

LAPORAN PRAKTIKUM JOBSHEET 8
STACK
MATA KULIAH ALGORITMA DAN STRUKTUR DATA



Disusun Oleh :
Jami'atul Afifah (2341760102)
SIB-1F

PROGRAM STUDI D4 SISTEM INFOEMASI BISNIS
JURUSAN TEKNOLOGI INFORMASI
POLITEKNIK NEGERI MALANG
2024

Praktikum 1

1. Buat folder dengan nama Praktikum07. Buat file Stack.java.
2. Tulis kode untuk membuat atribut dan konstruktor pada class Stack sebagai berikut:

```
public class Stack {  
    int data[];  
    int size;  
    int top;  
  
    public Stack(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() {
    if (!isEmpty()) {
        System.out.println("Elemen teratas stack: " + data[top]);
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

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.print(data[i] + " ");
    }
    System.out.println();
}

public void clear() {
    if (!isEmpty()) {
        top = -1;
        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.

```
1 public class StackDemo {
2     Run | Debug
3     public static void main(String[] args) {
4         Stack stack = new Stack(size:10);
5         stack.push(dt:8);
6         stack.push(dt:12);
7         stack.push(dt:18);
8         stack.print();
9         stack.pop();
10        stack.peek();
11        stack.pop();
12        stack.push(-5);
13        stack.print();
14    }
15 }
```

8. Compile dan run class StackDemo.

```
8092d62ae/69/91fac83cdd42c8a\redhat.java\jdk_w
Isi stack:
18 12 8
Data yang dikeluarkan dari stack: 18
Elemen teratas stack: 12
Data yang dikeluarkan dari stack: 12
Isi stack:
-5 8
PS D:\Matkul\SEM 2\ASD\Jobsheet8> █
```

Pertanyaan

1. Pada method `pop()`, mengapa diperlukan pemanggilan method `isEmpty()`? Apa yang terjadi jika tidak ada pemanggilan `isEmpty()`?
Pemanggilan method `isEmpty()` pada method `pop()` diperlukan untuk memastikan bahwa stack tidak kosong sebelum mencoba untuk menghapus elemen dari stack. Jika tidak ada pemanggilan `isEmpty()`, maka pada saat method `pop()` dipanggil, program akan mencoba untuk menghapus elemen dari stack tanpa memeriksa apakah stack tersebut kosong atau tidak. Hal ini dapat menyebabkan error atau kegagalan dalam program, terutama jika stack sudah kosong dan kita mencoba untuk menghapus elemen dari stack yang tidak memiliki elemen. Dengan memeriksa apakah stack kosong sebelum melakukan operasi `pop`, kita dapat menghindari kesalahan seperti itu dan menangani kasus stack kosong secara lebih aman.
2. Jelaskan perbedaan antara method `peek()` dengan method `pop()` pada class `Stack`.
 - a. `peek()`: Method `peek()` digunakan untuk melihat elemen teratas dari stack tanpa menghapusnya dari stack. Ini berarti bahwa elemen teratas tetap berada di stack setelah pemanggilan `peek()`. Method `peek()` hanya mengembalikan nilai dari elemen teratas tanpa mengubah keadaan stack.
 - b. `pop()`: Method `pop()` digunakan untuk menghapus elemen teratas dari stack dan mengembalikan nilainya. Setelah pemanggilan `pop()`, elemen teratas stack dihapus dan stack akan menjadi lebih pendek. Dengan kata lain, `pop()` tidak hanya mengembalikan nilai dari elemen teratas, tetapi juga mengubah keadaan stack dengan menghapus elemen teratasnya.

Praktikum 2

1. Perhatikan Diagram Class Pakaian berikut ini:

Pakaian
jenis: String warna: String merk: String ukuran: String harga: double
Pakaian(jenis: String, warna: String, merk: String, ukuran: String, harga: double)

Berdasarkan diagram class tersebut, akan dibuat program class `Pakaian` dalam Java.

2. Buat class baru dengan nama `Pakaian`.
3. Tambahkan atribut-atribut `Pakaian` seperti pada Class Diagram `Pakaian`, kemudian tambahkan pula konstruktornya seperti gambar berikut ini.

