**PROJEKT „BANKOMAT” – INSTRUKCJA**

Przed wykonaniem jakichkolwiek czynności w projekcie wypełnij krótką ankietę złożoną z trzech prostych pytań dotyczących samooceny w zakresie umiejętności programowania i testowania. Link do ankiety:

<https://forms.gle/6DoyZfzuY1skFt7N8>

Zadanie polega na implementacji kodu obsługującego działanie bankomatu. Projekt możesz pobrać pod adresem:

[**https://github.com/michaelmnich/Bankomat/**](https://github.com/michaelmnich/Bankomat/)

W celu wykonania zadania **musisz zaimplementować każdy interfejs** znajdujący się w katalogu Bankomat/src/com/uj/atm/interfaces/ (za wyjątkiem IDummySample, który jest umieszczony w projekcie w celach przykładowych – patrz poniższa sekcja „Jak napisać przykładowy test?”). Interfejsy, które należy zaimplementować to:

* **IAccount** – interfejs zawierający metody obsługujące konto bankowe
* **IAtm** – interfejs zawierający metody obsługujące działanie bankomatu
* **ICreditCard** – interfejs zawierający metody obsługujące funkcje karty płatniczej

Nie wolno tworzyć żadnej innej metody publicznej, która nie implementuje interfejsu.  Każda z metod w interfejsie posiada opis funkcjonalny, na którego podstawie powinieneś/powinnaś wykonać implementację danej metody.

Integralną częścią projektu jest stworzenie testów do zaimplementowanych przez Ciebie metod interfejsu.

**Projekt**

Projekt został stworzony w środowisku InteliJ. W celu pobrania darmowej wersji dla uczniów należy zalogować się mailem w domenie edu:

<https://www.jetbrains.com/community/education/#students>

**Testy**

**Każda z metod w interfejsie** **musi mieć do siebie napisane testy**, które będą testowały poprawność zaimplementowanych metod. Pamiętaj, aby testy nie były „trywialne”, tzn. nie sprawdzały wyłącznie oczywistych rzeczy. Testy powinny być na tyle mocne, aby być w stanie wykryć jak najwięcej możliwych błędów, jakie mógłby popełnić programista podczas implementacji metod.

Testy musisz wykonać w technologii **JUnit 4**. Możesz do tego celu przerobić projekt na projekt Mavenowy i użyć JUnita dla maven:

<https://mvnrepository.com/artifact/junit/junit/4.12>

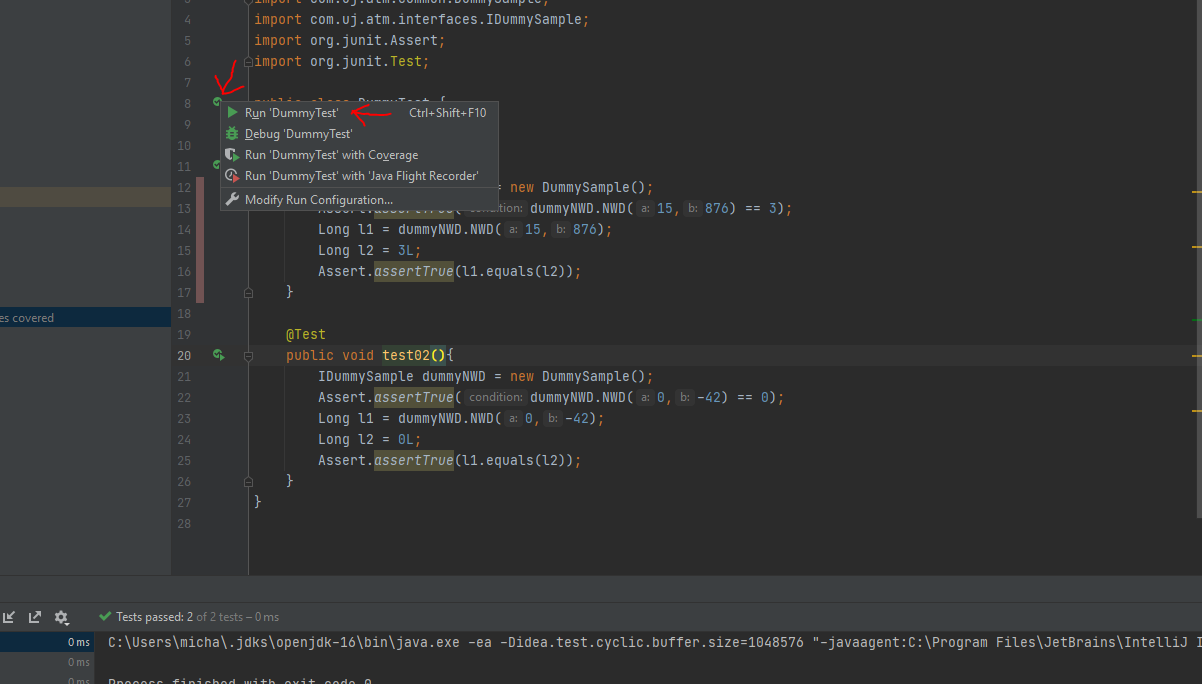
lub użyć JUnita w projekcie InteliJ – przykład jest opisany w filmiku pod adresem:

