Wrocław, 12 maja 2024

Platformy **P**rogramistyczne `.NET` i `Java`

Laboratorium 6

Projekt własnej aplikacji w języku `Java`

prowadzący: *Dr inż. Radosław Idzikowski, mgr inż. Michał Jaroszczuk*

1 Cel laboratorium

Celem laboratorium jest wykonanie własnego projektu w języku `Java`. Doświadczenie dydaktyczne pokazuje, że największy przyrost wiedzy i umiejętności jest wtedy, kiedy tematyka zadania nie jest narzucona z góry, lecz kiedy stanowi przedmiot zainteresowań studenta. Pomimo dowolności wyboru tematu, narzucone zostaną pewne założenia do spełnienia, związane głównie z technologią, w jakiej powinien być wykonany projekt oraz jej możliwościami.

2 Zadania i zasady zaliczenia

W ramach projektu należy:

- Zaproponować tematykę projektu i określić kluczowe założenia i wymagania.
  Na pierwszych zajęciach z niniejszego projektu należy odbyć rozmowę z prowadzącym w celu spisania ustaleń i wymogów dotyczących programu.
- Wykonać projekt w języku `Java` trzymając się przyjętych wcześniej założeń lub zmieniając je po konsultacji z prowadzącym.
- Zaprezentować projekt na ostatnich zajęciach przez co rozumie się przedstawienie działania aplikacji, przygotowanie dokumentacji na temat projektu, odpowiedzenie na pytania zadane przez prowadzącego.

Przy ocenie projektu będzie brane pod uwagę:

1. OCENA 3: Spełnienie początkowych założeń, poprawne działanie projektu oraz zastosowanie przynajmniej dwóch z poniższych możliwości technologii Java:
   - Połączenie z bazą danych (lokalny serwer bazodanowy),
   - Komunikacja z zewnętrznym API,
   - Aplikacja webowa,
   - Aplikacja okienkowa,
   - Obsługa wielowątkowości.
2. OCENA 4: Ogólna czytelność programu i przestrzeganie założeń *clean code*, stosowanie paradygmatów programowania obiektowego, sensowność zaimplementowanych akcji i koncepcji aplikacji, zastosowanie dodatkowych mechanizmów (zagnieżdżenie komponentu webowego, mechanizm logowania, mechanizmy przesyłu danych, itp.).

3. Stosowanie mechanizmów kontroli błędów (walidacje), dodanie w projekcie testów jednostkowych, przeprowadzenie testów manualnych programu (odpowiednio udokumentowanych), zastosowanie więcej niż dwóch elementów wymienionych w punkcie 1.

4. OCENA 5,5: Uwzględnienie naukowych aspektów w praktycznym zastosowaniu (np. implementacja wybranego algorytmu do rozwiązania danego problemu)

Ponieważ ocena projektów o dowolnej tematyce jest oceną bardziej subiektywną niż ocena z góry narzuconego rozwiązania, w projekcie oceniane będzie przede wszystkim podejście studenta do wybranego tematu oraz spełnienie podanych założeń. Za spełnienie założeń projektu i wykorzystanie wybranych technologii, student otrzyma ocenę dostateczną (3.0), za spełnienie punktu drugiego - ocenę dobrą (4.0), a za spełnienie wszystkich 3 punktów ocenę bardzo dobrą (5.0).

3 Tematyka

Tematyka projektu jest dowolna, natomiast dla ułatwienia poniżej podano przykładowe kierunki w których można rozwijać projekt:

- *wykrywanie kolizji* – obsługa wątków oraz graficzna wizualizacja: `Balls`, `Snake`, `Pacman`;
- *synchronizacja wątków* – symulacja (`Snails`) lub alg. optymalizacyjny (`Brute`, `Genetic`);
- *obsługa API* – pobieranie, przechowywanie i wizualizacja danych z API (`Weather`);
- *web API* – implementacja własnego API (`Distances`) ;

Poniżej podano również kilka ciekawych propozycji projektów na ocenę `5.0` i wyższą:

- `Balls` (*wykrywanie kolizji*) – aplikacja symulująca ruch piłek w 2D wraz z wizualizacją (np.: `swing`). Wątki odpowiadają ze sterowanie piłkami, kilka piłek na jeden wątek. Piłki muszą odbijać się zarówno od ścian jak od siebie nawzajem.
- `Snake` (*wykrywanie kolizji*) – zmodyfikowana gra w wężyka. Należy dodać wątki odpowiedzialne zarówno za ruchome jedzenie i przeszkody.
- `Pacman` (*wykrywanie kolizji*) – klasyczna gra z obsługą duszków za pomocą wątków.
- `Snail` (*synchronizacja wątków*) – aplikacja symulująca życie ślimaków wraz z wizualizacją. Ślimaki mają się poruszać po siatce, jeśli znajdą się na polu z trawą to ja zjadają. Na jednym polu w tym samym czasie może znajdować się tylko jeden ślimak. Trawa może wyrosnąć jedynie na pustym polu sąsiadującym z innym polem z trawą. Za każdego ślimaka ma odpowiadać dokładnie jeden wątek, ponadto jeden wątek ma obsługiwać rozrost trawy, a kolejny samą wizualizację.
- `Weather` (*obsługa API*) – aplikacja korzystająca z API pogodowego. Dane historyczne należy przechowywać w bazie danych (np.: `hibernate`) oraz wyświetlać w formie graficznej (wykresy).
- `Brute` (*synchronizacja wątków*) – równoległa implementacja algorytmu przeglądu zupełnego dla dowolnego problemu optymalizacji dyskretniej.
- `Genetic` (*synchronizacja wątków*) – równoległa implementacja algorytmu genetycznego dla dowolnego problemu optymalizacji dyskretniej.

- `Dinstances` (*web API*) – stworzenie dwóch aplikacji (serwer, klient) – aplikacja po stronie serwera ma generować losową "mapę", tzn. losujemy pozycją (x,y) lub (lat,lng) dla n miast oraz udostępniać dwa interfejsy: (1) zwrócenie położenia miasta na podstawie nazwy, (2) zwrócenie odległości między dwoma miastami. W kliencie ma być wizualizacja położenia wybranych miast oraz policzenia długości ścieżki miedzy nimi w zadanej przez użytkownika kolejności.c

- `ERP` (*web API*) Stworzenie systemu ERP, który będzie w stanie przyjąć od użytkownika nową operację produkcyjną, wygnerować dokument zlecenia produkcyjnego, wygenerować przesunięcie wyprodukowanego elementu na magazyn i zapisywać wszystkie operacje w bazie danych. Należy przyjąć różne metody harmonogramowania produkcji - FIFO, LIFO, optymalizacja zleceń produkcyjnych. Można tutaj zamodelować dowolny problem jednomaszynowy (np. RPQ) i sprawdzić różne ograniczenia.

- `Stock` (*obsługa API*) Należy stworzyć aplikację wspomagającą analizę trendów giełdowych / kursów walut na podstawie danych historycznych pobranych z API. Program powinien umożliwiać dokonanie predykcji zachowania się danego kursu na podstawie danych z podanego przez użytkownika przedziału czasowego z wykorzystanie ekstrapolacji funkcji. W programie powinno być możliwe generowanie różnych typów wykresów i zapis historii notowań w bazie danych.

- `System to manage...` (*Database*) Zadaniem jest stworzenie systemu pozwalającego na obsługę danej instytucji / danego procesu. Może być to proces zakupu biletów kolejowych, obsługa biblioteki, obsługa przychodni, itd. W zadaniu należy wykorzystać bazę danych oraz zaimplementować przejrzysty interfejs dla użytkownika z wykorzystaniem technologii frontendowych. Jako aspekt naukowy można potraktować analizę operacji przeprowadzanych przez użytkowników.

- `Accounting system` (*API*) Chcąc prowadzić sprzedaż międzynarodową i stosować 0% podatku VAT, sprzedający musi upewnić się, że jego kontrahent figuruje w bazie VIES. W przypadku dużych systemów, często takie sprawdzenie dotyczy wszystkich kontrahentów i następuje poza godzinami pracy firmy. Zadaniem będzie napisanie systemu do równoległego przeprocesowania zadanych instancji z fakturami i obliczenia należnej stawki podatku VAT w zależności od tego czy kontrahent figuruje w rejestrze VIES czy nie. Stworzony interfejs, powinien być przeznaczony do badań, a nie do rzeczywistego użycia (powinien umożliwiać generowanie instancji zamówień, zaznaczenie procenta partnerów dla których VAT ID jest ważny, wybór liczby wątków i zwracanie wyników).