```

public class Pakaian {
    String jenis, warna, merk, ukuran;
    double harga;

    public Pakaian(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;
    }
}

```

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

Stack
size: int top: int data[]: Pakaian
Stack(size: int) isEmpty(): boolean IsFull(): boolean push(): void pop(): void peek(): void print(): void clear(): void

Keterangan: Tipe data pada variabel data menyesuaikan dengan data yang 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

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

```

public class StackPakaian {
    int size;
    int top;
    Pakaian data[];

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

```

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

```

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

```

7. 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;  
    }  
}
```

8. Buat method push bertipe void untuk menambahkan isi elemen stack dengan parameter pkp

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

9. 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()) {  
        System.out.println("Data yang dikeluarkan dari stack: " + data[top].jenis + " " + data[top].warna + " " +  
            data[top].merk + " " + data[top].ukuran + " " + data[top].harga);  
        top--;  
    } else {  
        System.out.println(x:"Stack masih kosong");  
    }  
}
```

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

```
public void peek() {  
    if (!IsEmpty()) {  
        System.out.println("Elemen teratas: " + data[top].jenis + " " + data[top].warna + " " +  
            data[top].merk + " " + data[top].ukuran + " " + data[top].harga);  
    } else {  
        System.out.println(x:"Stack masih kosong");  
    }  
}
```

11. 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:"");  
}
```

12. Buat method clear bertipe void untuk menghapus seluruh isi stack.

```
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");
    }
}
```

13. Selanjutnya, buat class baru dengan nama StackMain. Buat fungsi main, kemudian lakukan instansiasi objek dari class Stack dengan nama stk dan nilai parameternya adalah `Stack stk = new Stack(5);`
14. Deklarasikan Scanner dengan nama sc
15. Tambahkan kode berikut ini untuk menerima input data Pakaian, kemudian semua informasi tersebut dimasukkan ke dalam stack

```
char pilih;
do {
    System.out.print(s:"Jenis: ");
    String jenis = sc.nextLine();
    System.out.print(s:"Warna: ");
    String warna = sc.nextLine();
    System.out.print(s:"Merk: ");
    String merk = sc.nextLine();
    System.out.print(s:"Ukuran: ");
    String ukuran = sc.nextLine();
    System.out.print(s:"Harga: ");
    double harga = sc.nextDouble();
    Pakaian p = new Pakaian(jenis, warna, merk, ukuran, harga);
    System.out.print(s:"Apakah Anda akan menambahkan data baru ke stack (y/n)?");
    pilih = sc.next().charAt(index:0);
    sc.nextLine();
    stk.push(p);
} while (pilih == 'y');
```

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

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

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

17. Compile dan jalankan class StackMain, kemudian amati hasilnya

```
c:\2f9ac3\bin' 'StackPakaianMain'
Jenis: kaos
Warna: hitam
Merk: nevada
Ukuran: M
Harga: 85000
Apakah Anda akan menambahkan data baru ke stack (y/n)?y
Jenis: kemeja
Warna: biru
Merk: levis
Ukuran: M
Harga: 60000
Apakah Anda akan menambahkan data baru ke stack (y/n)?y
Jenis: celana
Warna: pink
Merk: jjeans
Ukuran: S
Harga: 80000
Apakah Anda akan menambahkan data baru ke stack (y/n)?y
Jenis: outer
Warna: coklat
Merk: cardinal
Ukuran: XL
Harga: 90000
Apakah Anda akan menambahkan data baru ke stack (y/n)?n
Isi stack:
outer coklat cardinal XL 90000.0
celana pink jjeans S 80000.0
kemeja biru levis M 60000.0
kaos hitam nevada M 85000.0

Data yang dikeluarkan dari stack: outer coklat cardinal XL 90000.0
Elemen teratas: celana pink jjeans S 80000.0
Isi stack:
celana pink jjeans S 80000.0
kemeja biru levis M 60000.0
kaos hitam nevada M 85000.0
```

