# LAPORAN PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

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

# Jobsheet-9 STACK



Nama : Surya Rahmat Fatahillah

NIM : 2341760020

Prodi : Sistem Informasi Bisnis

# JURUSAN TEKNOLOGI INFORMASI POLITEKNIK NEGERI MALANG 2023/2024

# 7.2. Praktikum 1

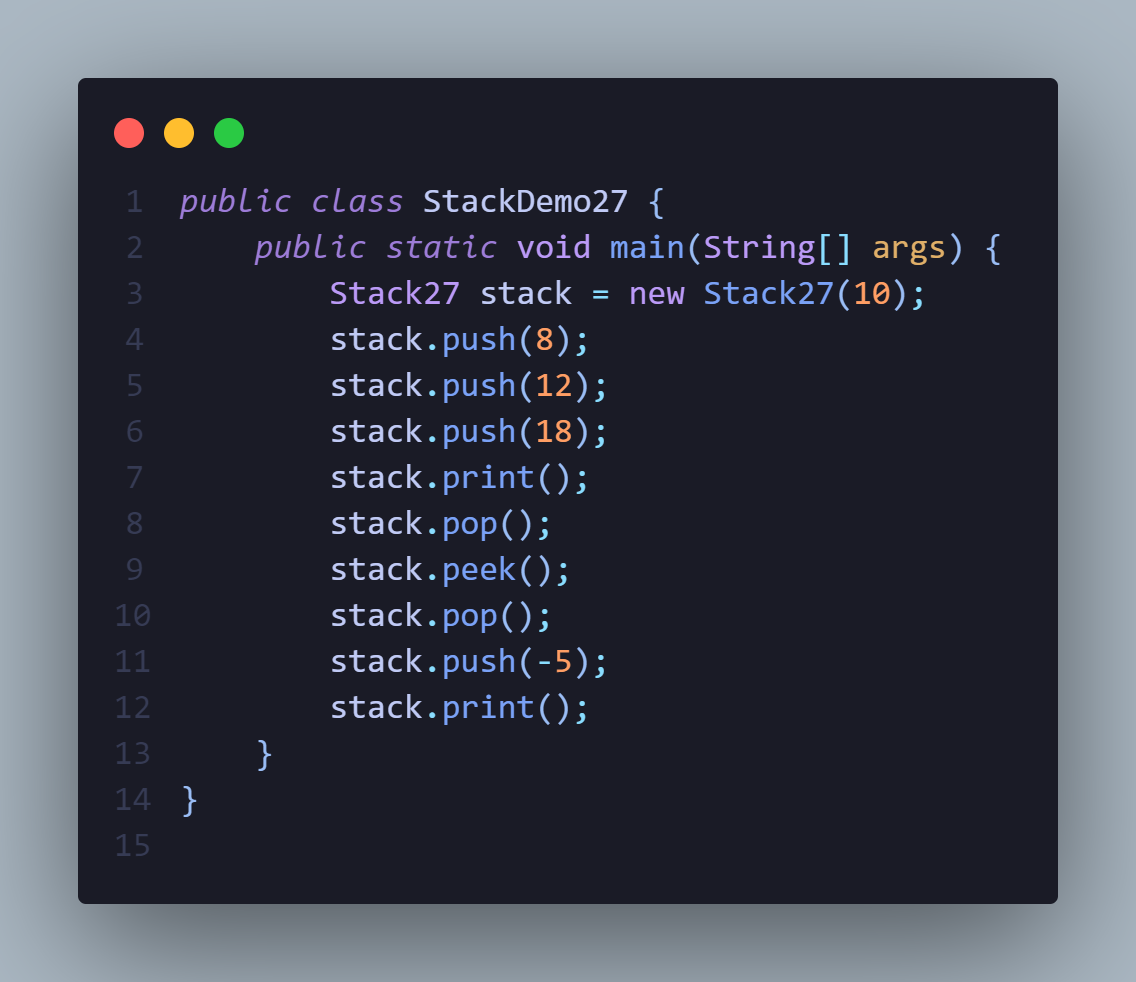
# Pada percobaan ini, kita akan membuat program yang mengimplementasikan struktur data Stack dan operasi-operasi dasar pada struktur data Stack menggunakan array.

**7.2.1 Langkah-langkah Percobaan**

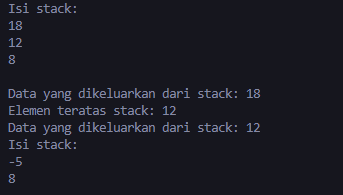
Buat folder dengan nama Praktikum07. Buat file Stack.java. Kemudian tulis kode untuk membuat atribut dan konstruktor pada class Stack dan tambahkan method nya sebagai berikut:

****

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



**7.2.2 Verifikasi Hasil Percobaan**

****

**7.2.3 Pertanyaan**

1. Pada method pop(), mengapa diperlukan pemanggilan method isEmpty()? Apa yang terjadi

jika tidak ada pemanggilan isEmpty()?

