:5);
5          Scanner scan = new Scanner(System.in);
6          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.println(x: "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();
            Mahasiswa11 mhs = new Mahasiswa11(nama, nim, kelas);
            stack.push(mhs);
            System.out.printf(format: "Tugas %s berhasil dikumpulkan\n", mhs.nama);
            break;
        case 2:
            Mahasiswa11 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);
            }
            break;
        case 3:
            Mahasiswa11 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.

2.1.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

Menu:

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

Pilih: 1

Nama: Dila

NIM: 1001

Kelas: 1A

Tugas Dila berhasil dikumpulkan

Menu:

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

Pilih: 1

Nama: Erik

NIM: 1002

Kelas: 1B

Tugas Erik berhasil dikumpulkan

Menu:

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

Pilih: 3

Tugas terakhir dikumpulkan oleh Erik

Menu:

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

Pilih: 1

Nama: Tika

NIM: 1003

Kelas: 1C

Tugas Tika berhasil dikumpulkan

Menu:

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

Pilih: 4

Daftar semua tugas

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

Menu:

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

Pilih: 2

Menilai tugas dari Tika

Masukkan nilai (0-100): 87

Nilai Tugas Tika adalah 87

Menu:

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

Pilih: 4

Daftar semua tugas

Nama	NIM	Kelas
Erik	1002	1B
Dila	1001	1A

Menu:

1. Mengumpulkan Tugas
2. Menilai Tugas
3. 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

```

Pada pilihan no. 4 tidak sama dengan Contoh hasil percobaan yang ditunjukkan

2.1.3 Pertanyaan

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

```

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
Tika    1003    1C
Erik    1002    1B
Dila    1001    1A

```

2. Berapa banyak data tugas mahasiswa yang dapat ditampung di dalam Stack?

Tunjukkan potongan kode programnya!

Jawab: 5 data

```
StackTugasMahasiswa11 stack = new StackTugasMahasiswa11(size:5);
```

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

`!isFull()` berfungsi untuk mencegah penambahan data saat stack sudah penuh, jika kondisi `if-else` di hapus akan terjadi:

- Program akan menimpa data di luar batas array
(ArrayIndexOutOfBoundsException)

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!

Menambahkan method baru peekBottom() di class StackTugasMahasiswa11()

```
public Mahasiswa11 peekBottom() {
    if (!isEmpty()) {
        return stack[0];
    } else {
        System.out.println(x:"Stack kosong! Tidak ada tugas.");
        return null;
    }
}
```

Tambahkan menu opsi 5

```
System.out.println(x:"5. Melihat Tugas Terbawah");
break;
case 5:
    Mahasiswa11 lihatBawah = stack.peekBottom();
    if (lihatBawah != null) {
        System.out.println("Tugas Pertama dikumpulkan oleh " + lihatBawah.nama);
    }
    break;
```

Hasil running:

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Melihat Tugas Terbawah
6. Jumlah Tugas yang sudah dikumpulkan
Pilih: 5
Tugas Pertama dikumpulkan oleh Dila
```

