PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

JOBSHEET 7

NAMA : DIMAS ADI BAYU SAMUDRA

KELAS : 1A

NO. ABSEN : 08

NIM : 2341720169

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

PRAKTIKUM

Percobaan 1

Code Barang08

```
package jobsheet7;

public class Barang08 {
    int kode;
    String nama, kategori;

    Barang08(int kode, String nama, String kategori) {
        this.kode = kode;
        this.nama = nama;
        this.kategori = kategori;
    }
}
```

Code Gudang08

```
package jobsheet7;

public class Gudang08 {
    Barang08[] tumpukan;
    int size, top;

    public Gudang08(int kapasitas) {
        size = kapasitas;
        tumpukan = new Barang08[size];
        top = -1;
    }

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

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

    public void tambahBarang(Barang08 brg) {
        if (!cekPenuh()) {
            top++;
            tumpukan[top] = brg;
            System.out.println("Barang " +brg.nama+ " Berhasil ditambahkan
ke Gudang");
```

```
        }else {
            System.out.println("Gagal! Barang di gudang sudah penuh");


        }
    }

    public Barang08 ambilBarang() {
        if (!cekKosong()) {
            Barang08 delete = tumpukan[top];
            top--;
            System.out.println("Barang " +delete.nama+ " diambil dari
gudang");
            return delete;
        }else {
            System.out.println("Tumpukan barang kosong");
            return null;
        }
    }

    public Barang08 lihatBarangTeratas() {
        if (!cekKosong()) {
            Barang08 barangTeratas = tumpukan[top];
            System.out.println("Barang teratas: " +barangTeratas.nama);
            return barangTeratas;
        } else {
            System.out.println("Tumpukan barang kosong");
            return null;
        }
    }

    public void tampilkanBarang() {
        if (!cekKosong()) {
            System.out.println("Rincian tumpukan barang di gudang:");
            for (int i = top; i >= 0; i--) {
                for (int j = 0; j < top; j++) {
                    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");
        }
    }
}
```

Code Utama08

```
package jobsheet7;

import java.util.Scanner;

public class Utama08 {
    public static void main(String[] args) {
```

```
        Gudang08 gudang = new Gudang08(7);
        Scanner sc = new Scanner(System.in);

        while (true) {
            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. Keluar");
            System.out.print("Pilih operasi: ");
            int pilihan = sc.nextInt();
            sc.nextLine();

            switch (pilihan) {
                case 1:
                    System.out.print("Masukkan kode barang: ");
                    int kode = sc.nextInt();
                    System.out.print("Masukkan nama barang: ");
                    String nama = sc.next();
                    System.out.print("Masukkan nama kategori: ");
                    String kategori = sc.next();
                    Barang08 barangBaru = new Barang08(kode, nama,
kategori);
                    gudang.tambahBarang(barangBaru);
                    break;
                case 2:
                    gudang.ambilBarang();
                    break;
                case 3:
                    gudang.tampilkanBarang();
                    break;
                case 4:
                    continue;
                default:
                    System.out.println("Pilihan tidak valid! silahkan coba
lagi");
            }
        }
    }
}
```

Hasil Program

```
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)
```

Pertanyaan

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

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

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

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

5. Commit dan push kode program ke Github

Jawaban

1. Agar output / hasil run sama dengan yang ada pada jobsheet 7 maka harus di lakukan perubahan pada kode program yaitu bagian method lihatBarangTeratas pada class Gudang08. Perubahan yang dilakukan pada kode program pada bagian pemilihan if(!isEmpty()) diubah menjadi if(!cekKosong()).

