LAPORAN HASIL PRAKTIKUM
ALGORITMA STRUKTUR DATA
JOBSHEET 9

NAMA : JIRO AMMAR WAFI

NIM : 244107020190

KELAS : 1-E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLINEMA

2025

|| JOBSHEET IX - STACK

1. Percobaan 1: Mahasiswa Mengumpulkan Tugas

A. Class Mahasiswa

1. Class Mahasiswa14 di folder Jobsheet9, membuat atribut beserta konstruktor parameter untuk mengisi nilai pada atribut nim, nama, dan kelas di dalam class:

```
package Jobsheet9;

public class Mahasiswa14 {
    String nim, nama, kelas;
    int nilai;

    Mahasiswa14 (String nim, String nama, String kelas) {
        this.nim = nim;
        this.nama = nama;
        this.kelas = kelas;
        nilai = -1;
    }
```

2. Method tugasDinilai dengan keluaran void untuk mengisi nilai untuk atribut nilai:

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

B. Class StackTugasMahasiswa

1. Pada package jobsheet9, dibuat class StackTugasMahasiswa14. Dalam class tersebut mendeklarasikan atribut stack[], size, dan top, beserta konstruktor parameter yang menginisialisasikan kapasitas maks. stack dan nilai awal top:

```
package Jobsheet9;

public class StackTugasMahasiswa14 {
    Mahasiswa14[] stack;
    int top, size;

    public StackTugasMahasiswa14(int size) {
        this.size = size;
        stack = new Mahasiswa14[size];
        top = -1;
    }
```

2. Method isFull untuk mengecek isi stack apakah telah penuh atau tidak sesuai dengan size yang ditentukan:

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

3. Method isEmpty untuk mengecek apakah stack kini telah kosong sepenuhnya:

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

4. Method push dengan parameter mhs dari class Mahasiswa, method ini berfungsi untuk menyimpan object ke stack. Apabila stack telah penuh, akan muncul notifikasi “Stack Penuh” yang menandakan bahwa array of object yang terbaru tak akan di push ke dalam stack.

```
public void push(Mahasiswa14 mhs) {
    if (!isFull()) {
        top++;
        stack[top] = mhs;
    } else {
        System.out.println(x:"Stack penuh! Tida
    }
}
```

5. Method pop dengan keluarannya ialah object dari class Mahasiswa14, yang fungsinya untuk mengeluarkan satu object yang berada pada top saat ini, apabila stack masih dalam keadaan kosong, akan muncul notifikasi di terminal bahwa stack kosong sehingga pop tak dapat dijalankan.

```
public Mahasiswa14 pop() {
    if (!isEmpty()){
        Mahasiswa14 m = stack[top];
        top--;
        return m;
    } else {
        System.out.println(x:"Stack kosong! Tidak ada tug
        return null;
    }
}
```

6. Method peek() dengan keluaran object dari class Mahasiswa14, method ini dijalankan untuk melihat isi paling atas pada stack dengan menggunakan indeks top saat ini, peek dieksekusi tanpa menghilangkan isi dari top tersebut seperti method pop() :

```
public Mahasiswa14 peek() {
    if (!isEmpty()) {
        return stack[top];
    } else {
        System.out.println(x:"Stack kosong!
        return null;
    }
}
```

7. Method print() dengan keluaran void, digunakan untuk menampilkan daftar inputan yang telah diisikan di object pada stack dengan indeks yang sesuai, menggunakan loop for:

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

C. Class Main

1. Class MahasiswaDemo14: dalam fungsi main, menginstansiasi object stack dari class StackTugasMahasiswa untuk menginisialisasi kapasitas stack sebanyak 5, menambahkan variabel scanner, dan variabel pilih untuk pengguna memilih menu yang disediakan menggunakan perulangan do-while:

```
package Jobsheet9;
import java.util.Scanner;
public class MahasiswaDemo14 {
    Run | Debug
    public static void main(String[] args) {
        StackTugasMahasiswa14 stack = new StackTugasMahasiswa14(size:5);
        Scanner scan = 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 = scan.nextInt(); scan.ne
```

2. Menggunakan switch-case untuk mengalamatkan tujuan yang sesuai dengan pilihan menu yang dipilih:
   A. Pada Case 1: Tahap mengisikan nilai untuk atribut yang pada konstruktor parameter melalui array of object Mhs dari class Mahasiswa14, dan melakukan push untuk menyimpan object tersebut pada stack.

```
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();
        Mahasiswa14 mhs = new Mahasiswa14(nim, nama, kelas);
        stack.push(mhs);
        System.out.printf(format:"Tugas %s berhasil dikumpulkan\n", mhs.
    break;
```

   B. Case 2: Melakukan pop pada stack tugas dan menilai tugas pada objek yang sesuai top yang telah di pop menggunakan method tugasDinilai dari class Mahasiswa14:

```
case 2 :
    Mahasiswa14 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
        dinilai.tugasDinilai(nilai);
        System.out.printf(format:"Nilai Tugas %s adalah %d\n", dinilai.na
    }
break;
```