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

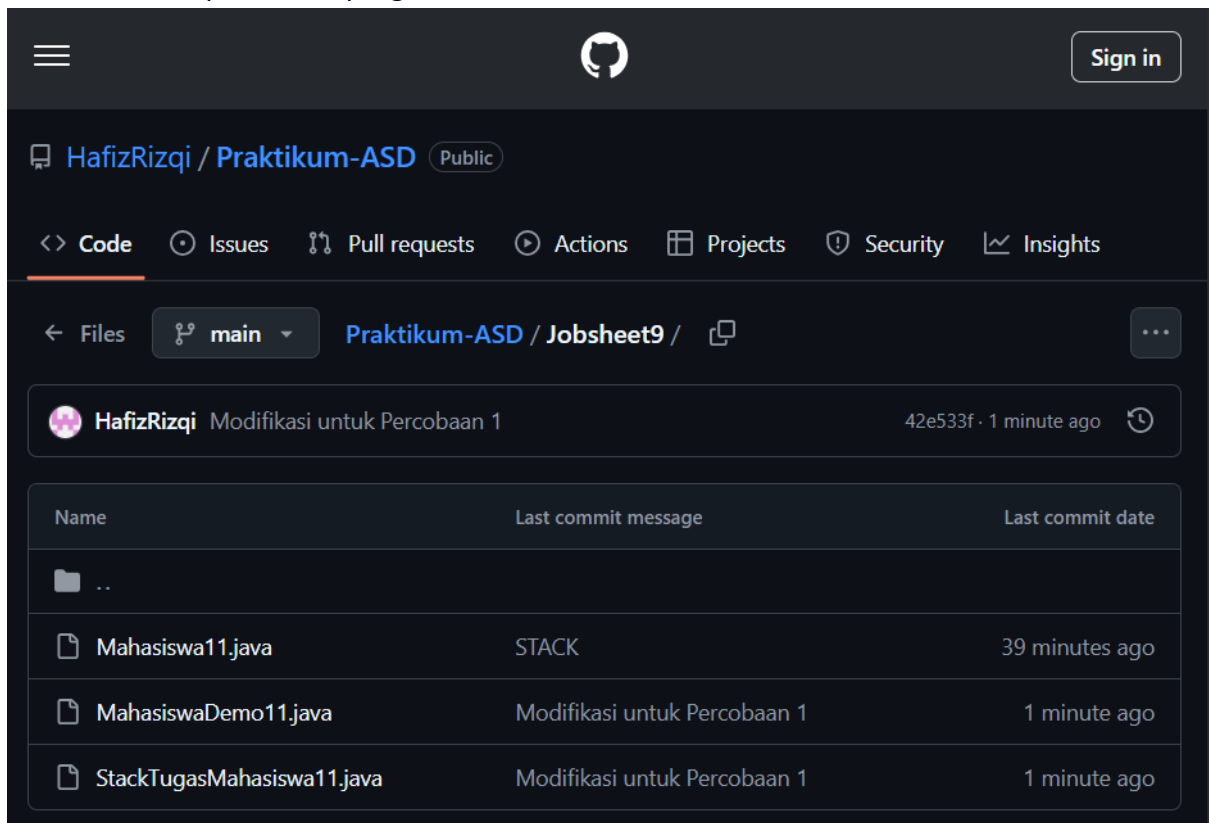
```
public int countTugas() {
    return top + 1;
}
```

```
System.out.println(x:"6. Jumlah Tugas yang sudah dikumpulkan");
break;
case 6:
    System.out.println("Jumlah Tugas yang sudah dikumpulkan: " + stack.countTugas());
    break;
```

Hasil running:

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

6. Commit dan push kode program ke Github



The screenshot shows the GitHub interface for a repository named 'Praktikum-ASD' by user 'HafizRizqi'. The repository is public. The 'Code' tab is selected, showing the file structure. The current branch is 'main'. The file 'Jobsheet9' is selected, showing a commit history table.

Name	Last commit message	Last commit date
..		
Mahasiswa11.java	STACK	39 minutes ago
MahasiswaDemo11.java	Modifikasi untuk Percobaan 1	1 minute ago
StackTugasMahasiswa11.java	Modifikasi untuk Percobaan 1	1 minute ago

2.2 Percobaan 2: Konversi Nilai Tugas ke Biner

2.2.1 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

```
public String konversiDesimalKeBiner(int nilai) {  
    StackKonversi11 stack = new StackKonversi11();  
    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