2. Data maksimal yang dapat di tamping di dalam tumpukan adalah 7

```
Gudang08 gudang = new Gudang08(kapasitas:7);
```

3. Pengecekan tersebut dilakukan untuk menentukan kondisi apakah ada barang yang dapat ditampilkan. Jika bagian tersebut dihapus, maka program akan menampilkan nilai “null”

4. kode yang telah dimodifikasi

```
package jobsheet7;

import java.util.Scanner;

public class Utama08 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Masukkan kapasitas gudang: ");
        int kapasitas = sc.nextInt();
        Gudang08 gudang = new Gudang08(kapasitas);

        while (true) {
            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. Tampilkan barang teratas");
            System.out.println("5. Keluar");
```

```
            System.out.print("Pilih operasi: ");
            int pilihan = sc.nextInt();
            sc.nextLine();

            switch (pilihan) {
                case 1:
                    System.out.print("Masukkan kode barang: ");
                    int kode = sc.nextInt();
                    System.out.print("Masukkan nama barang: ");
                    String nama = sc.next();
                    System.out.print("Masukkan nama kategori: ");
                    String kategori = sc.next();
                    Barang08 barangBaru = new Barang08(kode, nama,
kategori);
                    gudang.tambahBarang(barangBaru);
                    break;
                case 2:
                    gudang.ambilBarang();
                    break;
                case 3:
                    gudang.tampilkanBarang();
                    break;
                case 4:
                    gudang.lihatBarangTeratas();
                    break;
                case 5:
                    break;
                default:
                    System.out.println("Pilihan tidak valid! silahkan coba
lagi");
                    break;
            }
        }
    }
}
```

Hasil

```
Masukkan kapasitas gudang: 5

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih operasi: 1
Masukkan kode barang: 22
Masukkan nama barang: Mouse
Masukkan nama kategori: Komputer
Barang Mouse Berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih operasi: 1
Masukkan kode barang: 11
Masukkan nama barang: Asus
Masukkan nama kategori: Laptop
Barang Asus Berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih operasi: 4
Barang teratas: Asus
```

Percobaan 2

Kode StackKonversi08

```
public class StackKonversi08 {

    int size, top;
    int[] tumpukanBiner;

    public StackKonversi08() {
        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;
        }
    }
}
```

Kode tambahan

```
public Barang08 ambilBarang() {
        if (!cekKosong()) {
            Barang08 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;
        }
    }

public String konversiDesimalKeBiner(int kode) {
        StackKonversi08 stack = new StackKonversi08();
        while (kode > 0) {
            int sisa = kode % 2;
            stack.push(sisa);
            kode = kode / 2;
        }
        String biner = new String();
        while (!stack.isEmpty()) {
            biner += stack.pop();
        }
        return biner;
    }
```

Hasil

```
Masukkan kapasitas gudang: 1

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih operasi: 1
Masukkan kode barang: 13
Masukkan nama barang: asd
Masukkan nama kategori: asd
Barang asd berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Keluar
Pilih operasi: 2
Barang asd diambil dari Gudang.
Kode unik dalam biner:  1101
```

Pertanyaan

1. Pada method konversiDesimalKeBiner, ubah kondisi perulangan menjadi while (kode != 0),

bagaimana hasilnya? Jelaskan alasannya!

2. Jelaskan alur kerja dari method konversiDesimalKeBiner!

Jawaban

1. Saat kode program diatas diubah, tidak ada perubahan yang terjadi pada output yang dihasilkan. Hal itu di karenakan > 0 dan != 0 mempunyai prinsip yang sama, yaitu method akan terus melakukan perulangan selagi kode nilainya bukan 0.

2. Pertama, dibuat sebuah objek StackKonversi08 kosong yang diberi nama stack. Tumpukan ini akan digunakan untuk menyimpan angka biner nantinya.Program kemudian memasuki perulangan yang akan terus berjalan selama nilai kode (angka desimal) tidak sama dengan 0.

Di dalam perulangan:

Hitung sisa (sisa) dari pembagian kode saat ini dengan angka 2.Angka sisa (sisa) ini merepresentasikan angka biner (0 atau 1). Kita masukkan angka sisa ini ke dalam stack menggunakan operasi "push".kode kemudian dibagi 2 untuk mendapatkan nilai baru pada iterasi selanjutnya.Setelah perulangan selesai, kita buat variabel String kosong bernama biner untuk menyimpan representasi binernya.Program kemudian memasuki perulangan lagi, yang terus berjalan selama stack belum kosong.