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

**Jawaban!**

1. Pemanggilan method isEmpty() pada method pop() digunakan untuk memeriksa apakah stack sudah kosong sebelum mencoba untuk mengeluarkan elemen dari stack. Hal ini dilakukan untuk menghindari error yang disebabkan oleh mencoba untuk mengeluarkan elemen dari stack kosong. Jika tidak ada pemanggilan isEmpty() di dalam method pop(), maka program akan mencoba untuk mengeluarkan elemen dari stack tanpa memeriksa apakah stack tersebut kosong atau tidak. Hal ini dapat menyebabkan error.

Jika tidak ada pengecekan isEmpty() di dalam method pop(), maka stack akan mencoba untuk melakukan operasi pop() tanpa memeriksa apakah stack sudah kosong atau tidak. Hal ini dapat menyebabkan akses ke indeks array yang tidak valid (misalnya, ketika top = -1), yang berpotensi menyebabkan kesalahan runtime seperti ArrayIndexOutOfBoundsException atau error lainnya.

Dengan menggunakan pengecekan isEmpty() sebelum melakukan operasi pop(), kita dapat memastikan bahwa operasi pop() hanya dilakukan ketika stack tidak kosong. Ini membantu kita untuk meminimalkan risiko kesalahan runtime.

1.  Method **peek()** dan **pop()** adalah dua operasi yang berbeda dalam struktur data stack, berikut perbedaanya:

**peek()**:

* Method **peek()** digunakan untuk melihat elemen teratas dari stack tanpa menghapusnya.
* peek hanya mengembalikan nilai dari elemen teratas stack tanpa mengubah struktur stack itu sendiri.
* Operasi **peek()** tidak mempengaruhi nilai dari variabel **top** atau jumlah elemen dalam stack.
* Berguna ketika Anda hanya ingin melihat elemen teratas tanpa menghapusnya dari stack.

**pop()**:

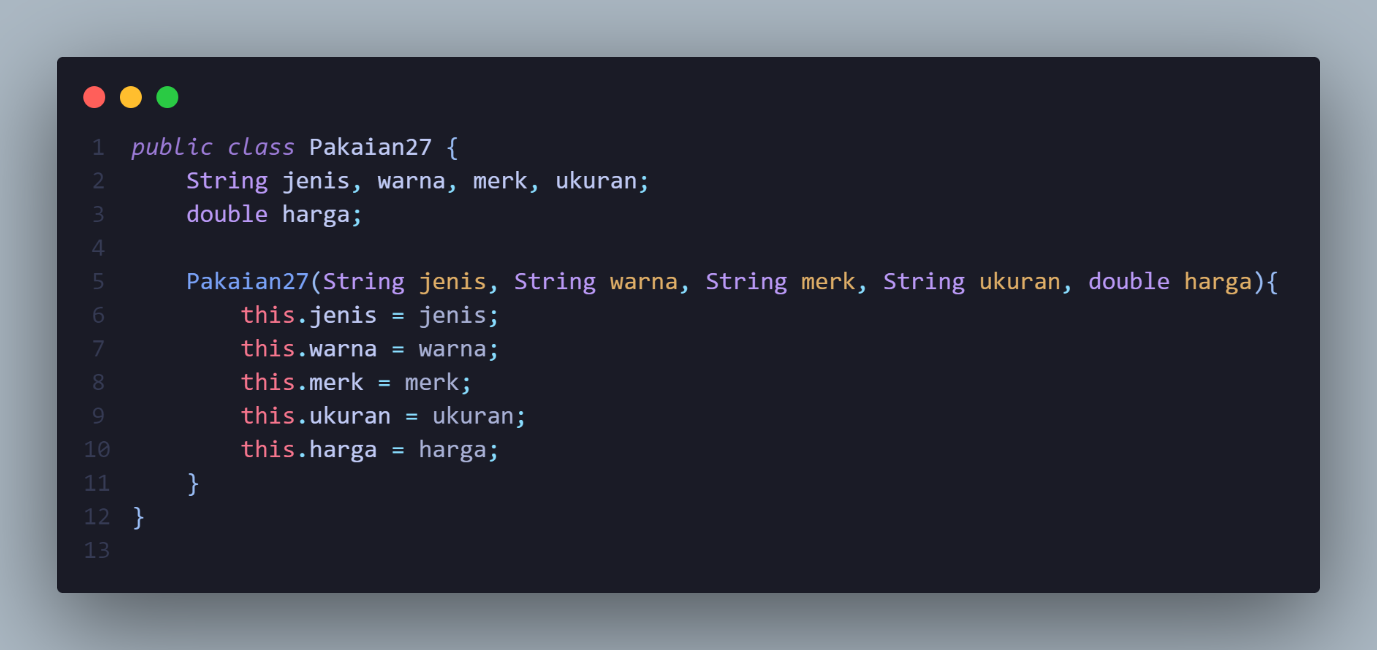
* Method **pop()** digunakan untuk menghapus dan mengembalikan elemen teratas dari stack.
* Pop menghapus elemen teratas stack dan mengurangi ukuran stack (dalam hal ini, nilai **top**).
* Operasi **pop()** secara efektif mengambil elemen teratas dari stack dan mengubah struktur stack itu sendiri.
* Berguna ketika Anda ingin mengambil elemen teratas dari stack untuk digunakan dan juga mengurangi ukuran stack.