```

J StackKonversi11.java > StackKonversi11 > pop()
1  public class StackKonversi11 {
2      int[] tumpukanBiner;
3      int size;
4      int top;
5
6      public StackKonversi11() {
7          this.size = 32;
8          tumpukanBiner = new int[size];
9          top = -1;
10     }
11
12     public boolean isEmpty() {
13         return top == -1;
14     }
15
16     public boolean isFull() {
17         return top == size - 1;
18     }
19
20     public void push(int data) {
21         if (isFull()) {
22             System.out.println(x:"Stack penuh! ");
23         } else {
24             top++;
25             tumpukanBiner[top] = data;
26         }
27     }
28
29     public int pop() {
30         if (isEmpty()) {
31             System.out.println(x:"Stack kosong! ");
32             return -1;
33         } else {
34             int data = tumpukanBiner[top];
35             top--;
36             return data;
37         }
38     }
39 }

```

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

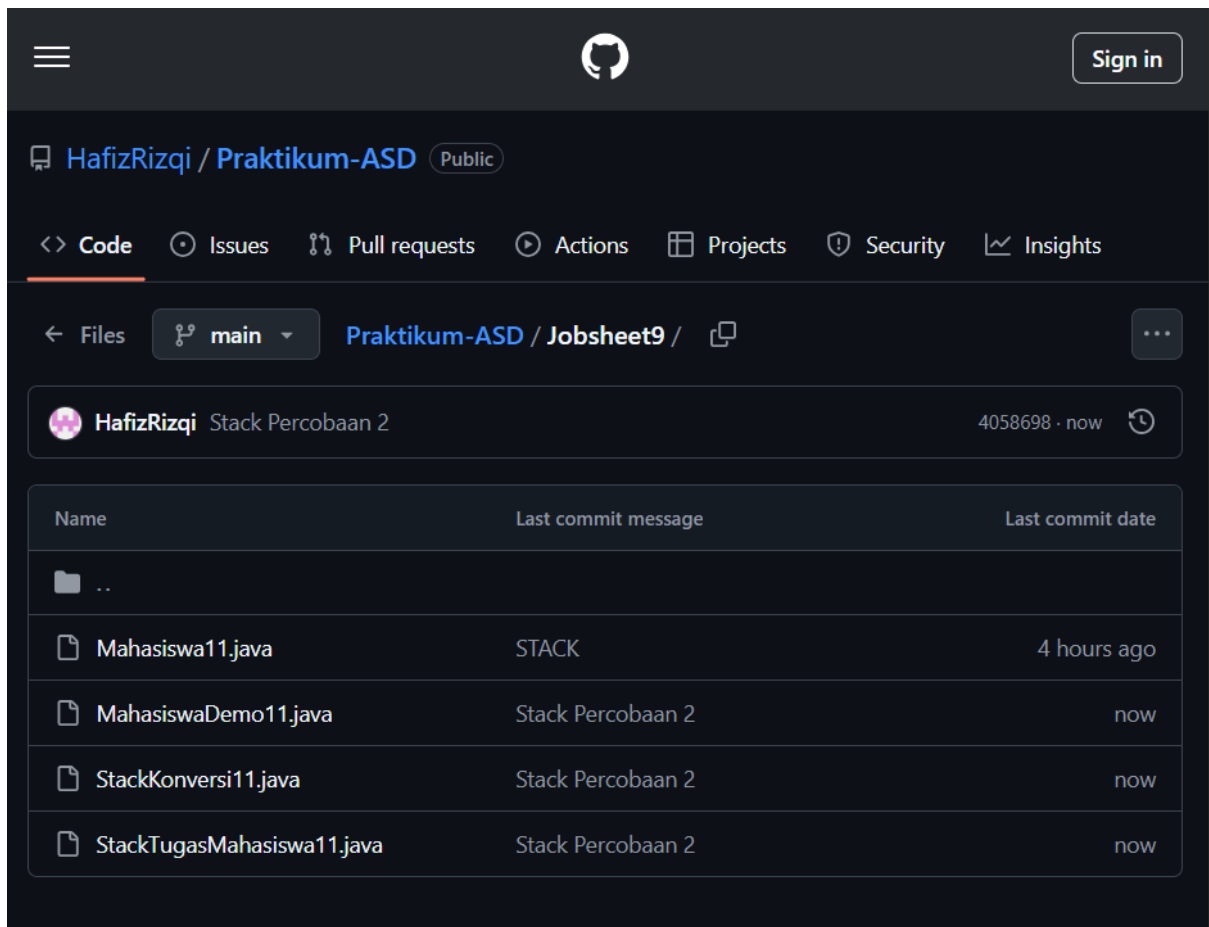
```

    case 2:
        Mahasiswa11 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);
            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



Navigation: <> Code, Issues, Pull requests, Actions, Projects, Security, Insights

Breadcrumb: < Files, main, Praktikum-ASD / Jobsheet9 /

Commit: HafizRizqi Stack Percobaan 2 4058698 · now

Name	Last commit message	Last commit date
..		
Mahasiswa11.java	STACK	4 hours ago
MahasiswaDemo11.java	Stack Percobaan 2	now
StackKonversi11.java	Stack Percobaan 2	now
StackTugasMahasiswa11.java	Stack Percobaan 2	now

2.2.2 Verifikasi Hasil Percobaan

Cocokkan hasil compile kode program Anda dengan gambar berikut ini.

```

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

