LAPORAN PRAKTIKUM
MATA KULIAH ALGORITMA DAN STRUKTUR DATA

Dosen Pengampu : Triana Fatmawati, S.T, M.T

PERTEMUAN - 7 - Searching

Nama : **M. Zidna Billah Faza**

NIM : **2341760030**

Prodi : **D-IV Sistem Informasi Bisnis**

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

Percobaan 1

1) Buat folder dengan nama Praktikum07. Buat file Stack.java.

2) Tulis kode untuk membuat atribut dan konstruktor pada class Stack sebagai berikut:

```
int data[];
int size;
int top;

public Stack_18(int size) {
    this.size = size;
    data = new int[size];
    top = -1;
}
```

3) Lalu tambahkan method isFull() dan isEmpty() pada class Stack sebagai berikut:

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

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

4) Tambahkan method push(int data) dan pop() sebagai berikut:

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

public void pop() {
    if (!isEmpty()) {
        int x = data[top];
        top--;
        System.out.println("Data yang dikeluarkan dari stack : " + x);
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

5) Tambahkan method peek() sebagai berikut:

```
public void peek() {
    System.out.println("Elemen teratas stack : " + data[top]);
}
```

6) Tambahkan method print() dan clear() sebagai berikut:

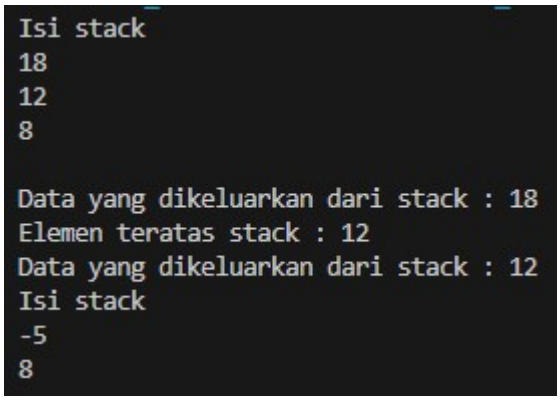
```
public void print() {
    System.out.println(x:"Isi stack");
    for (int i = top; i >= 0; i--) {
        System.out.println(data[i] + " ");
    }
    System.out.println(x:"");
}

public void clear() {
    if (!isEmpty()) {
        for (int i = top; i >= 0; i--) {
            top--;
        }
        System.out.println(x:"Stack sudah dikosongkan");
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

7) Buat file StackDemo.java untuk mengimplementasikan class StackDemo yang berisi fungsi main untuk membuat objek Stack dan mengoperasikan method-method pada class Stack.

```
public class StackDemo_18 {
    Run | Debug
    public static void main(String[] args) {
        Stack_18 stack = new Stack_18(size:10);
        stack.push(dt:8);
        stack.push(dt:12);
        stack.push(dt:18);
        stack.print();
        stack.pop();
        stack.peek();
        stack.pop();
        stack.push(-5);
        stack.print();
    }
}
```

8) Compile dan run class StackDemo dan verifikasi hasil percobaan



Pertanyaan Percobaan 1

1) Pada method pop(), mengapa diperlukan pemanggilan method isEmpty()? Apa yang terjadi jika tidak ada pemanggilan isEmpty()?

Untuk menjamin operasi yang aman pada struktur data dan menghindari pengecualian dalam metode pop(), maka metode isEmpty() harus dipanggil. Ketika isEmpty tidak dipanggil maka akan ada pengecualian.

2) Jelaskan perbedaan antara method peek() dengan method pop() pada class Stack.

- peek() : digunakan untuk melihat elemen teratas pada stack atau top tanpa menghapusnya
- pop() : digunakan untuk menghapus elemen teratas atau top

Percobaan 2

1) Buat class baru dengan nama Pakaian

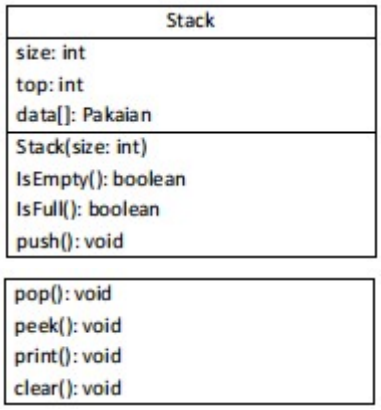
2) Tambahkan atribut-atribut Pakaian seperti pada Class Diagram Pakaian, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```
public class Pakaian_18 {

    String jenis, warna, merk, ukuran;
    double harga;

    Pakaian_18(String jenis, String warna, String merk, String ukuran, double harga) {
        this.jenis = jenis;
        this.warna = warna;
        this.merk = merk;
        this.ukuran = ukuran;
        this.harga = harga;
    }
}
```

