**LAPORAN PRAKTIKUM**

**Algoritma Dan Struktur Data**

**Jobsheet - 7 : Stack**



|  |  |
| --- | --- |
| Nama | : Ghoffar Abdul Ja’far |
| NIM | : 41720035 |
| Kelas | : 1E |

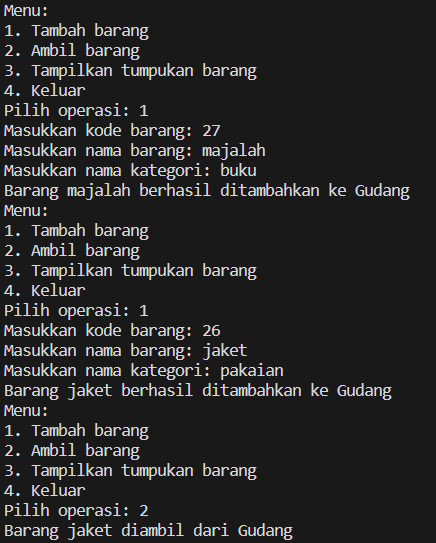
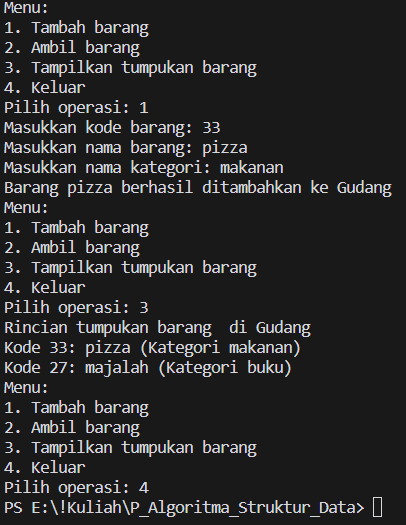
**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2023/2024**

**Percobaan 1: Penyimpanan Tumpukan Barang dalam Gudang**

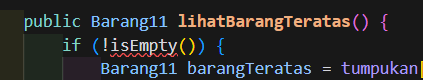
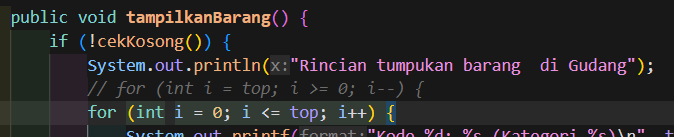
Hasil:

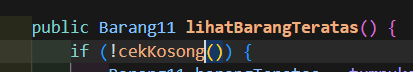
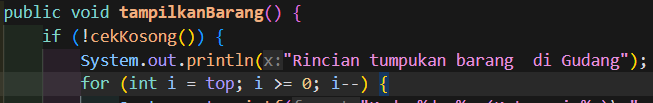
**Pertanyaan**

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

Before:

After:

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

= 7

Potongan kode:

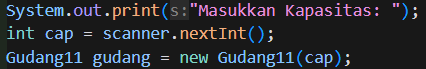
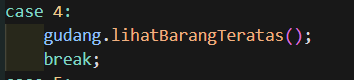


1. Mengapa perlu pengecekan kondisi **!cekKosong()** pada method **tampilkanBarang**? Kalau kondisi tersebut dihapus, apa dampaknya?

= Karena program perlu melakukan cek pada stack apakah ada object yang berada dalam stack atau tidak untuk ditampilkan, jika kondisi tersebut dihapus maka akan berdampak tidak ada pengecekan kondisi stack apakah ada objek atau tidak

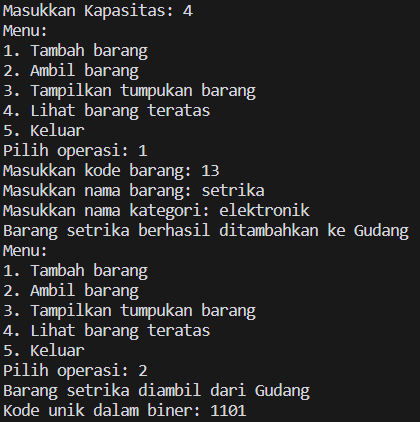
1. Modifikasi kode program pada class **Utama** sehingga pengguna juga dapat memilih operasi lihat barang teratas, serta dapat secara bebas menentukan kapasitas gudang!