   C. Case 3: Peek pada isi stack tugas di posisi teratas untuk mengkonfirmasi mahasiswa yang telah mengumpulkan tugas terakhir kali:

```
case 3 :
    Mahasiswa14 lihat = stack.peek();
    if (lihat != null) {
        System.out.println("Tugas terakhir dikumpulkan oleh " + lihat.nam
    }
break;
```

   D. Case 4: Print seluruh daftar tugas pada stack dan melakukan format yang menyesuaikan output dari method tersebut.

```
case 4 :
    System.out.println(x:"Daftar semua tugas");
    System.out.println(x:"Nama\tNIM\tKelas"); stack.print();
break;
```

3. Commit pada github:

Verifikasi Percobaan

❌ Hasil tidak sesuai dengan yang ada di jobsheet

```
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
```

1

```
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

|| Pertanyaan Percobaan 1

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

   Jawaban:

```
(int i = top; i >= 0; i--) {
```

Bagian ini yang perlu diperbaiki, inisialisasi loop dimulai dari top untuk mencetak object dari stack paling atas menggunakan decrement, dan kondisi untuk mencetak hingga i mencapai -1 untuk menghentikan loop.

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

   Jawaban:

```
StackTugasMahasiswa14 stack = new StackTugasMahasiswa14(size:5);
```

Potongan kode ini dari class MahasiswaDemo14.

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

   Jawaban:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index 5 out of boun
ds for length 5
```

Untuk menghindari stack overflow yang mana push yang telah dilakukan berakhir tak masuk ke stack karena stack sudah penuh. Jika kondisinya dihapus, saat stack telah penuh, akan muncul error pada kode program. Sehingga program akan langsung berhenti.

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!

   Jawaban:

   Menu 5:

```
:"5. Lihat Mahasiswa yang mengumpulkan tugas pertama kali");
```

   Switch-case:

```
case 5 :
    Mahasiswa14 test1 = stack.reversePeek();
    if (test1 != null) {
        System.out.println("Tugas pertama kali dikumpulkan oleh " + test1.nama);
    }
break;
```

Method reversePeek():

```
public Mahasiswa14 reversePeek() {
    if (!isEmpty()) {
        return stack[0];
    } else {
        System.out.println(x:"Stack kosong! Tidak ada tugas yang dikumpulkan");
        return null;
    }
}
```

Hasil:

```
Pilih: 5
Tugas pertama kali dikumpulkan oleh A
```

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

Jawaban:

Menu 6:

```
"6. Total Mahasiswa yang telah mengumpulkan tugas");
```

Switch-case:

```
case 6:
    stack.totalPengumpulanSaatIni();
break;
default :
    System.out.println(x:"Pilihan tidak valid.");
break;
```

Method totalPengumpulanSaatIni():

```
public void totalPengumpulanSaatIni() {
    System.out.println("Total tugas yang telah dikumpulkan saat ini ialah: " + (top+1));
}
```

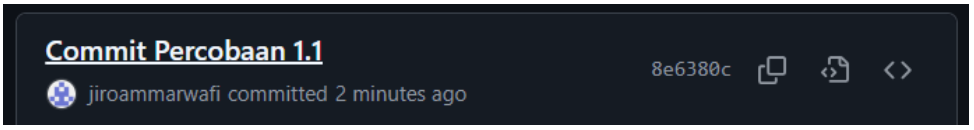
Hasil:

```
Pilih: 6
Total tugas yang telah dikumpulkan saat ini ialah: 2
```

Dengan bertambahnya 2 menu, loop while juga perlu diupdate:

```
} while (pilih >=1 && pilih <= 6);
```

6. **Commit dan push kode program ke Github**



2. Percobaan 2: Konversi Nilai Tugas ke Biner

1. Method konversiDesimalKeBiner dengan parameter nilai, dibuat pada class StackTugasMahasiswa14. Menggunakan array of object dari class StackKonvesi14 dengna nama objectnya yaitu stack :

```
public String konversiDesimalKeBiner(int nilai) {
    StackKonversi14 stack = new StackKonversi14();

    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;
}
```

2. Class StackKonversi14, didalamnya diberikan atribut stack serta method isEmpty, isFull, push, dan pull untuk operasi stack.

```
package Jobsheet9;
public class StackKonversi14 {
    int[] tumpukanBiner;
    int size, top;

    public StackKonversi14() {
        this.size = 32;
        tumpukanBiner = new int[size];
        top = -1;
    }
```

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

```
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;
    }
}
```

3. Menambahkan baris perintah berikut pada Class MahasiswaDemo14 pada Menu ke 2, menggunakan inputan untuk menilai tugas untuk di konversi menjadi bilangan biner:

```
String biner = stack.konversiDesimalKeBiner(nilai);
System.out.println("Nilai Biner Tugas: " + biner);
```