**7.3. Praktikum 2**

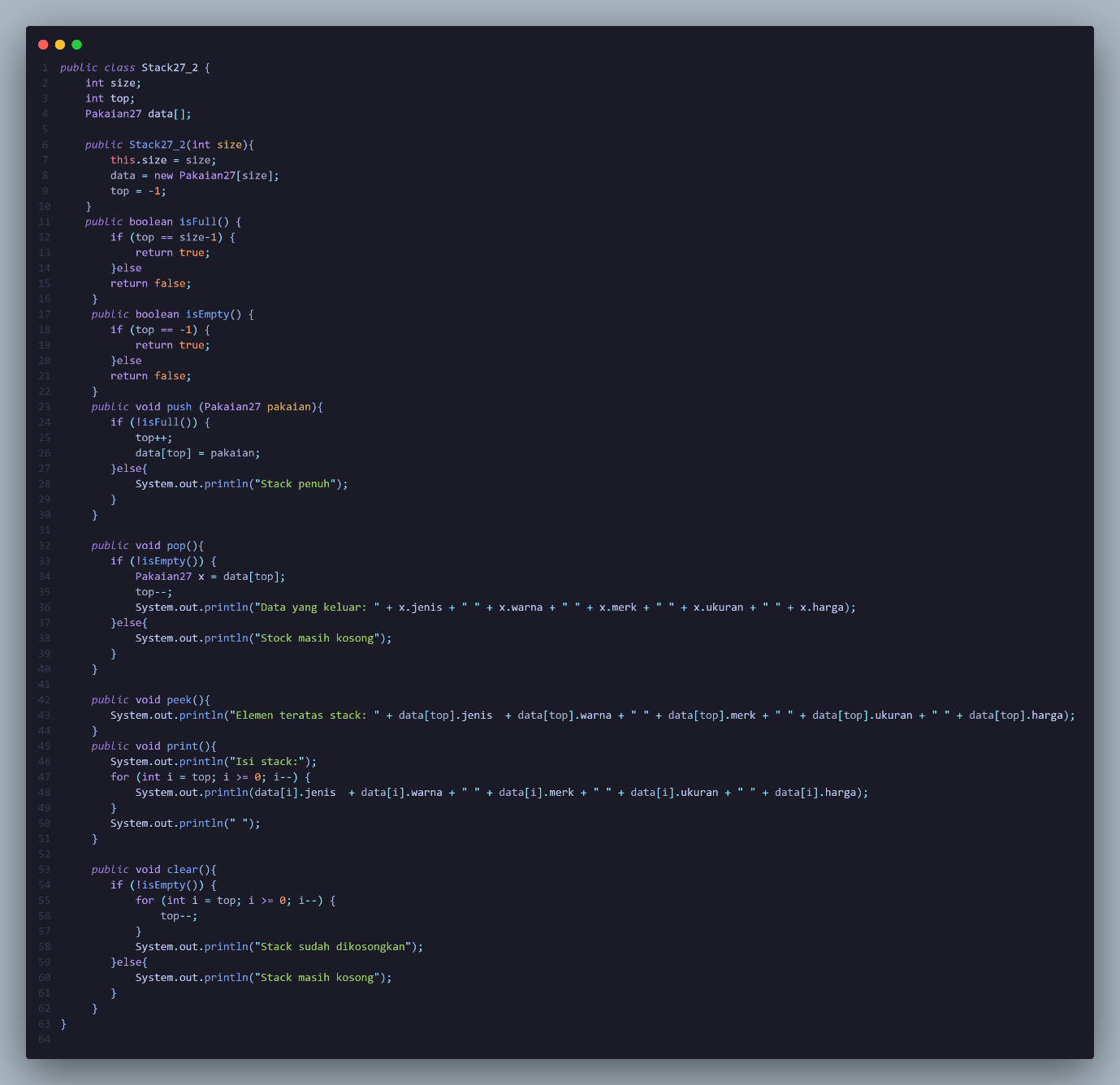
Pada percobaan ini, kita akan membuat program yang mengilustrasikan tumpukan pakaian yang disimpan ke dalam stack. Karena sebuah pakaian mempunyai beberapa informasi, maka implementasi Stack dilakukan dengan menggunakan array of object untuk mewakili setiap elemennya.