Di dalam perulangan: Keluarkan (operasi "pop") elemen teratas (sisa) dari stack.Tambahkan (+=) angka sisa (sisa) tersebut ke akhir string biner.

Setelah stack kosong dan semua angka biner telah diproses, string biner akan berisi representasi biner lengkap dari angka desimal awal (kode).dan yang terakhir, metode ini akan mengembalikan nilai biner sebagai hasil akhir.

Pecobaan 3

Kode PostFix08

```
public class PostFix08 {

    int n, top;
    char[] stack;

    public PostFix08(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;
    }
    }
```

Kode PostFixMain08

```
import java.util.Scanner;

public class PostFixMain08 {
    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();
        PostFix08 post = new PostFix08(total);
        P = post.konversi(Q);
        System.out.println("Postfix: " + P);
    }
}
```

Hasil

```
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?

2. Jelaskan alur kerja method konversi!

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

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

Jawaban

1. Dalam method derajat, beberapa nilai return memiliki nilai yang sama karena mereka mewakili tingkat precedence dari operator yang terkait. Precedence menentukan urutan operasi mana yang dievaluasi terlebih dahulu dalam sebuah ekspresi. Operator dengan precedence lebih tinggi dievaluasi sebelum operator dengan precedence lebih rendah.

2. Inisialisasi:

Sebuah String kosong P dibuat untuk menyimpan ekspresi postfix yang akan dihasilkan. Variabel karakter c akan digunakan untuk menampung karakter individual dari String Q. Asumsikan n adalah panjang dari String Q (jumlah karakter). Iterasi Melalui String Q: Program menggunakan perulangan for untuk iterasi melalui setiap karakter dalam Q.

Di dalam perulangan: Karakter pada indeks i dari Q diambil menggunakan charAt(i) dan disimpan di c. Pemrosesan Karakter: Operator atau Operan:Jika c adalah sebuah operan (angka), maka karakter tersebut ditambahkan langsung ke String P menggunakan operator penambahan (+). Kurung Buka ('('):Jika c adalah kurung buka '(', maka karakter tersebut dipush ke dalam stack (tumpukan). Stack ini digunakan untuk menyimpan operator sementara. Kurung Tutup (')'): Jika c adalah kurung tutup ')', maka program akan terus mengeluarkan (pop) elemen dari stack dan menambahkannya ke String P selama elemen teratas stack bukan kurung buka '('. Hal ini dilakukan untuk memproses operator terlebih dahulu sebelum operand yang mereka kaitkan. Setelah kurung buka '(' dikeluarkan dari stack, proses pop dihentikan. *Operator (mis. +, -, , /): Jika c adalah sebuah operator, program akan melakukan perulangan while selama operator teratas stack (diperiksa menggunakan stack[top]) memiliki tingkat kepentingan (derajat) yang lebih besar atau sama dengan operator c. Di dalam perulangan while: Elemen teratas stack dikeluarkan (pop) dan ditambahkan ke String P. Setelah perulangan while, operator c kemudian dipush ke

dalam stack untuk digunakan nanti dalam pemrosesan ekspresi. Mengembalikan Hasil: Setelah iterasi selesai, String P yang berisi ekspresi postfix dikembalikan sebagai hasil dari method konversi.

3. Fungsi dari kode program diatas adalah untuk mengubah tiap karakter dari notasi String ke dalam bentuk char yang kemudian disimpan ke dalam variabel c. Kode diatas dituliskan di dalam perulangan for sehingga tiap karakter pada notasi String yang dimasukkan dapat terkonversi semua.

Latihan Praktikum

Kode tambahan Gudang08

