Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Základy objektovo orientovaného programovania

Zadanie č.2 – Použité OOP princípy

Akademický rok 2021/2022

Meno: Ján Ágh
Cvičiaci: Mgr. Marián Potočný

Dátum: 14.11.2021
Počet strán: 2

Ropný vrt – Použité OOP princípy

V mojom projekte sa nachádzajú nasledovné OOP princípy:

Organizácia tried do balíkov – môj projekt pozostáva zo štyroch balíkov (*oilwell.mainpkg*, *oilwell.personnelpkg*, *oilwell.machinerypkg* a *oilwell.enumpkg*)

Dedenie – v balíku *oilwell.machinerypkg* (triedy *Crane.java*, *Drill.java* a *WaterPump.java* dedia z triedy *Machinery.java*) a tiež v balíku *oilwell.personnelpkg* (triedy *Repairman.java* a *Worker.java* dedia z triedy *Personnel.java*)

Viacnásobné dedenie – v balíku *oilwell.machinerypkg* (napr. trieda *Crane.java* dedí z triedy *Machinery.java* a tá zas dedí z triedy *Tools.java*)

Zapuzdrenie a modifikátory prístupu – všetky triedy obsahujú privátne premenné spolu so setter a getter metódami (príklad – trieda *Machinery.java* – všetky premenné sú privátne a tie, ktorých hodnoty sa počas behu programu môžu meniť, majú vytvorené vlastné getter a setter metódy (premenná *private int amountOfFuel*, hodnotu premennej získame pomocou *public int getFuel()* na riadku 68 a premennú môžeme meniť pomocou *public void setFuel()* na riadku 44))

Preťažovanie metód alebo konštruktorov – preťažovanie metód v triede *Repairman.java* (metódy *public void refillFuelTank(Machinery machine, Repairman helper)* na riadku 14 a *public void refillFuelTank()* na riadku 26) a preťažovanie konštruktorov v triede *Machinery.java* na riadkoch 21 a 25

Prekonávanie metód – metóda *public void machineIsBroken()* v generickej nadtriede *Machinery.java* je prekonaná v podtriedach *Crane.java*, *Drill.java* a *WaterPump.java* (metóda momentálne ešte nie je súčasťou samotného programu po spustení, nakoľko funkcionalita kazenia sa strojov ešte nebola v projekte implementovaná)

Kompozícia – napr. medzi triedami *Crane.java* a *MachineManager.java* alebo medzi *Worker.java* a *PersonnelManager.java* (objekt typov *Crane* alebo *Worker* nemôže existovať mimo *MachineManager* a *PersonnelManager*, ak tieto objekty zaniknú, zaniknú aj objekty *Crane* a *Worker*)

Agregácia – napr. medzi triedami *Machinery.java* a *Personnel.java* (objekt typu *Machinery* sa nachádza v *Personnel*, ale môže existovať aj mimo neho)

Asociácia – napr. jednostranná asociácia medzi *Crane.java* a *Drill.java* (objekt typu *Crane* má prístup k objektu typu *Drill*, ale nie naopak)

Ropný vrt – UML diagram