Code:

**Percobaan 2: Konversi Kode Barang ke Biner**

Hasil**:**

****

**Pertanyaan**

1. Pada method **konversiDesimalKeBiner**, ubah kondisi perulangan menjadi **while (kode != 0),** bagaimana hasilnya? Jelaskan alasannya!

Code:



Hasil:



= Hasilnya akan tetap sama, karena jika menggunakan kondisi while(kode > 0) perulangan akan berhenti jika bilangan sudah menccapai angka 0 atau dibawahnya, akan tetapi jika menggunakan kondisi while(kode != 0) maka perulangan akan berhenti saat bilangan benar” berada pada angka 0

1. Jelaskan alur kerja dari method **konversiDesimalKeBiner**!

* Tentukan bilangan desimal yang akan dikonversi.
* Lakukan operasi modulo 2 pada bilangan desimal untuk mendapatkan sisa bagi dari pembagian bilangan tersebut dengan 2.
* Masukkan sisa bagi tersebut ke dalam stack.
* Bagi bilangan desimal dengan 2 untuk mendapatkan hasil pembagian selanjutnya.
* Ulangi proses ini sampai bilangan desimal menjadi 0.

**Percobaan 3: Konversi Notasi Infix ke Postfix**

Hasil:



**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?

= Return value bernilai sama karena bebrapa operator memiliki tingkatan yang sama, menjadikan mempunyai return value yang sama, jika diubah setiap case nyam aka membuat operator yang memiliki tingkatan yang sama menjadi berbeda

1. Jelaskan alur kerja method **konversi**!

* Jika token adalah operand (bilangan), tambahkan langsung ke ekspresi postfix.
* Jika token adalah operator:
* Jika tumpukan kosong, atau operator pada tumpukan memiliki prioritas lebih rendah dari operator saat ini, tambahkan operator saat ini ke tumpukan.
* Jika operator pada tumpukan memiliki prioritas lebih tinggi atau sama dengan operator saat ini, keluarkan operator dari tumpukan dan tambahkan ke ekspresi postfix. Lakukan hal ini berulang kali sampai operator saat ini dapat ditambahkan ke tumpukan.
* Tambahkan operator saat ini ke tumpukan setelah selesai memprosesnya.

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



= Digunakan untuk mengambil character dari variable Q pada index i

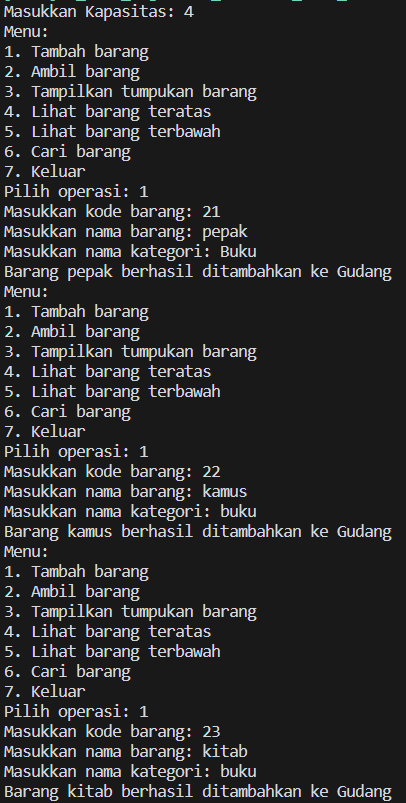
**Latihan Praktikum**

Perhatikan dan gunakan kembali kode program pada Percobaan 1. Tambahkan dua method berikut

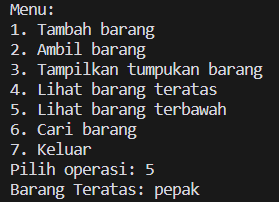
pada class Gudang:

* Method **lihatBarangTerbawah** digunakan untuk mengecek barang pada tumpukan terbawah
* Method **cariBarang** digunakan untuk mencari ada atau tidaknya barang berdasarkan kode barangnya atau nama barangnya

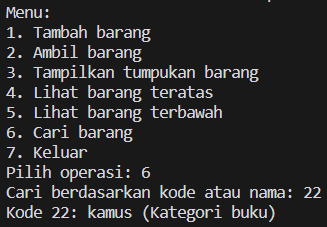
Data:



Lihat barang terbawah:



Cari barang:



**Link Github**: <https://github.com/GhoffarFitassin/P_Algoritma_Struktur_Data/tree/main/jobsheet7>