NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

Percobaan 1: Penyimpanan Tumpukan Barang dalam Gudang

```java
package P9;

import java.util.Scanner;

public class Utama27 {
    public static void main(String[] args) {
        Scanner roy27 = new Scanner(System.in);
        Gudang27 gudang = new Gudang27(7);
        boolean isBreak = false;
        while (!isBreak) {
            System.out.println("\nMenu:");
            System.out.println("1. Tambah barang");
            System.out.println("2. Ambil barang");
            System.out.println("3. Tampilkan tumpukan barang");
            System.out.println("4. Lihat barang teratas");
            System.out.println("5. Keluar");
            System.out.print("Pilih Operasi: ");
            int pilihan = roy27.nextInt();
            roy27.nextLine();

            switch (pilihan) {
                case 1:
                    System.out.print("Masukkan kode barang: ");
                    int kode = roy27.nextInt();
                    roy27.nextLine();
                    System.out.print("Masukkan nama barang: ");
                    String nama = roy27.nextLine();
                    System.out.print("Masukkan nama kategori: ");
                    String kategori = roy27.nextLine();
                    Barang27 barangBaru = new Barang27(kode, nama, kategori);
                    gudang.tambahBarang(barangBaru);
                    break;
                case 2:
                    gudang.ambilbarang();
                    break;
                case 3:
                    gudang.tampilkanBarang();
                    break;
                case 4:
                    gudang.lihatBarangTeratas();
                    break;
                case 5:
                    isBreak = true;
                    break;
                default:
                    System.out.println("Pilihan tidak valid. Silahkan coba lagi");
                    break;
            }
        }
    }
}

```

```java
package P9;

public class Barang27 {
    int kode;
    String nama;
    String kategori;

    Barang27(int kode, String nama, String kategori) {
        this.kode = kode;
        this.nama = nama;
        this.kategori = kategori;
    }
}
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

```java
package P9;

public class Gudang27 {
    Barang27 tumpukan[];
    int size;
    int top;

    Gudang27(int kapasitas) {
        this.size = kapasitas;
        tumpukan = new Barang27[kapasitas];
        top = -1;

    }

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

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

    void tambahBarang(Barang27 brg) {
        if (!cekPenuh()) {
            top++;
            tumpukan[top] = brg;
            System.out.println("Barang " + brg.nama + " berhasil ditambahkan ke Gudang ");
        } else {
            System.out.println("Gagal! Tumpukan barang di Gudang sudah penuh");
        }
    }

    Barang27 ambilbarang() {
        if (!cekKosong()) {
            Barang27 delete = tumpukan[top];
            top--;
            System.out.println("Barang " + delete.nama + " diambil dari gudang");
            return delete;
        } else {
            System.out.println("Tumpukan barang kosong");
            return null;
        }
    }

    Barang27 lihatBarangTeratas() {
        if (!cekKosong()) {
            Barang27 barangTeratas = tumpukan[top];
            System.out.println("Barang teratas: " + barangTeratas.nama);
            return barangTeratas;
        } else {
            System.out.println("Tumpukan barang kosong");
            return null;
        }
    }

    void tampilkanBarang() {
        if (!cekKosong()) {
            System.out.println("Rincian tumpukan barang di Gudang");
            for (int i = 0; i <= top; i++) {
                System.out.printf("Kode %d: %s (Kategori %s)\n", tumpukan[i].kode, tumpukan[i].nama,
                        tumpukan[i].kategori);
            }
        } else {
            System.out.println("Tumpukan barang kosong");
        }
    }

}

```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 21
Masukkan nama barang: Majalah
Masukkan nama kategori: Buku
Barang Majalah berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 26
Masukkan nama barang: Jaket
Masukkan nama kategori: Pakaian
Barang Jaket berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 2
Barang Jaket diambil dari gudang
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 3
Rincian tumpukan barang di Gudang
Kode 21: Majalah (Kategori Buku)
Kode 33: Pizza (Kategori Makanan)
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 33
Masukkan nama barang: Pizza
Masukkan nama kategori: Makanan
Barang Pizza berhasil ditambahkan ke Gudang
```

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

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

Jawab:

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 21
Masukkan nama barang: Majalah
Masukkan nama kategori: Buku
Barang Majalah berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 26
Masukkan nama barang: Jaket
Masukkan nama kategori: Pakaian
Barang Jaket berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 2
Barang Jaket diambil dari gudang
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 1
Masukkan kode barang: 33
Masukkan nama barang: Pizza
Masukkan nama kategori: Makanan
Barang Pizza berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 3
Rincian tumpukan barang di Gudang
Kode 33: Pizza (Kategori Makanan)
Kode 21: Majalah (Kategori Buku)