**7.3.1. Langkah-langkah Percobaan**

Buat class baru dengan nama Pakaian ,kemudian tambahkan atribut-atribut Pakaian seperti pada Class Diagram Pakaian, tambahkan pula konstruktornya dan method-method nya



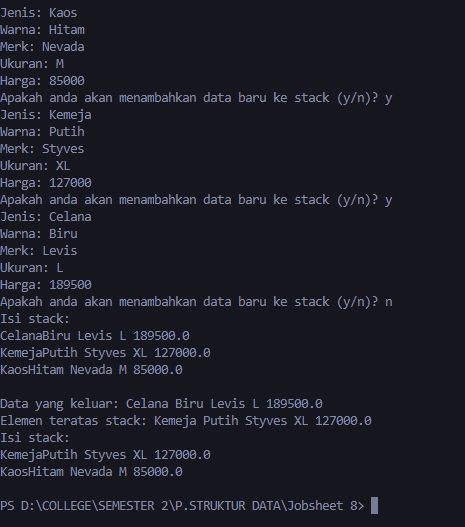
Buat class baru dengan nama Stack. Kemudian tambahkan atribut dan konstruktor seperti gambar berikut ini.



Selanjutnya, buat class baru dengan nama StackMain. Buat fungsi main, kemudian lakukan instansiasi objek dari class Stack dengan nama stk dan nilai parameternya adalah 5. Tambahkan kode untuk menerima input data Pakaian, kemudian semua informasi tersebut dimasukkan ke dalam stack dan Lakukan pemanggilan method print, method pop, dan method peek dengan urutan sebagai berikut



**7.3.2. Verifikasi Hasil Percobaan**

****

**7.3.3. Pertanyaan**

1. Berapa banyak data pakaian yang dapat ditampung di dalam stack? Tunjukkan potongan kode

program untuk mendukung jawaban Anda tersebut!

2. Perhatikan class StackMain, pada saat memanggil fungsi push, parameter yang dikirimkan adalah

p. Data apa yang tersimpan pada variabel p tersebut?

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

4. Modifikasi kode program pada class StackMain sehingga pengguna dapat memilih operasioperasi pada stack (push, pop, peek, atau print) melalui pilihan menu program dengan

memanfaatkan kondisi IF-ELSE atau SWITCH-CASE!

**Jawaban!**

1. Dalam kelas **StackMain27**, kita mendefinisikan variabel stk yang merupakan sebuah objek dari kelas Stack27\_2 dengan ukuran yang diberikan saat objek stack dibuat. Dalam kasus ini yaitu, ukuran stack diberikan saat konstruksi stack dalam method **main()** adalah 5:



Sehingga, stack ini dapat menampung hingga 5 objek **Pakaian27** dalam array **data**.

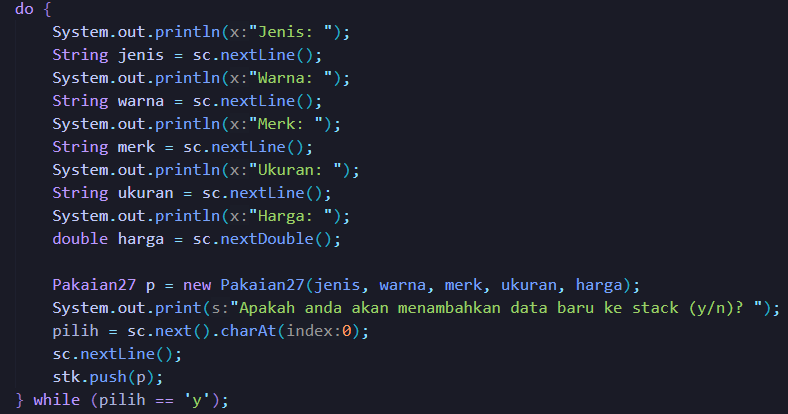
Jadi, jumlah data pakaian yang dapat ditampung di dalam stack adalah 5.

1. Pada saat memanggil fungsi **push** dalam class **StackMain27**, parameter **p** yang dikirimkan adalah objek **Pakaian27** yang baru dibuat menggunakan input pengguna. Objek **Pakaian27** ini menyimpan informasi tentang pakaian yang dimasukkan oleh pengguna melalui input.

Ketika membuat objek **Pakaian27**, parameter **jenis**, **warna**, **merk**, **ukuran**, dan **harga** diinisialisasi dengan nilai-nilai yang diberikan oleh pengguna melalui input. Jadi, pada saat memanggil **push**, data pakaian yang dimasukkan oleh pengguna disimpan dalam objek **Pakaian27** yang dikirimkan sebagai parameter **p**.