```
public Barang08 lihatBarangTerbawah() {
        if (!cekKosong()) {
            Barang08 barangTerbawah = tumpukan[0];
            System.out.println("Barang terbawah: "+ barangTerbawah.nama);
            return barangTerbawah;
        } else {
            System.out.println("Tumpukan barang kosong.");
            return null;
        }
    }

public Barang08 cariBarang(String search) {
        if (!cekKosong()) {
            for (int i = top; i >= 0 ; i--) {
                String kode = String.valueOf(tumpukan[i].kode);
                if (tumpukan[i].nama.equalsIgnoreCase(search) ||
kode.equals(search)) {
                    System.out.println("Barang ditemukan: ");
                    System.out.println("Kode:\t"+ tumpukan[i].kode);
                    System.out.println("Nama:\t"+ tumpukan[i].nama);
                    System.out.println("Kategori:\t"+ tumpukan[i].kategori);
                    return tumpukan[i];
                }
            }
            System.out.println("Barang tidak ditemukan.");
        } else {
            System.out.println("Tumpukan barang kosong.");
        }
        return null;
    }
```

Kode tambahan UtamaMain08

```
while (true) {
            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. Tampilkan barang teratas");
            System.out.println("5. Tampilkan barang terbawah");
            System.out.println("6. Cari Barang");
            System.out.println("7. Keluar");
```

```
            System.out.print("Pilih operasi: ");
            int pilihan = sc.nextInt();

            switch (pilihan) {
                case 1:
                    System.out.print("Masukkan kode barang: ");
                    int kode = sc.nextInt();
                    sc.nextLine();
                    System.out.print("Masukkan nama barang: ");
                    String nama = sc.nextLine();
                    System.out.print("Masukkan nama kategori: ");
                    String kategori = sc.nextLine();
                    Barang08 barangBaru = new Barang08(kode, nama,
kategori);
                    gudang.tambahBarang(barangBaru);
                    break;
                case 2:
                    gudang.ambilBarang();
                    break;
                case 3:
                    gudang.tampilkanBarang();
                    break;
                case 4:
                    gudang.lihatBarangTeratas();
                    break;
                case 5:
                    gudang.lihatBarangTerbawah();
                    break;
                case 6:
                    sc.nextLine();
                    System.out.print("Masukkan nama / kode barang: ");
                    String search = sc.nextLine();
                    gudang.cariBarang(search);
                    break;
                case 7:
                    System.exit(1);
                    break;
                default:
                    System.out.println("Pilihan tidak valid. Silakan coba
lagi.");
                    break;
            }
        }
```

Hasil

```
Masukkan kapasitas gudang: 3

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Tampilkan barang terbawah
6. Cari Barang
7. Keluar
Pilih operasi: 1
Masukkan kode barang: 13
Masukkan nama barang: Kaos
Masukkan nama kategori: Pakaian
Barang Kaos berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Tampilkan barang terbawah
6. Cari Barang
7. Keluar
Pilih operasi: 1
Masukkan kode barang: 14
Masukkan nama barang: Hoodie
Masukkan nama kategori: Pakaian
Barang Hoodie berhasil ditambahkan ke Gudang

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Tampilkan barang terbawah
6. Cari Barang
7. Keluar
Pilih operasi: 1
Masukkan kode barang: 15
Masukkan nama barang: Kemeja
Masukkan nama kategori: Pakaian
Barang Kemeja berhasil ditambahkan ke Gudang
```

```
Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Tampilkan barang terbawah
6. Cari Barang
7. Keluar
Pilih operasi: 5
Barang terbawah: Kaos

Menu:
1. Tambah barang
2. Ambil barang
3. Tampilkan tumpukan barang
4. Tampilkan barang teratas
5. Tampilkan barang terbawah
6. Cari Barang
7. Keluar
Pilih operasi: 6
Masukkan nama / kode barang: 14
Barang ditemukan:
Kode:   14
Nama:   Hoodie
Kategori:       Pakaian
```