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 4
```

```java

void tampilkanBarang() {
    if (!cekKosong()) {
        System.out.println("Rincian tumpukan barang di Gudang");
        for (int i = top; i >= 0; i--) {
            System.out.printf("Kode %d: %s (Kategori %s)\n", tumpukan[i].kode, tumpukan[i].nama,
                    tumpukan[i].kategori);
        }
    } else {
        System.out.println("Tumpukan barang kosong");
    }
}
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

2. Berapa banyak data barang yang dapat ditampung di dalam tumpukan? Tunjukkan potongan kode programnya!

   Jawab:

   Banyak data barang yang dapat ditampung adalah 7.

```
Gudang27 gudang = new Gudang27(kapasitas:7);
```

3. Mengapa perlu pengecekan kondisi !cekKosong() pada method tampilkanBarang? Kalau kondisi tersebut dihapus, apa dampaknya?

   Jawab:

   Diperlukan pengecekan kondisi !cekKosong() karena jika barang di gudang tidak ada data, program akan tetap berjalan akan tetapi tidak menampilkan barang apapun. Jika terdapat pengecekan lalu ketika ada barang yang kosong akan ada peringatan.

   Output kondisi dihapus:

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi: 3
Rincian tumpukan barang di Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Keluar
Pilih Operasi:
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

4. Modifikasi kode program pada class Utama sehingga pengguna juga dapat memilih operasi lihat barang teratas, serta dapat secara bebas menentukan kapasitas gudang!

   Jawab:

```
System.out.print(s:"Tentukan kapasitas gudang");
int kapasitas = roy27.nextInt();
Gudang27 gudang = new Gudang27(kapasitas);
```

```
Tentukan kapasitas gudang: 7

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi:
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 4
Barang teratas: Pizza
```

```
case 4:
    gudang.lihatBarangTeratas();
    break;
```

```
System.out.println(x:"4. Lihat barang teratas");
System.out.println(x:"5. Keluar");
```

5. Commit dan push kode program ke Github

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

2.2 Percobaan 2: Konversi Kode Barang ke Biner

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 13
Masukkan nama barang: Setrika
Masukkan nama kategori: Elektronik
Barang Setrika berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 2
Barang Setrika diambil dari gudang
Kode unik dalam biner: 1101
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

```java
package P10;

public class StackKonversi {
    int size;
    int[] tumpukanBiner;
    int 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("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;
        }
    }
}

```

```java
Barang27 ambilbarang() {
    if (!cekKosong()) {
        Barang27 delete = tumpukan[top];
        top--;
        System.out.println("Barang " + delete.nama + " diambil dari gudang");
        System.out.println("Kode unik dalam biner: " + konversiDesimalKeBiner(delete.kode));
        return delete;
    } else {
        System.out.println("Tumpukan barang kosong");
        return null;
    }
}

```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

2.2.3 Pertanyaan

1. Pada method konversiDesimalKeBiner, ubah kondisi perulangan menjadi while (kode != 0), bagaimana hasilnya? Jelaskan alasannya!
   Jawab:
   Jika kode barang yang diinput 0 atau kurang dari 0 maka kode barang tersebut tidak akan dikonversi ke biner, hasilnya akan seperti ini:

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Lihat barang teratas
5. Keluar
Pilih Operasi: 1
Masukkan kode barang: 0
Masukkan nama barang: df
Masukkan nama kategori: df
Barang df berhasil ditambahkan ke Gudang
```

2. Jelaskan alur kerja dari method konversiDesimalKeBiner!
   Jawab:
   1. Inisialisasi stack yang digunakan untuk menyimpan data kode biner
   2. Pengecekan kondisi pada looping while apakah kode barang > 0
   3. Setelah loop selesai (saat kode menjadi 0), metode melanjutkan ke loop while kedua untuk membangun string biner dari digit-digit yang tersimpan di stack
   4. String biner yang telah lengkap dengan representasi biner dari bilangan desimal dikembalikan sebagai hasil dari metode ini.

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

2.3 Percobaan 3: Konversi Notasi Infix ke Postfix

```java
package P10;

public class Postfix27 {
    int n;
    int top;
    char[] stack;

    public Postfix27(int total) {
        n = total;
        top = -1;
        stack = new char[n];
        push('(');
    }

    public void push(char c) {
        top++;
        stack[top] = c;