3) Setelah membuat class Pakaian, selanjutnya perlu dibuat class Stack yang berisi atribut dan method sesuai diagram Class Stack berikut ini:



Keterangan: Tipe data pada variabell data menyesuaikan dengan data yang akan akan disimpan di dalam Stack. Pada praktikum ini, data yang akan disimpan merupakan array of object dari Pakaian, sehingga tipe data yang digunakan adalah Pakaian

4) Buat class baru dengan nama Stack. Kemudian tambahkan atribut dan konstruktor seperti gambar berikut ini.

```
public class StackPakaian_18 {

    int size;
    int top;
    Pakaian_18 data[];

    public StackPakaian_18(int size) {
        this.size = size;
        data = new Pakaian_18[size];
        top = -1;
    }
```

5) Buat method IsEmpty bertipe boolean yang digunakan untuk mengecek apakah stack kosong.

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

6) Buat method IsFull bertipe boolean yang digunakan untuk mengecek apakah stack sudah terisi penuh.

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

7) Buat method push bertipe void untuk menambahkan isi elemen stack dengan parameter pakaian yang berupa object pakaian

```
public void push(Pakaian_18 pakaian) {
    if (!isFull()) {
        top++;
        data[top] = pakaian;
    } else {
        System.out.println(x:"Isi stack penuh");
    }
}
```

8) Buat method Pop bertipe void untuk mengeluarkan isi elemen stack. Karena satu elemen stack terdiri dari beberapa informasi (jenis, warna, merk, ukuran, dan harga), maka ketika mencetak data juga perlu ditampilkan semua informasi tersebut

```
public void pop() {
    if (!isEmpty()) {
        Pakaian_18 x = data[top];
        top--;
        System.out
                .println("Data yang keluar : " + x.jenis + " " + x.warna + x.merk + " " + x.ukuran + " " + x.harga);
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

9) Buat method peek bertipe void untuk memeriksa elemen stack pada posisi paling atas.

```
public void peek() {
    System.out.println("Elemen teratas   : " + data[top].jenis + " " + data[top].warna + " " + data[top].merk + " "
            + data[top].ukuran + " " + data[top].harga);
}
```

10) Buat method print bertipe void untuk menampilkan seluruh elemen pada stack.

```
public void print() {
    System.out.println(x:"Isi stack");
    for (int i = top; i >= 0; i--) {
        System.out.println(data[i].jenis + " " + data[i].warna + " " + data[i].merk + " "
            + data[i].ukuran + " " + data[i].harga);
    }
    System.out.println(x:"");
}
```

11) Buat method clear bertipe void untuk menghapus seluruh isi stack.

```
public void clear() {
    if (!isEmpty()) {
        for (int i = top; i >= 0; i--) {

        }
        System.out.println(x:"Stack sudah dikosongkan");
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

12) Selanjutnya, buat class baru dengan nama StackMain. Buat fungsi main, kemudian lakukan instansiasi objek dari class Stack dengan nama stack dan nilai parameternya adalah 5.

```
StackPakaian_18 stack = new StackPakaian_18(size:5);
```

13) Deklarasikan Scanner dengan nama input18

```
Scanner input18 = new Scanner(System.in);
```

14) Tambahkan kode berikut ini untuk menerima input data Pakaian, kemudian semua informasi tersebut dimasukkan ke dalam stack

```
do {

    System.out.print(s:"Jenis     : ");
    String jenis = input18.nextLine();
    System.out.print(s:"Warna     : ");
    String warna = input18.nextLine();
    System.out.print(s:"Merk      : ");
    String merk = input18.nextLine();
    System.out.print(s:"Ukuran    : ");
    String ukuran = input18.nextLine();
    System.out.print(s:"Harga     : ");
    double harga = input18.nextDouble();

    Pakaian_18 pakaian = new Pakaian_18(jenis, warna, merk, ukuran, harga);

    System.out.print(s:"Apakah anda akan menambahkan data baru ke stack (y/n)?  : ");
    pilihan = input18.next().charAt(index:0);
    input18.nextLine();

    stack.push(pakaian);

} while (pilihan == 'y');
```

Catatan: sintaks sc.nextLine() sebelum sintaks st.push(p) digunakan untuk mengabaikan karakter new line

15) Lakukan pemanggilan method print, method pop, dan method peek dengan urutan sebagai berikut.

```
stack.print();
stack.pop();
stack.peek();
stack.print();
```