1. Fungsi penggunaan **do-while** dalam class **StackMain27** adalah untuk mengulang proses input data pakaian oleh pengguna dan penambahan data pakaian ke dalam stack sampai pengguna memilih untuk berhenti (dengan mengetikkan 'n' saat ditanya apakah ingin menambahkan data baru ke stack).

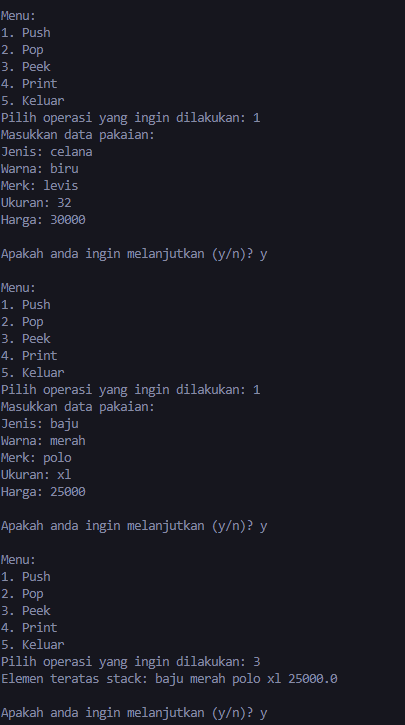
Berikut adalah potongan kode yang menunjukkan penggunaan **do-while** dalam class **StackMain27**:

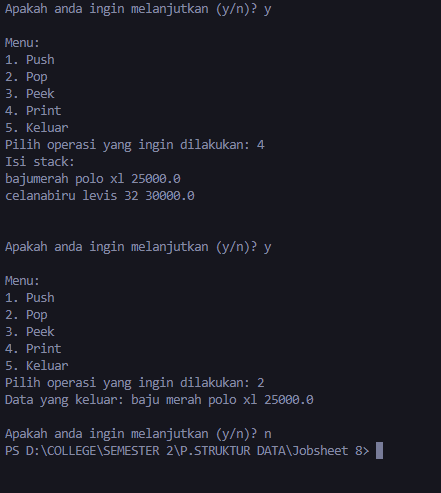


1. Berikut adalah modifikasi pada class **StackMain27** agar pengguna dapat memilih operasi yang ingin dilakukan pada stack melalui pilihan menu program menggunakan switch-case:



Kemudian berikut output programnya:



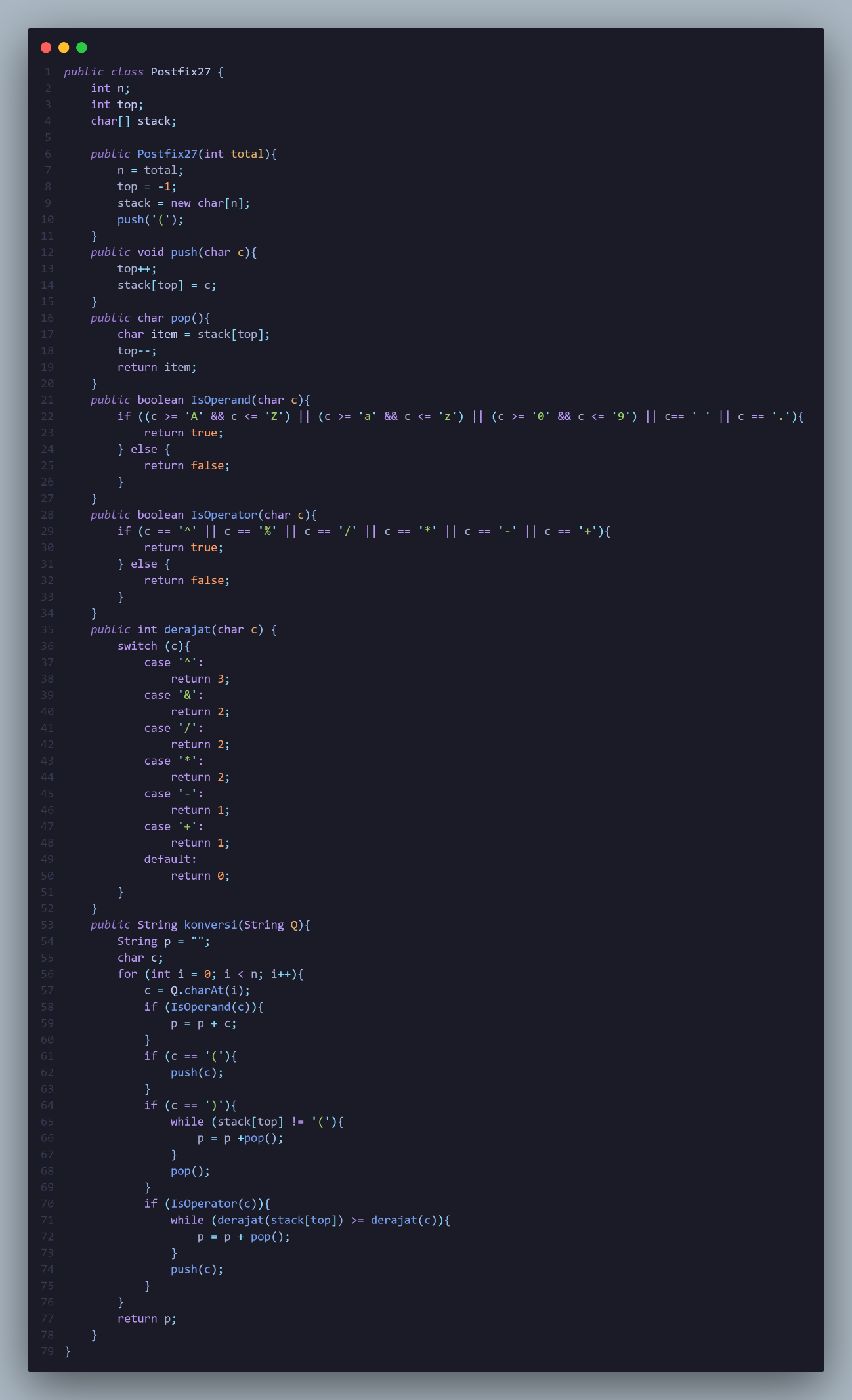


**7.4. Praktikum 3**

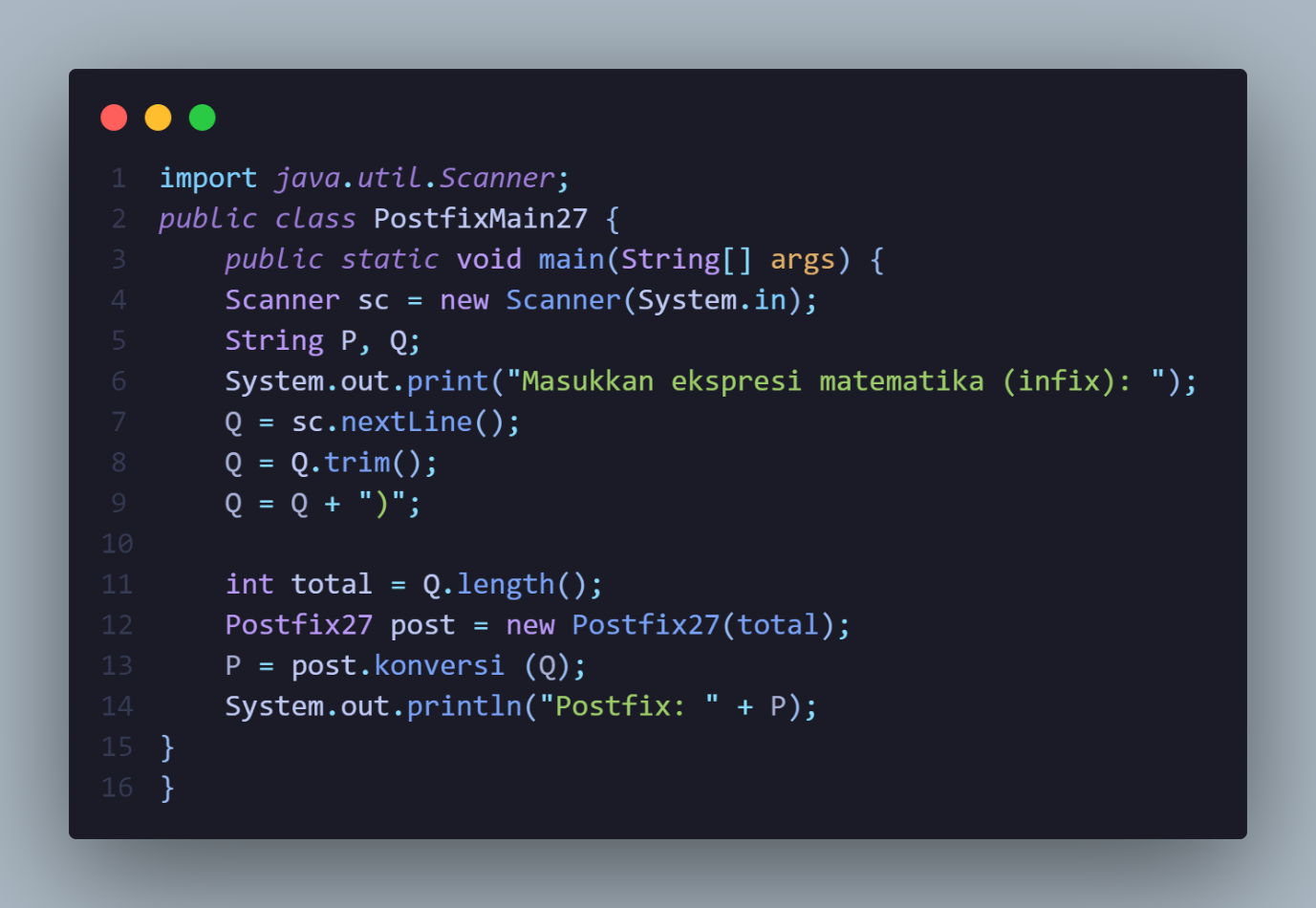
Pada percobaan ini, kita akan membuat program untuk melakukan konversi notasi infix menjadi notasi postfix.