```

```

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

```

2.2.3 Pertanyaan

1. Jelaskan alur kerja dari method konversiDesimalKeBiner!

Jawab: Pada Method konversiDesimalKeBiner() berfungsi untuk Mengubah **bilangan desimal** menjadi **bilangan biner** menggunakan bantuan **stack**

Untuk kode Program ini:

```

while (nilai > 0) {
    int sisa = nilai % 2;
    stack.push(sisa); // simpan sisa pembagian di stack
    nilai = nilai / 2; // bagi nilai dengan 2
}

```

Proses ini menyimpan sisa pembagian 2 kedalam stack, dikarenakan konversi biner dibaca dari bawah ke atas (sisa terakhir jadi yang paling kiri), maka stack digunakan agar nanti bisa di-pop terbalik.

Untuk kode Program ini:

```

while (!stack.isEmpty()) {
    biner += stack.pop(); // ambil satu per satu dari atas stack
}

```

Setelah selesai, isi stack dipop satu per satu dimulai dari top pada stack untuk membentuk string biner.

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

Jika while (nilai > 0) diganti menjadi while (kode != 0) akan terjadi error karena variabel kode itu tidak terbaca,

```

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

```

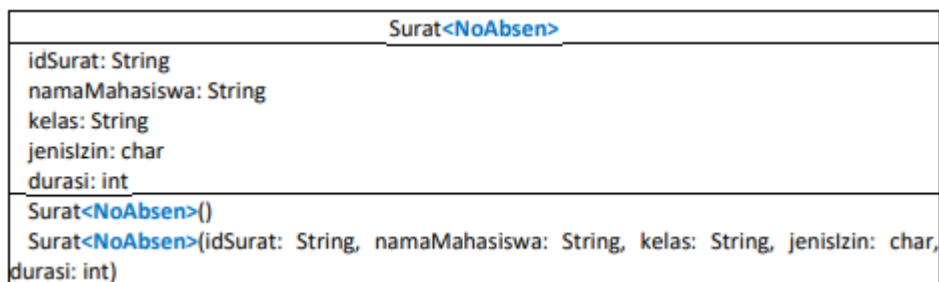
jika yang di maksud diganti menjadi while (nilai != 0)


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

Jika di run akan berjalan tetapi nilai != 0 itu berarti kan dapat menangani bilangan negatif sedangkan nilai sebuah tugas itu pasti tidak akan sampai bilangan negatif

2.4 Latihan Praktikum

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.



Atribut jenis Izin 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

```
J Surat11.java > Surat11 > tampilkanInfo()
1 public class Surat11 {
2     String idSurat;
3     String namaMahasiswa;
4     String kelas;
5     char jenisIzin;
6     int durasi;
7
8     public Surat11() {
9     }
10
11     public Surat11(String idSurat, String namaMahasiswa, String kelas, char jenisIzin, int durasi) {
12         this.idSurat = idSurat;
13         this.namaMahasiswa = namaMahasiswa;
14         this.kelas = kelas;
15         this.jenisIzin = jenisIzin;
16         this.durasi = durasi;
17     }
18
19     public void tampilkanInfo() {
20         System.out.println(idSurat + "\t" + namaMahasiswa + "\t" + kelas + "\t" + jenisIzin + "\t" + durasi);
21     }
22 }
```

```

J StackSurat11.java > StackSurat11
1 public class StackSurat11 {
2     Surat11[] stack;
3     int top;
4     int size;
5
6     public StackSurat11(int size) {
7         this.size = size;
8         stack = new Surat11[size];
9         top = -1;
10    }
11
12    public boolean isFull() {
13        return top == size - 1;
14    }
15
16    public boolean isEmpty() {
17        return top == -1;
18    }
19
20    public void push(Surat11 surat) {
21        if (!isFull()) {
22            top++;
23            stack[top] = surat;
24        } else {
25            System.out.println(x:"Stack penuh! Tidak bisa menambahkan surat izin lagi.");
26        }
27    }
28
29    public Surat11 pop() {
30        if (!isEmpty()) {
31            Surat11 surat = stack[top];
32            top--;
33            return surat;
34        } else {
35            System.out.println(x:"Stack kosong! Tidak ada surat izin untuk diambil.");
36            return null;
37        }
38    }
39

