Projekt GUI I - Budowa.

W ramach projektu należy utworzyć oraz zaimplementować poniższe klasy i metody.

Klasa Abstrakcyjna Osoba, powinna zawierać pola: imie,nazwisko,pesel,nrTelefonu i waga. Każdy pesel powinien być unikalny, w przypadku próby utworzenia obiektu z duplikatem peselu, należy wyrzucić nieUnikalnyPeselException.

Klasa Architekt powinna zawierać pola klasy Osoba, a także pole typu Specjalizacja.

Klasa Kopacz powinna zawierać pola klasy Osoba, a także pola: iloscMachniecLopata, czyZdolnyDoPracy (domyślnie ustawione jako True). Kopacz powinien posiadać metodę void kop(), która ma za zadanie wykonać następujące funkcje:

-w wątku ma wypisywać na ekran “Kopacz X machnął łopatą”, losową ilość razy, nie mniejszą niż 5, z losowymi odstępami czasu
-Jeżeli sumarycznie kopacz wykonał już akcję ponad 15 razy, należy przerwać wątek i wyrzucić zlamanaLopataException, z wiadomością “Łopata zużyła się i pękła”.
-Każde machnięcie łopatą ma 1% szans na przypadkowe złamanie łopaty. W takim przypadku należy wyrzucić zlamanaLopataException z wiadomością “Łopata była wadliwa i złamała się niespodziewanie w trakcie użytkowania”.
-Metoda ta powinna zwiększać o 1 wartosc pola “iloscMachniecLopataBrygady” znajdującego się w klasie Brygada. Zmiana wartości powinna być zsynchronizowana tak, aby dwóch kopaczy nie mogło jednocześnie zwiększyć jej wartości, jeżeli obecnie jeden kopacz operuje na „iloscMachniecLopataBrygady”, drugi musi zaczekać.
-Jeżeli łopata zostanie złamana, Kopacz przestaje być zdolny do pracy.

Kopacz powinien posiadać także metodę pozwalającą na przerwanie kopania.

Klasa Brygadzista, posiadająca pola i metody klasy Kopacz a także pola: pseudonim, dlugoscZmiany, brygada. Brygadzista powinien posiadać metodę sprawdzCzyBrygadaNiezdolnaDoPracy(), która w wątku (przez czas określony zmienną dlugoscZmiany) będzie sprawdzała czy w brygadzie jest chociaż jedna osoba zdolna do pracy. Jeżeli nie, należy wyświetlić stosowny wyjątek. Jeżeli wątek dobiegnie końca, należy wypisac na konsole raport.

Klasa Brygada powinna posiadać następujące pola: brygadzista, pracownicy, maksymalnaIloscPracownikow, iloscMachniecLopataBrygady. Brygada powinna posiadać także następujące metody:

-int ileArchitektów – zwróci ilość architektów w brygadzie
-boolean czyPelnaBrygada – zwróci True jeżeli ilość pracowników równa się maksymalnej ilości pracowników, False wpp.
-dodajPracownika – doda pracownika do listy pracowników
-dodajPracownikow – doda podaną listę pracowników do listy pracowników.
-setBrygadzista.

Interface IPracownik, posiadający następujące metody: int pobierzPensje, powiedzIleRazyKopales, powiedzCoRobisz, zakonczDzialanie (kończy działanie wątku), dodajSieDoBrygady(Brygada). Interface powinien być implementowany przez Kopacza, Brygadzistę i Architekta. Implementacja metod powinna być realna i optymalna.

Wymagania strukturalne:

-Należy zaimplementować interface Comparable w stosownym miejscu.
-Należy skorzystać w streamów i wyrażeń lambda.

Ogólne uwagi do projektu:
-Implementacje powyższych obiektów powinny być optymalne i możliwie realnie odzwierciedlać
-Rozwiązania, metody lub pola nie opisane w zadaniu należy dodać zgodnie z własnym rozsądkiem.

Sprawdzenie:

Należy stworzyć klasę Main wraz z metodą main i użyć zaimplementowanych metod w jasny i czytelny sposób.

Data oddania:

Projekt należy wysłać przez Teams Assignments do 23:59 28.04.2022. 29.05.2022 odbędą się obrony projektów na zajęciach stacjonarnych.