Pertanyaan

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

Jumlah data pakaian yang dapat ditampung di dalam stack ditentukan oleh ukuran stack yang didefinisikan pada saat pembuatan objek stack. Dalam kasus ini, ukuran stack ditentukan saat pembuatan objek StackPakaian. Sebagai contoh, jika kita membuat objek StackPakaian dengan ukuran 5, maka stack tersebut dapat menampung hingga 5 data pakaian.

```
StackPakaian stk = new StackPakaian(size:5);
```


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

stk.push(p);

Pada saat memanggil fungsi push dalam class StackMain, parameter yang dikirimkan adalah objek p yang merupakan instansiasi dari kelas Pakaian. Data yang tersimpan pada variabel p adalah informasi tentang pakaian yang dimasukkan oleh pengguna, yaitu jenis, warna, merk, ukuran, dan harga.

- Apakah fungsi penggunaan do-while yang terdapat pada class StackMain?
Fungsi penggunaan do-while pada class StackMain digunakan untuk meminta input dari pengguna untuk menambahkan data pakaian ke dalam stack. Program akan terus meminta input hingga pengguna memilih untuk tidak menambahkan data lagi (memasukkan input selain 'y'). Ini memungkinkan pengguna untuk menambahkan lebih dari satu data pakaian secara berturut-turut.
- 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!

```
int choice;
do {
    System.out.println(x:"Menu:");
    System.out.println(x:"1. Push");
    System.out.println(x:"2. Pop");
    System.out.println(x:"3. Peek");
    System.out.println(x:"4. Print");
    System.out.println(x:"5. Exit");
    System.out.print(s:"Pilih operasi yang ingin dilakukan: ");
    choice = sc.nextInt();
    sc.nextLine();

    switch (choice) {
        case 1:
            System.out.print(s:"Jenis: ");
            String jenis = sc.nextLine();
            System.out.print(s:"Warna: ");
            String warna = sc.nextLine();
            System.out.print(s:"Merk: ");
            String merk = sc.nextLine();
            System.out.print(s:"Ukuran: ");
            String ukuran = sc.nextLine();
            System.out.print(s:"Harga: ");
            double harga = sc.nextDouble();
            Pakaian p = new Pakaian(jenis, warna, merk, ukuran, harga);
            stk.push(p);
            break;
        case 2:
            stk.pop();
            break;
        case 3:
            stk.peek();
            break;
        case 4:
            stk.print();
            break;
        case 5:
            System.out.println(x:"Program berakhir.");
            break;
        default:
            System.out.println(x:"Pilihan tidak valid.");
    }
} while (choice != 5);
```

Praktikum 3

1. Perhatikan Diagram Class berikut ini:

Postfix
n: int top: int stack: char[]
Postfix(total: int) push(c: char): void pop(): void IsOperand(c: char): boolean IsOperator(c: char): boolean derajat(c: char): int konversi(Q: String): string

2. Buat class baru dengan nama Postfix. Tambahkan atribut n, top, dan stack sesuai diagram class Postfix tersebut.
3. Tambahkan pula konstruktor berparameter seperti gambar berikut ini.

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

4. 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;  
}
```

5. 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;  
    }  
}
```

6. 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 == '+') {  
        return true;  
    } else {  
        return false;  
    }  
}
```

7. Buat method `derajat` yang mempunyai nilai kembalian integer untuk menentukan derajat operator.