```

```

40    public Surat11 peek() {
41        if (!isEmpty()) {
42            return stack[top];
43        } else {
44            System.out.println(x:"Stack kosong! Tidak ada surat izin yang dikumpulkan.");
45            return null;
46        }
47    }
48
49    public void cariSurat(String nama) {
50        boolean ditemukan = false;
51        for (int i = 0; i <= top; i++) {
52            if (stack[i].namaMahasiswa.equalsIgnoreCase(nama)) {
53                System.out.println("Surat izin dari " + nama + " ditemukan.");
54                ditemukan = true;
55                break;
56            }
57        }
58        if (!ditemukan) {
59            System.out.println("Surat izin dari " + nama + " tidak ditemukan.");
60        }
61    }
62 }

```

J SuratMain11.java > SuratMain11 > main(String[])

```
1  import java.util.Scanner;
2  public class SuratMain11 {
    Run | Debug
3      public static void main(String[] args) {
4          Scanner scan = new Scanner(System.in);
5          StackSurat11 stack = new StackSurat11(size:10);
6          int pilih;
7
8          do {
9              System.out.println(x:"\nMenu:");
10             System.out.println(x:"1. Terima Surat Izin");
11             System.out.println(x:"2. Proses Surat Izin");
12             System.out.println(x:"3. Lihat Surat Izin Terakhir");
13             System.out.println(x:"4. Cari Surat Mahasiswa");
14             System.out.println(x:"0. Keluar");
15             System.out.print(s:"Pilih: ");
16             pilih = scan.nextInt();
17             scan.nextLine();
18
19             switch (pilih) {
20                 case 1:
21                     System.out.print(s:"ID Surat: ");
22                     String id = scan.nextLine();
23                     System.out.print(s:"Nama Mahasiswa: ");
24                     String nama = scan.nextLine();
25                     System.out.print(s:"Kelas: ");
26                     String kelas = scan.nextLine();
27                     System.out.print(s:"Jenis Izin (S/I): ");
28                     char jenis = scan.nextLine().charAt(index:0);
29                     System.out.print(s:"Durasi (hari): ");
30                     int durasi = scan.nextInt();
31                     scan.nextLine();
32
33                     Surat11 surat = new Surat11(id, nama, kelas, jenis, durasi);
34                     stack.push(surat);
35                     break;
36
37                 case 2:
38                     Surat11 diproses = stack.pop();
39                     if (diproses != null) {
40                         System.out.println(x:"Surat diproses:");
41                         System.out.println(x:"ID Surat\tNama Mahasiswa\tKelas\tJenis Izin\tDurasi");
42                         diproses.tampilkanInfo();
43                     }
44             }
9
```

```

45         break;
46
47     case 3:
48         Surat11 terakhir = stack.peek();
49         if (terakhir != null) {
50             System.out.println(x:"Surat terakhir masuk:");
51             System.out.println(x:"ID Surat\tNama Mahasiswa\tKelas\tJenis Izin\tDurasi");
52             terakhir.tampilkanInfo();
53         }
54         break;
55
56     case 4:
57         System.out.print(s:"Masukkan nama mahasiswa yang dicari: ");
58         String cari = scan.nextLine();
59         stack.cariSurat(cari);
60         break;
61
62     case 0:
63         System.out.println(x:"Keluar program.");
64         break;
65
66     default:
67         System.out.println(x:"Pilihan tidak valid.");
68     }
69 } while (pilih != 0);
70
71 scan.close();
72 }
73 }

```

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 1

ID Surat: S001

Nama Mahasiswa: Arief

Kelas: 1B

Jenis Izin (S/I): S

Durasi (hari): 3

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 1

ID Surat: S002

Nama Mahasiswa: Rizky

Kelas: 1A

Jenis Izin (S/I): S

Durasi (hari): 2

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 3

Surat terakhir masuk:

S002	Rizky	1A	S	2
------	-------	----	---	---

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 2

Surat diproses:

S002 Rizky 1A S 2

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 4

Masukkan nama mahasiswa yang dicari: Rizky
Surat izin dari Rizky tidak ditemukan.

Menu:

1. Terima Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
0. Keluar

Pilih: 4

Masukkan nama mahasiswa yang dicari: Arief
Surat izin dari Arief ditemukan.