Ülesanne Laoarvestus

Teie sõber peab ladu, mille haldamine on siiani toimunud paberi peal. Sõber soovib lihtsat laohaldustarkvara (warehouse management system), mis võimaldaks lattu saabuvaid tooteid registreerida ning tooteid laost üles otsida.

Toode (**Product**) on ese, millel on nimi ja suurus. Sama nime ja sama suurusega tooteid käsitleme samaväärsetena. Koostage klass **Product** ning selles defineerige ümber (*override*) baasklassist **Object** päritud meetodid **equals()** ja **hashCode()**. Testige hulga (nt. **HashSet**) abil, kas need meetodid töötavad korrektselt.

Lao asukohti arvestatakse riiuli lõikes, igal riiulil on tekstiline nimetus (nt. A, B, C jne). Ühel riiulil võib asuda mitu toodet, samas võib nt. „Sokid suurus 42“ asuda laos mitmel erineval riiulil. Lisaks võib ühel riiulil olla sama toodet mitu tükki.

Koostada klass **Warehouse**, mis sisaldab endas kaupade asukohtadele paigutamiseks vajalikku andmestruktuuri. Realiseerida järgmised meetodid:

* **receive(prod, loc)** – lisab lattu uue toote prod ning paigutab selle asukohale loc.
* **dispatch(prod)** - otsib laost etteantud toodet prod. Kui toode on laos, tagastatakse selle asukoht ning vastava asukohaga toode eemaldatakse laost. Kui laos on sama toodet mitu tükki, väljastatakse ikkagi 1 toode (nt. esimesel kättejuhtuval asukohal paiknev). Kui laos toodet ei ole, tagastab meetod väärtuse **null**.
* **getItemCount(prod)** - tagastab etteantud toote prod kohta, mitu tükki seda laos leidub. Nt. „Sokid nr. 42“ on laos 2 tükki.

Loodud klasside ja meetodite testimiseks kasutage järgmist testprogrammi:

//loome kaubad

Product ps1 = new Product("sokid", 42);

Product ps2 = new Product("sokid", 42);

Product pk1 = new Product("kindad", 10);

Product pk2 = new Product("kindad", 9);

Product pm = new Product("mantel", 60);

Product pp = new Product("püksid", 56);

// testime equals() meetodi toimimist:

// objekt iseendaga:

System.out.println("ps1 ?= ps1: " + ps1.equals(ps1) ); // true

// kaks sama sisuga kaupa:

System.out.println("ps1 ?= ps2: " + ps1.equals(ps2) ); // true

// sama nime, aga erineva suurusega kaup:

System.out.println("pk1 ?= pk2: " + pk1.equals(pk2) ); // false

// testime hashCode() meetodi toimimist:

Set<Product> productTestSet = new HashSet<Product>();

// Hulka lisati uus element:

System.out.println("lisan ps1: " + productTestSet.add(ps1) ); // true

// Sellise "sisuga" element juba eksisteerib, ei lisatud topelt:

System.out.println("lisan ps2: " + productTestSet.add(ps2) ); // false

// Loome uue objekti, mille "sisu" on sama, mis ps1-l. Otsing peab selle leidma

System.out.println("otsin nr.42 sokke: "+ productTestSet.contains(new Product("sokid", 42)) ); // true

// Testime lao toimimist

// lisame kaubad lattu

Warehouse w = new Warehouse();

w.receive(ps1, "A");

w.receive(pk1, "A");

w.receive(pk2, "B");

w.receive(ps2, "B");

w.receive(pm, "B");

w.receive(pp, "L");

System.out.println( ps1 + " on laos " + w.getItemCount(ps1) + " tk"); //2tk

System.out.println( pk1 + " on laos " + w.getItemCount(pk1) + " tk"); //1tk

// loome otsitava kauba

Product po1 = new Product("sokid", 42);

// väljastame laost kauba:

if (w.dispatch(po1) != null) {

System.out.println( "Väljastasin toote " + po1 + " asukohalt " + w.dispatch(po1));

} else {

System.out.println( "Toodet, " + po1 + ", ei leitud loast");

}

// kontrolliks prindime välja laoseisu ja vaatame, kas antud kauba kogus on vähenenud

System.out.println( ps1 + " on laos " + w.getItemCount(ps1) + " tk"); //1tk