Programowanie Obiektowe
Podstawy języka Java
Zadanie oceniane nr 1b
2-04-2025

Po zakończeniu pracy konieczne jest wgranie zmian do repozytorium (add + commit + push) w katalogu o nazwie: Zadanie_oceniane_1b. Fakt poprawnego wgrania pracy do swojego repozytorium można zweryfikować samodzielnie poprzez zalogowanie się (via www) na swoje konto i sprawdzenie czy pojawiły się tam wszystkie zmiany, co pomoże uniknąć ewentualnej przykrej niespodzianki.

"Droga"

W dniu dzisiejszym należy stworzyć mini symulator tego co się dzieje z drogami w ciepłe dni. Modelem na którym będziemy pracować jest struktura reprezentująca dwuwymiarową jezdnię po której przemieszczają się pojazdy różnych typów. Na rysunku poniżej jest zaprezentowany poglądowy model obrazujący powstawanie kolein w drodze. Na potrzeby zadania puścimy wodze bajkowej wyobraźni i nie będziemy się trzymać prawdziwej asfaltowej rzeczywistości. W przypadku użycia polskiego nazewnictwa, nie używamy znaków diakrytycznych. Tym razem przymknę oko na zwracanie nulli przez metody. Pamiętamy jak najmniejszej możliwej do wykonania widoczności dla pól, metod. To samo dotyczy ich rodzaju jak element może być statyczny to niech taki będzie. Unikamy redundancji. Pamiętamy o sensownym nazewnictwie, pakietach i ładnym formatowaniu kodu -to się zawsze opłaca.

Prace do wykonania:

1. **Stworzyć hierarchię klas odzwierciedlającą elementy występujące w tym zadaniu wraz z danymi które przechowują (podane w nawiasach). Za encję przyjmiemy ograniczony obiekt (powiedzmy kawałek nawierzchni) zrobiony z asfaltu o pewnych właściwościach**

   Każdy element (o ile jest stanie to zrobić) sam ustawia swoje pola.
   Dla zakresów lub wymienionych wartości dokonjemy losowania. O ile nie ma dodatkowych wymagań, zakłada się że opcje losowane są jednakowo prawdopodobne.

   - Asfalt miękki (lepiszcze)
     nazwa: 100/150, bazowa wysokość elementu = 100, aktualna wysokość elementu, temperatura
     Reprezentacja tekstowa: -aktualna wysokość elementu- (liczba całkowita > 0)
   - Asfalt bardzo miękki (lepiszcze) (nie jest asfaltem miękkim)
     nazwa: 160/220, bazowa wysokość elementu = 100, aktualna wysokość elementu 100, temperatura
     Reprezentacja tekstowa: -aktualna wysokość elementu- (liczba całkowita > 0)
   - Asfalt ścieralny
     nazwa: 50/70, bazowa wysokość elementu = 80, aktualna wysokość elementu, temperatura
     Reprezentacja tekstowa: {aktualna wysokość elementu} (liczba całkowita > 0)
   - Asfalt twardy
     nazwa: 20/30, bazowa wysokość elementu = 60, aktualna wysokość elementu, temperatura
     Reprezentacja tekstowa: [aktualna wysokość elementu] (liczba całkowita > 0)

   Należy wkomponować w hierarchię metodę runOver która przyjmuje jako argument nacisk (masę) i ustawia nową wysokość elementu nierzeczywiście symulując jego odkształcenie:
   Niech bazowa wysokość elementu = BWE
   Dla:
   - ➔ asfalt twardy BWE - pierwiastek (1/ (BWE + 1)*masa*temperatura)/10000
   - ➔ asfalt ścieralny BWE - pierwiastek (1/(BWE + 1)*masa*temperatura)/7000
   - ➔ asfalt miękki BWE - pierwiastek (1/ (BWE + 1)*masa*temperatura)/100
   - ➔ asfalt bardzo miękki BWE - pierwiastek (1/(BWE + 1)*masa*temperatura)/100

2. **Stworzyć klasę generującą, przechowującą i zarządzającą symulacją**
   Posiada ona:
   - podczas tworzenia przekazywane są parametry: szerokość jezdni oraz sposób wypełnienia asfaltem (Twardy, Twardy + Ścieralny albo Twardy + Ścieralny + Miks "Lepiszczów") a także temperatura.

- ✔ przyjmujemy że wymaga ona aby wartości szerokości były z przedziału <20;50>, bo tylko wtedy jest w stanie sensownie zainicjować obiekt. Oczywiście zakładamy że może się tak zdarzyć, iż przekazane do niej wartości mogą być większe ale wtedy inicjalizacja nie będzie mogła być kontynuowana, bo nie są to okoliczności w przypadku których metoda zobowiązana nas do dalszego działania (nie poczuwa się do tego żeby to kontynuować) i jasno to zaznacza.
- ✔ W zależności od wybranego sposobu wypełnienia asfaltem tworzy jezdnię o długości 50 stworzoną z konkretnego rodzaju podłoża.

- Posiada metodę print wypisującą elementy na konsoli tak żeby jeznia była widoczna jako prostokąt (tak jak na poglądowym rysunku z tym że zamiast kolorów mamy cyferki przedstawiające grubość jej komórek).
- Posiada metodę generującą n samochodów które jadą po jezdni (każdy raz) a na koniec wypisuje zawartość na konsolę za pomocę w/w metody.

3. **Stworzyć klasę reprezentującą pojazd która:**
   - posiada referencję na model jezdni przekazaną podczas tworzenia jego instancji
   - posiada masę i liczbę kół oraz szerokość osi (jako przesunięcie indeksu tablicy)
   - posiada metodę "go", która:
     1) przyjmuje jako argument indeks dla lewego koła
     2) realizuje przejazd jednego koła pojazdu po danej komórce jako wywołanie na niej metody runOver a jako parametr wstawia masę jaka przypada na koło
     3) "przejeżdża" lewą połową kół na całej długości jezdni i to samo robi dla prawej połowy kół, z tym że ślad jest oddalony o index będący szerokością osi. Pamiętamy że masę dzielimy na koło.

     W ten sposób symulujemy przejazd danego pojazdu.

4. **Stworzyć klasę reprezentującą rzeczywisty pojazd który:**
   - umie wszystko co pojazd zdefiniowany powyżej
   - z P=0.1 zaburza indeks o jeden w lewo lub w prawo (losowo)
   - uzywa jednej instancji klasy Random wspólnej z innymi pojazdami

5. **Stworzyć klasę reprezentującą samochód osobowy która:**
   - umie wszystko co poj. rzeczywisty ma 4 koła i waży 1000-1500 kg (losujemy)

6. **Stworzyć klasę reprezentującą samochód ciężarowy która:**
   - umie wszystko co poj. rzeczywisty ma 8 kół i waży 8000-12000 kg (losujemy)

7. **Demonstracja działania**

Pozwolić na przejazd pewnej liczbie samochodów w ramach symulacji przy zadanej temperaturze i ilości samochodów tak żeby na wyświetlonej na konsoli jezdni widać było koleiny (mniejsze wartości grubości).