16) Compile dan jalankan class StackMain, kemudian amati hasilnya serta verifikasi hasil percobaan.

```
Jenis     : Kaos
Warna     : Nevada
Merk      : Hitam
Ukuran    : M
Harga     : 85000
Apakah anda akan menambahkan data baru ke stack (y/n)?  : y
Jenis     : Kemeja
Warna     : Putih
Merk      : Styvea
Ukuran    : XL
Harga     : 127000
Apakah anda akan menambahkan data baru ke stack (y/n)?  : y
Jenis     : Celana
Warna     : Biru
Merk      : Levis
Ukuran    : L
Harga     : 189500
Apakah anda akan menambahkan data baru ke stack (y/n)?  : n
Isi stack
Celana Biru Levis L 189500.0
Kemeja Putih Styvea XL 127000.0
Kaos Nevada Hitam M 85000.0

Data yang keluar : Celana BiruLevis L 189500.0
Elemen teratas   : Kemeja Putih Styvea XL 127000.0
Isi stack
Kemeja Putih Styvea XL 127000.0
Kaos Nevada Hitam M 85000.0
```

Pertanyaan Percobaan 2

1) Berapa banyak data pakaian yang dapat ditampung di dalam stack? Tunjukkan potongan kode program untuk mendukung jawaban Anda tersebut!

Terdapat 5 data yang dapat ditampung pada stack

```
StackPakaian_18 stack = new StackPakaian_18(size:5);
```

2) Perhatikan class StackMain, pada saat memanggil fungsi push, parameter yang dikirimkan adalah p. Data apa yang tersimpan pada variabel p tersebut?

Data yang tersimpan adalah data objek dengan atribut jenis, merk, ukuran, dan harga

```
Pakaian_18 pakaian = new Pakaian_18(jenis, warna, merk, ukuran, harga);
stack.push(pakaian);
```

Catatan p = pakaian

3) Apakah fungsi penggunaan do-while yang terdapat pada class StackMain?

Untuk perulangan apabila user ingin memasukkan data baru

4) Modifikasi kode program pada class StackMain sehingga pengguna dapat memilih operasi operasi pada stack (push, pop, peek, atau print) melalui pilihan menu program dengan memanfaatkan kondisi IF-ELSE atau SWITCH-CASE!

```java
import java.util.Scanner;

public class StackMainPakaian_18 {

    public static void main(String[] args) {

        Scanner input18 = new Scanner(System.in);
        Scanner input18String = new Scanner(System.in);

        StackPakaian_18 stack = new StackPakaian_18(5);
        char pilihan;
        int pilihanInt = 0;

        do {

            System.out.println("============================");
            System.out.println("            MENU");
            System.out.println("============================");
            System.out.println("1. Push");
            System.out.println("2. Pop");
            System.out.println("3. Peek");
            System.out.println("4. Print");
            System.out.println("0. Exit");
            System.out.println("============================");
            System.out.print("Masukkan pilihan : ");
            pilihanInt = input18.nextInt();

            switch (pilihanInt) {
                case 1:
                    do {
                        if (stack.isFull()) {
                            System.out.println("Stack sudah penuh!");
                            break;
                        }

                        System.out.print("\nJenis     : ");
                        String jenis = input18String.nextLine();
                        System.out.print("Warna     : ");
                        String warna = input18String.nextLine();
                        System.out.print("Merk      : ");
                        String merk = input18String.nextLine();
                        System.out.print("Ukuran    : ");
                        String ukuran = input18String.nextLine();
                        System.out.print("Harga     : ");
                        double harga = input18.nextDouble();

                        Pakaian_18 pakaian = new Pakaian_18(jenis, warna, merk, ukuran, harga);

                        System.out.print("Apakah anda akan menambahkan data baru ke stack (y/n)?  : ");
                        pilihan = input18.next().charAt(0);

                        stack.push(pakaian);

                    } while (pilihan == 'y');
                    break;
                case 2:
                    System.out.println("============================");
                    stack.pop();
                    break;
                case 3:
                    System.out.println("============================");
                    stack.peek();
                    break;
                case 4:
                    System.out.println("============================");
                    stack.print();
                    break;
                case 0:
                    System.out.println("============================");
                    System.out.println("    Keluar dari program.");
                    break;
                default:
                    System.out.println("============================");
                    System.out.println(" Masukkan pilihan yang BENAR");
                    break;
            }

        } while (pilihanInt != 0);

    }

}
```

```
============================
           MENU
============================
1. Push
2. Pop
3. Peek
4. Print
0. Exit
============================
Masukkan pilihan : 1

Jenis     : Kaos
Warna     : Hitam
Merk      : Nevada
Ukuran    : M
Harga     : 85000
Apakah anda akan menambahkan data baru ke stack (y/n)?  : y

Jenis     : Kemeja
Warna     : Putih
Merk      : Styvea
Ukuran    : XL
Harga     : 127000
Apakah anda akan menambahkan data baru ke stack (y/n)?  : y

Jenis     : Celana
Warna     : Biru
Merk      : Levis
Ukuran    : L
Harga     : 189500
Apakah anda akan menambahkan data baru ke stack (y/n)?  : n
```

```
==============================
            MENU
==============================
1. Push
2. Pop
3. Peek
4. Print
0. Exit
==============================
Masukkan pilihan : 2
==============================
Data yang keluar : Celana BiruLevis L 189500.0
==============================
            MENU
==============================
1. Push
2. Pop
3. Peek
4. Print
0. Exit
==============================
Masukkan pilihan : 3
==============================
Elemen teratas   : Kemeja Putih Styvea XL 127000.0
==============================
            MENU
==============================
1. Push
2. Pop
3. Peek
4. Print
0. Exit
==============================
Masukkan pilihan : 4
==============================
Isi stack
Kemeja Putih Styvea XL 127000.0
Kaos Hitam Nevada M 85000.0

==============================
            MENU
==============================
1. Push
2. Pop
3. Peek
4. Print
0. Exit
==============================
Masukkan pilihan : 0
==============================
    Keluar dari program.
```