<https://www.youtube.com/watch?v=Bld3644bIAo&ab_channel=BrianFraser>

Testy muszą odnosić się **tylko do metod z interfejsu**. Celem zadania jest wykonanie testów, które będą kompatybilne w stosunku do innego projektu stosującego ten sam interfejs.  Testy mogą badać poprawność działania całego programu, ale tylko na podstawie metod z interfejsu.

**Jak napisać przykładowy test?**

W projekcie został dodany przykład prezentujący mechanikę testowania jednostkowego, który zawiera testy operujące tylko i wyłącznie na metodach interfejsów. Testy zostały napisane do klasy ‘DummySample’ implementujacej interfejs ‘IDummySample’ . Testy znajdują się w klasie ‘DummyTest’. Zeby uruchomić testy należy wcisnać w inteliJ prawy klawisz myszki na zielonym guziku. Przykład poniżej.



W projekcie stworzono trzy przykładowe testy testujące metodę NWD zwracającą największy wspólny dzielnik dwóch liczb. Test test01 sprawdza typową sytuację – obliczenie NWD dla dwóch liczb, które nie są względnie pierwsze, tzn. których największy wspólny dzielnik jest większy od 1. Test test02 sprawdza sytuację, w których pierwsza liczba wynosi 0. Test test03 sprawdza sytuację, w której jedna liczba jest wielokrotnością drugiej. W każdym z testów znajdują się dwie asercje – jedna sprawdza równość wprost na wartościach liczbowych, a druga – analogiczną równość ale na wartościach zmiennych, do których podstawiono odpowiednie liczby. Testy mogą w ogólności posiadać jedną lub więcej asercji.

**Wymagania biznesowe**

System składa się z obiektów trzech typów: **konto**, **karta kredytowa**, **bankomat**.

**Konto**. Każde konto posiada nieujemne saldo.

**Karta kredytowa**. Każda karta kredytowa posiada swój kod PIN i może być stowarzyszona z co najwyżej jednym kontem. Po stowarzyszeniu karty z kontem nie jest możliwa zmiana tego konta (stowarzyszenie z kontem jest permanentne). Pin jest wartością numeryczną złożoną z dokładnie czterech znaków, np. 1234. Domyślny PIN dla nowej karty to 0000.

**Bankomat**. Bankomat realizuje operacje: 1) logowania się przy pomocy kodu PIN użytej karty, 2) sprawdzania salda na koncie stowarzyszonym z daną kartą, 3) możliwość zmiany kodu PIN na karcie na inny kod PIN, 4) funkcję wpłatomatu realizującą wpłatę na konto stowarzyszone z daną kartą, 5) funkcję wypłaty pieniędzy z konta stowarzyszonego z daną kartą, 6) funkcję przelewu środków z konta stowarzyszonego z kartą na inne konto.

**Uwagi ogólne dotyczące funkcjonalności systemu**

Wszystkie funkcjonalności systemu są opisane w komentarzach do kodu interfejsu. Podczas pisania testów korzystaj tylko i wyłącznie z metod dostępnych w interfejsie (czyli tak, aby Twoje testy mogły być bezpośrednio wykonane na analogicznym projekcie zaimplementowanym przez inną osobę). W klasie main nie wpisuj nic.