```
public int derajat(char c) {  
    switch (c) {  
        case '^':  
            return 3;  
        case '*':  
        case '/':  
            return 2;  
        case '+':  
        case '-':  
            return 1;  
        default:  
            return 0;  
    }  
}
```

8. 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 += c;  
        }  
        if (c == '(') {  
            push(c);  
        }  
        if (c == ')') {  
            while (stack[top] != '(') {  
                P += pop();  
            }  
            pop(); // discard '('  
        }  
        if (IsOperator(c)) {  
            while (derajat(stack[top]) >= derajat(c)) {  
                P += pop();  
            }  
            push(c);  
        }  
    }  
    return P;  
}
```

9. 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 sc = new Scanner(System.in);
String P, Q;

System.out.println(x:"Masukkan ekspresi matematika (infix): ");
Q = sc.nextLine();
Q = Q.trim() + " ";
int total = Q.length();
```

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

10. Buat variabel total untuk menghitung banyaknya karakter pada variabel Q.

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

11. 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 post = new Postfix(total);
P = post.konversi(Q);

System.out.println("Postfix: " + P);
}
```

12. Compile dan jalankan class PostfixMain dan amati hasilnya.

```
Masukkan ekspresi matematika (infix):
a+b*(c+d-e)/f
Postfix: abcd+e-*f/+
PS D:\Matkul\SEM 2\ASD\Jobsheet8> 
```

Pertanyaan

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

Method derajat(char c) pada class Postfix berfungsi untuk menentukan derajat operator matematika. Derajat operator digunakan untuk menentukan urutan operasi dalam ekspresi matematika, di mana operator dengan derajat yang lebih tinggi akan dievaluasi terlebih dahulu. Alur kerja method derajat adalah sebagai berikut:

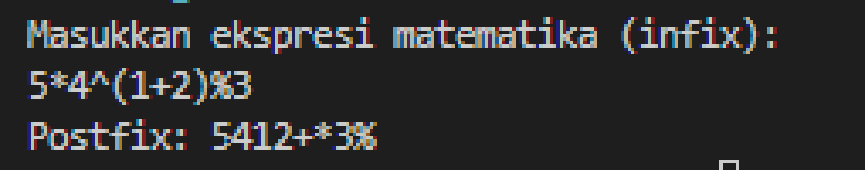
- a. Method menerima satu karakter operator sebagai argumen.

- b. Kemudian, menggunakan sebuah switch-case, method memeriksa karakter tersebut dan mengembalikan nilai derajat sesuai dengan operator tersebut.
 - c. Untuk operator $^$ (pangkat), derajatnya adalah 3.
 - d. Untuk operator $*$ dan $/$ (perkalian dan pembagian), derajatnya adalah 2.
 - e. Untuk operator $+$ dan $-$ (penjumlahan dan pengurangan), derajatnya adalah 1.
 - f. Jika karakter tidak cocok dengan operator apa pun, method mengembalikan nilai 0.
2. Apa fungsi kode program berikut?

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

Kode program `c = Q.charAt(i);` berfungsi untuk mengambil karakter pada indeks ke-i dari string Q dan menyimpannya ke dalam variabel c. Dalam konteks ini, string Q adalah ekspresi matematika dalam notasi infix, dan variabel c digunakan untuk mewakili karakter saat iterasi melalui 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!

Tanda kurung "(" dan ")" tidak ditampilkan pada hasil konversi karena dalam algoritma konversi dari infix ke postfix, tanda kurung hanya digunakan sebagai pembatas untuk mengatur urutan operasi. Ketika menemui tanda kurung buka "(", maka tanda kurung tersebut akan didorong ke dalam stack, dan ketika menemui tanda kurung tutup ")", maka operator-operator yang ada dalam stack akan dikeluarkan hingga menemui tanda kurung buka yang sesuai. Tanda kurung tersebut tidak termasuk dalam ekspresi postfix karena hanya digunakan untuk mengatur urutan operasi dan tidak menyatakan operasi matematika itu sendiri. Oleh karena itu, tanda kurung tidak ditampilkan pada hasil konversi.

Tugas

1. Tambahkan method getMax pada class Stack yang digunakan untuk mencari dan menampilkan data pakaian dengan harga tertinggi dari semua data pakaian yang tersimpan di dalam stack!

```
public void getMax() {
    if (!isEmpty()) {
        double hargaMax = data[0].harga;
        int indexMax = 0;
        for (int i = 1; i <= top; i++) {
            if (data[i].harga > hargaMax) {
                hargaMax = data[i].harga;
                indexMax = i;
            }
        }
        System.out.println("Data dengan harga tertinggi: " + data[indexMax].jenis + " " + data[indexMax].warna +
            " " + data[indexMax].merk + " " + data[indexMax].ukuran + " " + data[indexMax].harga);
    } else {
        System.out.println(x:"Stack masih kosong");
    }
}
```

2. Setiap hari Minggu, Dewi pergi berbelanja ke salah satu supermarket yang berada di area rumahnya. Setiap kali selesai berbelanja, Dewi menyimpan struk belanjanya di dalam laci. Setelah dua bulan, ternyata Dewi sudah mempunyai delapan struk belanja. Dewi berencana mengambil lima struk belanja untuk ditukarkan dengan voucher belanja. Buat sebuah program stack untuk menyimpan data struk belanja Dewi, kemudian lakukan juga proses pengambilan data struk belanja sesuai dengan jumlah struk yang akan ditukarkan dengan voucher. Informasi yang tersimpan pada struk belanja terdiri dari:
 - a. Nomor transaksi
 - b. Tanggal pembelian
 - c. Jumlah barang yang dibeli
 - d. Total harga bayar

Tampilkan informasi struk belanja yang masih tersimpan di dalam stack

Buat File Struk17.java

```

tugas > J Struk14.java
1 package tugas;
2
3 public class Struk14 {
4     int noTransaksi, jumlahBarang;
5     String tglBeli;
6     int totalBayar;
7     int size;
8     int top;
9     Struk14 data[];
10    Struk14[] stk;
11
12    Struk14(int no, String tgl, int jb, int tb) {
13        noTransaksi = no;
14        tglBeli = tgl;
15        jumlahBarang = jb;
16        totalBayar = tb;
17    }
18
19    public Struk14(int size) {
20        this.size = size;
21        data = new Struk14[size];
22        top = -1;
23    }
24
25    public boolean isEmpty() {
26        return top == -1;
27    }
28
29    public boolean isFull() {
30        return top == size - 1;
31    }
32
33    public void push(Struk14 dt) {
34        if (!isFull()) {
35            top++;
36            data[top] = dt;
37        } else {
38            System.out.println(x:"Isi Stack Penuh!");
39        }
40    }

```

```

tugas > J Struk14.java
3 public class Struk14 {
42    public void pop() {
43        if (!isEmpty()) {
44            Struk14 x = data[top];
45            top--;
46            System.out.println("Data yang keluar: " + x.noTransaksi + " "
47                + x.tglBeli + " " + x.jumlahBarang + " "
48                + x.totalBayar + " ");
49        } else {
50            System.out.println(x:"Stack masih kosong");
51        }
52    }
53
54
55    public void peek() {
56        System.out.println("Elemen teratas: " + data[top].noTransaksi + " "
57            + " " + data[top].tglBeli + " " + data[top].jumlahBarang
58            + " " + data[top].totalBayar);
59    }
60
61    public void print() {
62        System.out.println(x:"Isi stack: ");
63        for (int i = top; i >= 0; i--) {
64            System.out.println(data[i].noTransaksi + " " + data[i].tglBeli + " "
65                + data[i].jumlahBarang + " " + data[i].totalBayar +
66                " ");
67        }
68        System.out.println(x:"");
69    }
70
71    public void clear() {
72        if (!isEmpty()) {
73            for (int i = top; i >= 0; i--) {
74                top--;
75            }
76            System.out.println(x:"Stack sudah dikosongkan");
77        } else {
78            System.out.println(x:"Gagal! Stack masih kosong");
79        }
80    }

```

Buat File StrukMain.java

```
1 package tugas;
2
3 import java.util.Scanner;
4
5 public class StrukMain14 {
6     Scanner sc = new Scanner(System.in);
7
8     Run|Debug
9     public static void main(String[] args) {
10         Scanner sc = new Scanner(System.in);
11         Struk14 stk = new Struk14(size:20);
12         StrukMain14 main = new StrukMain14();
13
14         int pil;
15         while (true) {
16             System.err.println(x:"=====");
17             System.err.println(x:"Pilihan");
18             System.err.println();
19             System.out.println(x:"1. Push");
20             System.out.println(x:"2. Pop");
21             System.out.println(x:"3. Peek");
22             System.out.println(x:"4. Print");
23             System.out.println(x:"5. Kupon");
24             System.out.println(x:"6. Keluar");
25             System.out.print(s:"Pilih salah satu (1/2/3/4/5/6) : ");
26             pil = sc.nextInt();
27             switch (pil) {
28                 case 1:
29                     main.pushPakaian(stk);
30                     break;
31                 case 2:
32                     stk.pop();
33                     break;
34                 case 3:
35                     stk.peak();
36                     break;
37                 case 4:
38                     stk.print();
39                     break;
40                 case 5:
41                     main.kupon(stk);
42                     break;
43                 case 6:
44                     return;
45                 default:
46                     System.out.println(x:"Pilihan tidak tersedia. ");
47                     break;
48             }
49         }
50
51         public void pushPakaian(Struk14 stk) {
52             char pilih;
53             do {
54                 System.out.print(s:"No Belanja: ");
55                 int noTra = sc.nextInt();
56                 System.out.print(s:"Tanggal (dd/mm/yyyy) : ");
57                 String tanggal = sc.next();
58                 System.out.print(s:"Jumlah: ");
59                 int jumlah = sc.nextInt();
60                 System.out.print(s:"Total Bayar: ");
61                 int total = sc.nextInt();
62
63                 Struk14 d = new Struk14(noTra, tanggal, jumlah, total);
64
65                 System.out.print(s:"Apakah Anda akan menambahkan data baru ke stack (y/n) ? ");
66                 pilih = sc.next().charAt(index:0);
67                 stk.push(d);
68             } while (pilih == 'y');
69         }
70
71         public void kupon(Struk14 stk){
72             System.out.println(x:"Data yang diambil");
73             int i = 0;
74             while (i < 5) {
75                 stk.pop();
76                 i++;
77             }
78
79             System.out.println();
80             System.out.println(x:"Data yang tersisa");
81             stk.print();
82         }
83     }
84 }
```


Hasil:

```
Apakah Anda akan menambahkan data baru ke stack (y/n) ? y
No Belanja: 13
Tanggal (dd/mm/yyyy) : 30/03/2023
Jumlah: 5
Total Bayar: 25000
Apakah Anda akan menambahkan data baru ke stack (y/n) ? y
No Belanja: 14
Tanggal (dd/mm/yyyy) : 31/03/2023
Jumlah: 6
Total Bayar: 30000
Apakah Anda akan menambahkan data baru ke stack (y/n) ? n
=====
Pilihan

1. Push
2. Pop
3. Peek
4. Print
5. Kupon
6. Keluar
Pilih salah satu (1/2/3/4/5/6) : 4
Isi stack:
14 31/03/2023 6 30000
13 30/03/2023 5 25000
12 29/03/2023 5 20000
11 28/03/2023 4 15000
10 27/03/2023 3 14000

=====
Pilihan

1. Push
2. Pop
3. Peek
4. Print
5. Kupon
6. Keluar
Pilih salah satu (1/2/3/4/5/6) : 5
Data yang diambil
Data yang keluar: 14 31/03/2023 6 30000
Data yang keluar: 13 30/03/2023 5 25000
Data yang keluar: 12 29/03/2023 5 20000
Data yang keluar: 11 28/03/2023 4 15000
Data yang keluar: 10 27/03/2023 3 14000

Data yang tersisa
Isi stack:
```