    }

    public char pop() {
        char item = stack[top];
        top--;
        return item;
    }

    public boolean isOperand(char c) {
        if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z') || (c >= '0' && c <= '9') || c == ' ' || c == '.') {
            return true;
        } else {
            return false;
        }
    }

    public boolean isOperator(char c) {
        if (c == '^' || c == '%' || c == '/' || c == '*' || c == '-' || c == '+') {
            return true;
        } else {
            return false;
        }
    }

    public int derajat(char c) {
        switch (c) {
            case '^':

                return 3;

            case '%':

                return 2;

            case '/':

                return 2;

            case '*':

                return 2;

            case '-':

                return 1;

            case '+':

                return 1;

            default:
                return 0;
        }
    }

    public String konversi(String Q) {
        String P = "";
        char c;
        for (int i = 0; i < n; i++) {
            c = Q.charAt(i);
            if (isOperand(c)) {
                P = P + c;
            }
            if (c == '(') {
                push(c);
            }
            if (c == ')') {
                while (stack[top] != '(') {
                    P = P + pop();
                }
                pop();
            }
            if (isOperator(c)) {
                while (derajat(stack[top]) >= derajat(c)) {
                    P = P + pop();
                }
                push(c);
            }
        }
        return P;
    }
}
```

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

```java
package P10;

import java.util.Scanner;;

public class PostfixMain27 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        String P, Q;
        System.out.println("Masukkan ekspresi matematika (infix):");
        Q = sc.nextLine();
        Q = Q.trim();
        Q = Q + ")";
        int total = Q.length();

        Postfix27 post = new Postfix27(total);
        P = post.konversi(Q);
        System.out.println("Postfix: " + P);
    }
}

```

```
Masukkan ekspresi matematika (infix):
a+b*(c+d-e)/f
Postfix: abcd+e-*f/+
```

Pertanyaan

1. Pada method derajat, mengapa return value beberapa case bernilai sama? Apabila return value diubah dengan nilai berbeda-beda setiap case-nya, apa yang terjadi?

   Jawab:
   Derajat operator dalam metode derajat return case disesuaikan untuk memastikan bahwa aturan operasi matematika mempertahankan urutan yang benar dari operasi sesuai dengan aturan matematika dasar.

   Jika mengubah nilai return dari metode derajat sehingga setiap operator memiliki nilai unik yang berbeda akan mengubah cara ekspresi dikonversi menjadi format postfix.

2. Jelaskan alur kerja method konversi!

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

Jawab:

Inisialisasi string P untuk menyimpan hasil ekspresi dalam bentuk postfix.

Iterasi dilakukan pada setiap karakter dalam string masukan Q.

Pengolahan setiap karakter:

Untuk setiap karakter c dalam string Q:

Lalu pengecekan kondisi apakah argumen operand,operator, buka kurung atau tutup kurung.

Jika operand => karakter tersebut langsung ditambahkan ke dalam P

Jika ‘(‘ => karakter tersebut dimasukkan ke dalam tumpukan/stack

Jika ‘)’=> karakter di tumpukan akan dipop dan ditambahkan ke P sampai ditemukan kurung buka

Jika operator => metode ini membandingkan prioritas operator tersebut dengan operator di puncak tumpukan

Operand (karakter dan angka):

Jika c adalah operand, karakter tersebut langsung ditambahkan ke dalam P.

Setelah menambahkan operand, keadaan string P dicetak, yang membantu melacak pembangunan string postfix langkah demi langkah.

*Operator (+, -, , /, %, ^):

Jika c adalah operator, metode ini membandingkan prioritas operator tersebut dengan operator di puncak tumpukan:

Loop berakhir setelah semua karakter dari Q telah diproses. Namun, biasanya ada fase pembersihan di akhir dimana semua operator yang tersisa dalam tumpukan dipop dan ditambahkan ke P untuk melengkapi ekspresi. Bagian ini tampaknya hilang dalam potongan kode yang diberikan dan penting untuk memastikan semua operator termasuk dalam output.

NAMA : Roy Wijaya
NIM : 2341720120
KELAS : 1-G
MATERI : Stack (Jobsheet 7)

3. Pada method konversi, apa fungsi dari potongan kode berikut?

```
c = Q.charAt(i);
```

Jawab:

Kode tersebut digunakan untuk mengambil karakter pada indeks ke-i dari string Q.

LINK GITHUB: https://github.com/RoyW12/AlgoritmaStrukturData_1G_28