Verifikasi Percobaan

✔️ Hasil sesuai dengan yang ada di jobsheet

```
Menu:
1. Mengumpulkan Tugas
2. Menilai Tugas
3. Melihat Tugas Teratas
4. Melihat Daftar Tugas
5. Lihat Mahasiswa yang mengumpulkan tugas pertama kali
6. Total Mahasiswa yang telah mengumpulkan tugas
Pilih: 2
Menilai tugas dari Tika
Masukkan nilai (0-100): 87
Nilai Tugas Tika adalah 87
Nilai Biner Tugas: 1010111
```

4. Commit and Push Github:

|| Pertanyaan Percobaan 2

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

   Jawaban:

Saat method dipanggil, inputan nilai yang telah dilakukan sebelumnya akan dibawa ke method ini untuk di konversi menjadi bilangan biner.

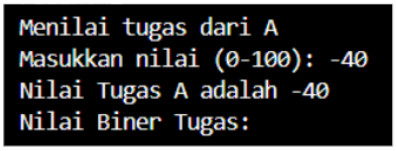
Masuk dalam method:

A. Loop Pertama (nilai > 0): dilakukan operasi modulus 2 pada variabel nilai saat ini, dan menyimpan output operasi tersebut dalam variabel sisa. Variabel sisa akan di push ke dalam stack melalui method pada class StackKonversi14. Setelah itu, dilakukan pembagian 2 pada var. nilai karena nilai sisa telah ditemukan. Proses Loop/Konversi ini akan berlanjut, dan berhenti apabila menyentuh nilai = 0.
B. Loop kedua (!stack.isEmpty): Sebelumnya telah dibuat variabel untuk menampung nilai biner tersebut yaitu biner (String). Dalam proses ini, nilai pada stack sebelumnya akan dikeluarkan dari posisi teratas hingga bawah menggunakan method pop pada class StackKonversi14 dan menyusun angka yang telah di pop ke barisan string yaitu biner. Loop berlanjut hingga seluruh isi stack telah dikeluarkan dan menjadi kosong.

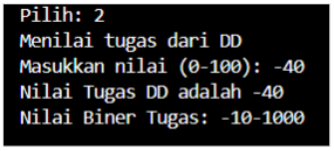
Demikian, setelah proses diatas. String biner menjadi nilai kembalian yang didalamnya terdapat angka biner yang sesuai dengan angka nilai tersebut.

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

   Jawaban:
   Perbandingan keduanya:



nilai > 0



nilai != 0

Algoritma yang dijalankan pada method tersebut hanyalah untuk konversi angka positif saja, sehingga apabila dimasukkan angka negatif, angka biner yang ditampilkan menjadi tidak sesuai dengan nilai biner yang sesungguhnya.

40 = 101000
- 40 = 010111

Apabila algoritma ini dipublikasikan, algoritma tak akan berfungsi dengan tepat sasaran, karena konversi hanya bisa dilakukan pada bilangan positif. Kesimpulannya, angka bulat negatif juga bisa diberikan apabila (kode !=0) yang berbeda dari tujuan awalnya yakni angka 0-100 yang merupakan angka bulat positif.

3. Latihan Praktikum

Demonstrasi Praktikum:

A. Menginputkan 3 surat izin;

B. Cek surat izin siapa yang terakhir dikumpulkan dan memverifikasi surat;

```
Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 1
Nama Mahasiswa: Jiro
Kelas: 1E
Keterangan Izin < S (Sakit) / I (Izin) >: S
Durasi Jam Pelajaran: 3
Surat Jiro berhasil dikumpulkan

Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 1
Nama Mahasiswa: Rika
Kelas: 1C
Keterangan Izin < S (Sakit) / I (Izin) >: I
Durasi Jam Pelajaran: 4
Surat Rika berhasil dikumpulkan

Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 1
Nama Mahasiswa: Maru
Kelas: 1A
Keterangan Izin < S (Sakit) / I (Izin) >: S
Durasi Jam Pelajaran: 1
Surat Maru berhasil dikumpulkan
```

A

```
Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 3
Tugas terakhir dikumpulkan oleh Maru

Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 2
Surat atas nama Maru telah di-approve


Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 3
Tugas terakhir dikumpulkan oleh Rika
```

B

C. Mencari surat yang sesuai dengan nama mahasiswa tertentu dan menampilkan informasi surat izin, dan exit program;

```
Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 4
Masukkan Nama Mahasiswa terkait: Maru
Tidak ada surat izin atas nama Maru

Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 4
Masukkan Nama Mahasiswa terkait: Rika
Surat ini ditemukan !
ID-Surat   Nama Mahasiswa      Kelas      Jenis Izin      Durasi
3          Rika               1C         I - Izin        4

Menu:
1. Buat Surat Izin
2. Proses Surat Izin
3. Lihat Surat Izin Terakhir
4. Cari Surat Mahasiswa
5. Exit Program
Pilih: 5
Program berakhir
PS C:\Users\luvma\Downloads\KULIAH\Semester 2\Praktikum-ASD>
```