Percobaan 3

1) Buat class baru dengan nama Postfix. Tambahkan atribut n, top, dan stack sesuai diagram class Postfix tersebut.

2) Tambahkan pula konstruktor berparameter seperti gambar berikut ini.

```
public Postfix_18(int total) {
    n = total;
    top = -1;
    stack = new char[n];
    push(c:'(');
}
```

3) Buat method push dan pop bertipe void

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

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

4) Buat method IsOperand dengan tipe boolean yang digunakan untuk mengecek apakah elemen data berupa operand.

```
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;
    }
}
```

5) Buat method IsOperator dengan tipe boolean yang digunakan untuk mengecek apakah elemen data berupa operator.

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

6) Buat method derajat yang mempunyai nilai kembalian integer untuk menentukan derajat operator.

```
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;
    }
}
```

7) Buat method konversi untuk melakukan konversi notasi infix menjadi notasi postfix dengan cara mengecek satu persatu elemen data pada String Q sebagai parameter masukan.

```
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;
        } else if (c == '(') {
            push(c);
        } else if (c == ')') {
            while (stack[top] != '(') {
                p = p + pop();
            }
            pop();
        } else if (isOperator(c)) {
            while (derajat(stack[top]) >= derajat(c)) {
                p = p + pop();
            }
            push(c);
        }
    }
    return p;
}
```

8) Selanjutnya, buat class baru dengan nama PostfixMain. Buat class main, kemudian buat variabel P dan Q. Variabel P digunakan untuk menyimpan hasil akhir notasi postfix setelah dikonversi, sedangkan variabel Q digunakan untuk menyimpan masukan dari pengguna berupa ekspresi matematika dengan notasi infix. Deklarasikan variabel Scanner dengan nama sc, kemudian panggil fungsi built-in trim yang digunakan untuk menghapus adanya spasi di depan atau di belakang teks dari teks persamaan yang dimasukkan oleh pengguna.

```
Scanner input18 = new Scanner(System.in);
String P = "";
String Q;

System.out.print(s:"Masukkan ekspresi matematika (infix) : ");
Q = input18.nextLine();
Q = Q.trim();
Q = Q + ")";
```

Penambahan string “)” digunakan untuk memastikan semua simbol/karakter yang masih berada di stack setelah semua persamaan terbaca, akan dikeluarkan dan dipindahkan ke postfix.

9) Buat variabel total untuk menghitung banyaknya karaketer pada variabel Q.

```
int total = Q.length();
```

10) Lakukan instansiasi objek dengan nama post dan nilai parameternya adalah total. Kemudian panggil method konversi untuk melakukan konversi notasi infix Q menjadi notasi postfix P.

```
Postfix_18 post = new Postfix_18(total);
P = post.konversi(Q);
System.out.println("Postfix : " + P);
```

11) Compile dan jalankan class PostfixMain dan amati hasilnya serta verifikasi percobaan.

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

Pertanyaan Percobaan 3

1) Perhatikan class Postfix, jelaskan alur kerja method derajat!

Sebuah char akan dipindahkan atau dikeluarkan dari stack jika posisinya derajat lebih tinggi.

2) Apa fungsi kode program berikut?

c adalah inisialisaliasi type data character Q yang digunakan untuk mengambil karakter dari sebuah string Q

3) Jalankan kembali program tersebut, masukkan ekspresi 5*4^(1+2)%3. Tampilkan hasilnya!

```
Masukkan ekspresi matematika (infix) : 5*4^(1+2)%3
Postfix : 5412+^*3%
```

4) Pada soal nomor 3, mengapa tanda kurung tidak ditampilkan pada hasil konversi? Jelaskan!

Karena tanda kurung hanya digunakan untuk mengatur urutan operasi, tetapi tidak diikutsertakan dalam hasil akhir postfix

Link Repository : https://github.com/zidnafaz/Praktikum-Algoritma-Struktur-Data