**7.4.1. Langkah-langkah Percobaan**

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. Kemudian buat method push dan pop bertipe void, method IsOperand dengan tipe Boolean, method IsOperator dengan tipe Boolean. method derajat yang mempunyai nilai kembalian integer untuk menentukan derajat operator dan method konversi untuk melakukan konversi notasi infix menjadi notasi postfix dengan cara mengecek satu persatu elemen data pada String Q sebagai parameter masukan.



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.



**7.4.2. Verifikasi Hasil Percobaan**

****

**7.4.3. Pertanyaan**

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

2. Apa fungsi kode program berikut?



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

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

**Jawaban!**

* 1. Method derajat(char c) dalam class Postfix27 digunakan untuk menentukan prioritas operator matematika yang diberikan. Prioritas operator tersebut ditentukan berdasarkan derajat atau tingkatannya. Semakin tinggi nilai derajat suatu operator, semakin tinggi pula prioritasnya.

Berikut adalah alur kerja dari method derajat(char c):

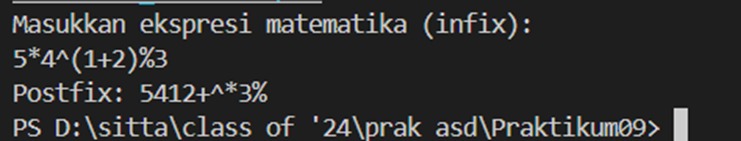
* Parameter c merupakan operator matematika yang akan dinilai derajatnya.
* Di dalam method, kita menggunakan sebuah switch-case untuk mengecek nilai dari c.
* Setiap operator memiliki derajat yang berbeda-beda, dan nilai derajat diatur sesuai dengan prioritas operator dalam operasi matematika.
* Setelah ditemukan operator yang sesuai, method akan mengembalikan nilai derajat dari operator tersebut.
* Jika c tidak sesuai dengan operator yang didefinisikan, method akan mengembalikan nilai 0.

Dengan demikian, method **derajat(char c)** berfungsi untuk memberikan informasi tentang tingkat prioritas atau derajat operator matematika yang diberikan sebagai input. Informasi ini kemudian dapat digunakan dalam proses konversi ekspresi infix menjadi postfix untuk memastikan bahwa operasi dilakukan sesuai dengan prioritas operator yang benar.

* 1. Kode program **c = Q.charAt(i);** memiliki fungsi untuk mengambil karakter pada posisi indeks tertentu dalam string **Q** dan menyimpannya ke dalam variabel **c**.
* Q.charAt(i) mengambil karakter dari string Q pada posisi indeks i.
* Nilai karakter yang diambil kemudian disimpan dalam variabel c.

Dengan demikian, pada setiap iterasi, kode ini akan mengambil karakter dari string Q pada posisi indeks yang ditentukan oleh variabel iterasi i dan menyimpannya dalam variabel c.

* 1. Berikut hasil program dengan ekspresi 5\*4^(1+2)% 3:



* 1. Tanda kurung "(" dan ")" tidak ditampilkan pada hasil konversi postfix karena dalam ekspresi postfix, urutan operasi sudah ditentukan secara langsung berdasarkan urutan operand dan operator tanpa memerlukan tanda kurung untuk menandai grup ekspresi.

Selama proses konversi dari infix ke postfix dilakukan, tanda kurung "(" digunakan untuk menandai awal dari grup ekspresi, sementara tanda kurung ")" digunakan untuk menandai akhir dari grup ekspresi. Namun, dalam ekspresi postfix, tanda kurung tidak lagi diperlukan karena urutan operasi sudah terdefinisi berdasarkan posisi operand dan operator.

Jadi, tanda kurung hanya diperlukan dalam proses evaluasi ekspresi matematika secara keseluruhan, tetapi tidak diperlukan dalam representasi postfix yang telah dihasilkan setelah konversi. Itulah mengapa tanda kurung tidak ditampilkan pada hasil konversi postfix.

**7.5. Tugas**

1. Perhatikan dan gunakan kembali kode program pada Praktikum 2. 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!

2. Setiap hari Minggu, Dewi pergi berbelanja ke salah satu supermarket yang berada di area

rumahnya. Setiap kali selesai berbelanja, Dewi menyimpan struk belanjaannya 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:

• Nomor transaksi

• Tanggal pembelian

• Jumlah barang yang dibeli

• Total harga bayar

Tampilkan informasi struk belanja yang masih tersimpan di dalam stack!

