Podczas pracy nad różnymi systemami informatycznymi, programiści często spotykają się z problemem obsługi błędów w programie. Czasem podczas tworzenia rozwiązania programista nie przewidział pewnego skrajnego przypadku który powoduje błąd. Czasem przed tym błędem nie da się sensownie zabezpieczyć. W każdym przypadku programista powinien mieć możliwość analizowania i wykrywania błędów w swoim rozwiązaniu.

Przykłady błędów na których wystąpienie programista nie ma wpływu:

- Utracono połączenie z bazą danych (np. nastąpił restart serwera bazy danych, albo wykonuje się rollback z kopii zapasowej itd.)
- Brak uprawnień do zasobów bazodanowych
- Utracono połączenie z serwisem z którego ściągane są dane programu (np. gdy klient utracił chwilowo połączenie z internetem, albo serwis z jakiegoś powodu przestał odpowiadać)
- Plik jest używany przez inny proces i nie da się go w danej chwili otworzyć ani zapisać
- Brak pliku w podanej ścieżce

Twoim zadaniem jest przygotowanie klasy SafeInvoker z metodą o nazwie invoke, która będzie dawała możliwość bezpiecznego wywołania jakiejkolwiek operacji potencjalnie wyrzucającej wyjątki.

```
new SafeInvoker.invoke(() -> doSomethingDangerous());
```

W zależności od rodzaju wyjątku jaki może zostać wyrzucony, obiekt ww. klasy będzie dawał możliwość:

- kilkukrotnego wywołania operacji
- odczekania pewnego odstępu czasu między kolejnymi wywołaniami operacji
- wykonania pewnej wcześniej zadeklarowanej akcji, gdy limit prób wykonania operacji został przekroczony
- logowania informacji o błędzie do pliku z logami (należy wybrać sobie jakiś framework do logowania np. Log4J)

Do rozwiązania zadania przyda się także napisać prosty mechanizm rejestrujący wyjątki oraz strategie na ich wystąpienie.

Należy przygotować zestaw testów jednostkowych, sprawdzających poprawność rozwiązania.