Programowanie Obiektowe
Kolekcje, wyjątki i generyki
Zadanie oceniane nr 2
4-05-2022

Po zakończeniu pracy konieczne jest wgranie zmian do repozytorium (add + commit + push) w katalogu o nazwie w stylu: zadanie_oceniane_2. Fakt wgrania plików do swojego repozytorium można sprawdzić samodzielnie logując się (via www) na swoje konto i sprawdzając czy pojawiły się tam wszystkie zmiany.

"Pralnia"

Parę lat temu nie odebrałem na czas pewnego elementu garderoby który oddałem do pralni. Po kilku tygodniach gdy się po niego zgłosiłem okazało się z wspomniana szata zniknęła. Rozpoczęto zakrojone na szeroką skalę poszukiwania które zapewne trwają po dziś dzień. Na pamiątkę tego wydarzenia zasymulujemy sobie dzisiaj proces czyszczenia odzieży. Tym razem żaden obiekt nie zaginie w czeluściach pralnianych kubełków, bo będziemy pamiętać o equalsach i hashcode`ach... które oczywiście wstawione będą tylko tam gdzie jest to niezbędne.

Prace do wykonania:

1. **Stworzyć klasy i ich hierarchie odzwierciedlające wspomniane poniżej elementy (nazewnictwo dobrać wedle uznania)**
   - koszula (czystość tak/nie*, rozmiar: 35-40, guziki**, kieszeń***)
   - płaszcz (czystość tak/nie*, kieszeń***, waga: 5-10)
   - kufajka (czystość tak/nie*, rozmiar: 35-40, kieszeń***)

     * czystość – domyśnie ustawione na nie

     ** guziki
     Kolekcja guzików przyszytych do koszuli w której nie mogą się one powtarzać ale pamiętana jest kolejność ich dodawania. Będzie ich 6. Kolekcję wypełniamy przy tworzeniu koszuli.

     *** kieszeń elementu garderoby przechowuje elementy takie jak:
     - guzik(id) – dodatkowe luzem oprócz guzików przyszytych do koszuli (guziki**)
     - kartka z adresem (ulica – wstawiana z zewnątrz, nr domu 1-222)
     - granat (odbezpieczony – tak/nie – jeden na 10 jest odbezpieczony)

     Mogą się one powtarzać, mają swój indeks i będą często usuwane.

     Każdy element garderoby mający kieszeń, wypełnia ją co najwyżej 1-3 elementami, których prawdpodobieństwo wystąpienia jest takie samo. Adres to jakikolwiek ciąg znaków.

     Zakładamy że kartka z adresem jest równa innej kartce jeśli przechowywana w niej nazwa ulicy i nr domu się zgadzają. Natomiast to czy dwa guziki są sobie równe, rozpoznajemy po id.
     Pozostałe elementy są "fizyczne". 1 element = 1 miejsce w pamięci.
     Dwa obiekty (instancje klasy) reprezentują zawsze dwa różne elementy.

   - Pralnia
     Posiada pola zawierające referencje na:
     - kolekcję elementów garderoby przyjętych do prania (kolejność wstawiania nieistotna, bez powtórzeń)
     - kolekcję wypranych elementów (kolejność wstawiania istotna, bez powtórzeń)
     - coś co przechowuje relacje na lini element garderoby (klucz) z kolekcją rzeczy wyjętych z kieszeni (wartość)

     Należy w dobrze znany sposób wystawić na zewnątrz metody przeznaczone do obsługi pralni.

     - ✔ putToWash (Element garderoby)
       Wstawia element do kolekcji zawierającej elementy przyjęte do prania
     - ✔ washAll()
       - ➔ przegląda szaty przeznaczone do prania pod kątem zawartości kieszeni i opróżnia je, a zawartość wrzuca do kolekcji będącej będącej w relacji do obiektu garderoby z którego kieszeni były one wyjęte (patrz jedno z

wyżej opisanych pól Pralni)
- ➔ w przypadku gdy w kieszeni znaleziony zostanie odbezpieczony granat, wtedy jest to traktowane jak niedopuszczalna sytuacja, która zagraża bezpieczeństwu oraz w obliczu której zgodnie z kontraktem nie jest możliwe kontynuowanie prania ubrań. Zdecydowanie wymaga to widocznego zasygnalizowania przerwania tego procesu.
- ➔ Ustawia czystość na true
- ➔ Wrzuca do wypranych elementów usuwając z kolekcji ubrań przyjętych do prania

- ✔ pickUpWashedClothes()
  Zwraca wyprane elementy garderoby, usuwając je jednocześnie z kolekcji elementów wypranych.
- ✔ getPocketStuffByClothes(element garderoby)
  Dla przekazanego elementu garderoby zwraca kolekcję elementów wyjętych z kieszeni, usuwając jednocześnie całą jego relację z tą kolekcją.

- Klient
  Ma dostęp do pralni (taki żeby widział metody które są mu potrzebne i żadnych innych) bez wnikania w to co jaki byt tak naprawdę dostarcza ich implementacji.

  Posiada metodę doJob(), która tworzy po jednym elemencie garderoby każdego typu, przekazuje je do pralni po kolei, pierze wszystkie, odbiera ubrania i zawartość kieszeni. Nigdzie ich nie przechowuje. Obsługuje sytuację gdy trafi się ubranie z odbezpieczonym granatem w kieszeni.

- Świat
  Posiada metodę goLive(), która tworzy pralnię, dodaje 10-ciu klientów do indeksowalnej kolekcji bez konieczności szybkiego usuwania, listuje ją uruchamiając metodę doJob() na każdym z nich.

2. **Uczynić pralnię parametryzowalną**
Wstawić dodatkowe pole: certificate do Pralni, którego typu nie musimy znać na etapie implementacji tego przybytku. Właściwy typ (trzeba go stworzyć) będzie przekazany dopiero podczas tworzenia instancji pralni oraz przez konstruktor. Jedyna logika biznesowa to jakikolwiek ciąg znaków zwracany w ramach reprezentacji tekstowej tego obiektu (np. "hdkr4826df94kf9444jdfllhh"). Utworzyć metodę retrieveCertificate, która zwraca certyfikat po uprzednim wypisaniu go na konsoli.

3. **Zademonstrować działanie**