**Jawaban!**

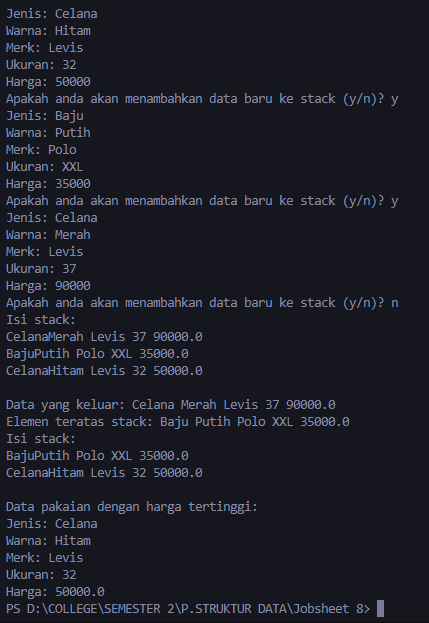
* 1. Berikut modifikasi dari praktikum 2 dengan menambahkan method getMax pada class Stack yang digunakan untuk menampilkan data pakaian dengan harga tertinggi dari semua data pakaian yang tersimpan di dalam stack



Kemudian panggil di kelas main:



Berikut output nya:



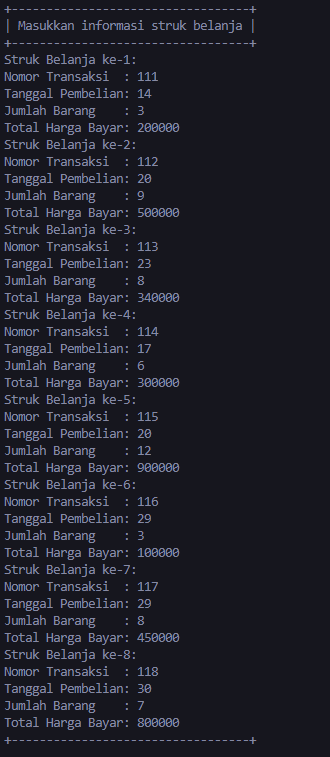
* 1. Berikut merupakan program stack untuk menyimpan data struk belanja Dewi dan kemudian melakukan proses pengambilan data struk belanja sesuai dengan jumlah struk yang akan ditukarkan dengan voucher.
* Class StrukBelanja27

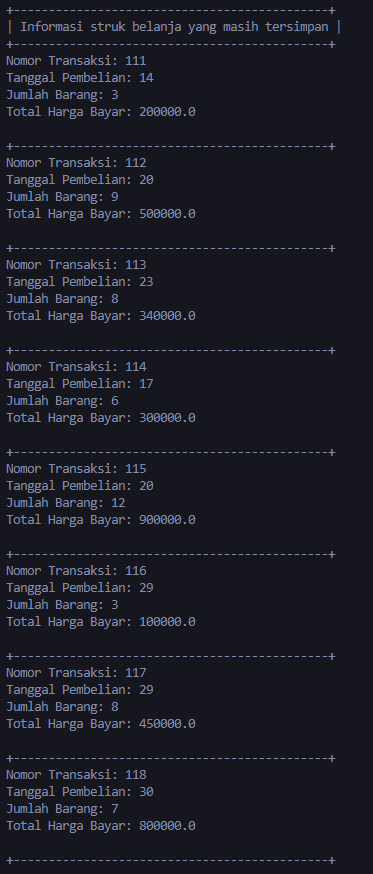


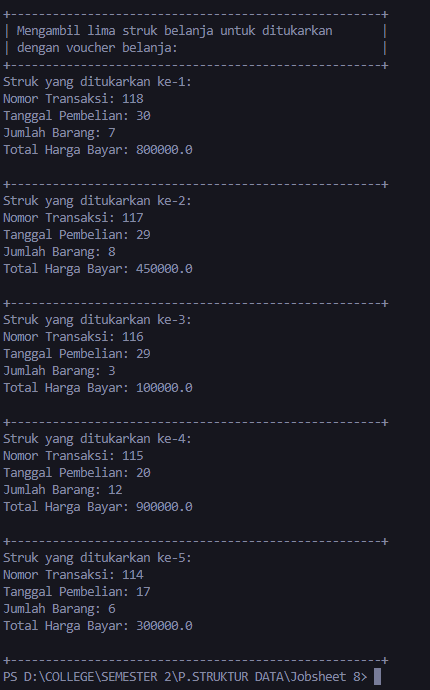
* Class StrukMain27



Kemudian berikut merupakan output dari program diatas:







Commit dan Push Ke Github: