Zaawansowane Programowanie Obiektowe i Funkcyjne
Refleksje

Zadanie oceniane nr 4a

28-11-2022

Zadania znajdują się w repozytorium (każdy ma swoje konto w podkatalogu zpoif_zadanie_4a). Należy zsynchronizować się za pomocą polecenia git pull. Po zakończeniu pracy konieczne jest wgranie zmian do repozytorium (commit + push). Proszę pamiętać o poleceniu add.

"Budka 2"

Poniższe punkty zadania wykonać **za pomocą mechanizmu** refleksji dostarczonego przez standardową bibliotekę Java. **Nie wolno** używać operatora "**new**", "**instanceof**" ani **niczego zmieniać** w pakiecie: `pl.edu.pw.mini.zpoif.task4b` (chyba że podpunkt zadania będzie tego wymagał wprost). **Można** stosować casting typów.

Kod rozwiązania i jego demonstracji należy umieścić w pakiecie: `pl.edu.pw.mini.zpoif.task4b.solution`. Tam też znajduje się abstrakcyna klasa Solution, posiadająca metody których definicjami (w rozszerzającej ją podklasie) będą rozwiązania poszczególnych podzadań. Zatem należy utworzyć klasę dziedziczącą i tam wstawiać swój kod do odpowiednich nadpisywanych metod. Na koniec wywołać w jakimś demonstratorze metodę `demonstrate()`.

Prace do wykonania:

Tam, gdzie jest mowa o przekazanym jako argument metody obiekcie (tak żeby było na czym pracować), można go stworzyć z użyciem "new".

1. W metodzie `task1` utworzyć obiekt klasy "`pl.edu.pw.mini.zpoif.task4b.building.WygodnaBudka`" używając jej nazwy tekstowej i go zwrócić.
2. W metodzie `task2` używając przekazanego obiektu klasy WygodnaBudka wypisać na konsolę wartość pola ***`bazgrol`***.
3. W metodzie `task3` na rzecz przekazanego obiektu klasy WygodnaBudka ustawić wartość pola `szyfrDoSejfu` wartością pola ***`UNIWERSALNY_SZYFR_DO_SEJFU`***.
4. W metodzie `task4` na rzecz przekazanego obiektu klasy WygodnaBudka uruchomić jedną (pierwszą lepszą) bezparametrową metodę zwracającą Integer oraz wypisać na konsolę jej nazwę i to co zwróciła.
5. W metodzie `task5` wypisać proste nazwy (bez pakietów) wszystkich nadklas w hierarchii obiektu będącego typem pola o nazwie "`dobreWyrko`".
6. W metodzie `task6` odszukać konstruktor klasy `WygodnaBudka` , który przyjmuje dwa argumenty typu String. Za jego pomocą utworzyć i zwrócić instancję tej klasy przekazując parametry: "Super" oraz "Dobre graty".
7. W metodzie `task7` na rzecz przekazanego w demonstratorze obiektu klasy WygodnaBudka (dokładnie tej samej instancji która była przekazana do zadania 3) wywołać metodę "`open`" na obiekcie "`sejf`" i przekazać do niej wartość pola: "`szyfrDoSejfu`".
8. W metodzie `task8` z nadklasy klasy "`WygodnaBudka`" wypisać nazwy klas wewnętrznych o zasięgu protected.
9. W metodzie `task9` bazując na przekazanym do metody obiekcie, należy od sumy wartości wszystkich pól (Integer) zadeklarowanych **tylko** w nadklasie klasy `WygodnaBudka` odjąć sumę pól zadeklarowanych **tylko** w klasie `WygodnaBudka`.