Realizacja każdej z dostępnych funkcjonalności powinna powodować poprawność stanu wszystkich obiektów związanych z daną operacją. Na przykład: przelew z konta stowarzyszonego z daną kartą na inne powinien powodować zmniejszenie salda pierwszego konta i powiększenie salda drugiego konta o tę samą wartość; udana wypłata danej kwoty w bankomacie powinna skutkować zmniejszeniem salda konta stowarzyszonego z kartą, przy pomocy której operacja jest realizowana, o tę samą kwotę; z konta o ujemnym saldzie nie można wypłacać pieniędzy; itd. Generalnie, powinny być zachowane (i przetestowane) wszystkie „zdroworozsądkowe” zasady korzystania z kart, kont i bankomatów znane Ci z codziennej praktyki.

**Sposób oddawania zadań i zasady zaliczenia projektu**

Aby rozwiązanie tego zadania było zaliczone jako ocena z egzaminu, projekt musi spełniać wszystkie wymagania opisane powyżej. Aby sprawdzić, czy Twój projekt je spełnia, możesz wykorzystać serwis znajdujący się pod adresem: XXXXXX. System automatycznie poinformuje Cię, czy wszystkie wymogi są spełnione, a jeśli nie, to poinfomuje Cię co należy poprawić.

Gdy upewnisz się, że projekt jest poprawny, wyślij go [TU ADRES ORAZ SPOSÓB WYSYŁANIA PROJEKTU]. **Uwaga – oddanie programu, który nie przejdzie weryfikacji przez serwis XXXXXX będzie skutkowało niezaliczeniem projektu i koniecznością zdawania egzaminu**!

[MICHAŁ SPRAWDŹ TO I EW. POPRAW, TAK ŻEBY NIE BYŁO ŻADNYCH TECHNICZNYCH PROBLEMÓW Z EKSPERYMENTEM KRZYŻOWYM tu wpisać instrukcję dot. tego gdzie student może sprawdzić poprawność „techniczną” swojego projektu – serwis który zrobi Jarek na frege; serwis powinien sprawdzać: 1) czy projekt zawiera tylko trzy interfejsy, 2) czy interfejsy zawierają tylko te metody, co trzeba, 3) czy każda metoda posiada jakąś (niepustą) implementację, 4) czy projekt zawiera co najmniej jeden test dla każdej klasy, 5) czy funkcja main jest pusta, 6) czy można odpalić testy studenta na ‘domyślnym’ projekcie (który będzie symulowa projekt innego studenta), 7) czy testy ‘domyślnego’ projektu można odpalić na kodzie studenta]

**Uwagi końcowe**

Podczas wykonywania zadania możesz korzystać z informacji (np. znalezionych w sieci) dotyczących kwestii technicznych, programistycznych, tego jak wykorzystać dane narzędzie itp. **Ważne jest jednak, abyś zadanie wykonał/a samodzielnie. Nie wyszukuj w Internecie przykładowych rozwiązań implementacji kodu bądź testów w podobnych projektach – kod i testy twórz samodzielnie. Nie konsultuj się z innymi studentami, którzy również biorą udział w tym zadaniu. Nie przesyłaj innym studentom żadnych informacji, jakie uzyskasz od osób prowadzących eksperyment**. Jeśli będziesz mieć jakieś problemy podczas wykonywania zadania i będziesz potrzebował/a pomocy, zwróć się do osoby nadzorującej eksperyment (dr Michał Mnich). Wyniki uzyskane przez studentów będą podlegały badaniom naukowym i dlatego ważne jest, aby nie były zaburzone poprzez komunikowanie się z innymi osobami biorącymi udział w eksperymencie.

**Informacja o ochronie danych osobowych**

**Na potrzeby badania naukowego związanego z niniejszym projektem każdy student będzie proszony o wypełnienie imiennej ankiety zawierającej kilka pytań (np. o doświadczenie w programowaniu lub o średnią ocen z poprzedniego roku akademickiego), a napisany przez niego kod również nie będzie anonimowy lecz przypisany do konkretnego studenta.**

**Jednak informacje te będą użyte wyłącznie na potrzeby technicznego opracowania obliczeń związanych z rezultatami zadania. Wyniki ankiety nie będą miały żadnego wpływu na ocenę pracy studenta, a wszelkie publikowane wyniki badań będą ANONIMIZOWANE, co oznacza, że w przypadku jakiejkolwiek publikacji wyników badań opartych na niniejszym projekcie ŻADNE dane osobowe studentów biorących w nim udział nie będą NIGDY ujawnione.**