Projekt PROI- 22L Oddział banku

Maciej Bieńkowski i Kacper Murygin

1.Cel i opis projektu

Projekt jest symulacją pracy oddziału banku. Symulacja bierze pod uwagę szereg czynników takich jak liczba klientów (indywidualnych lub biznesowych) zarejestrowanych przez bank, liczba pracowników, czas obsługi w zależności od wybranej usługi w oddziale. Użytkownik symulacji wprowadza dane zawierające informacje o ilościach klientów, pracownikach, długości symulacji.

Symulacja umożliwia analizę przepustowości banku w zależności od czynników. Celem oddziału banku jest sprawna obsługa klientów przy minimalnym zatrudnieniu pracowników. Bank odwiedzany jest w zależności od pory dnia i dnia tygodnia przez różną średnią liczbę klientów. Zmienia się również stosunek obsługiwanych klientów indywidualnych do biznesowych, ponieważ ci pierwsi odwiedzają oddział zazwyczaj w godzinach porannych lub wieczornych. Symulacja pomogłaby w efektywnym zarządzaniu bankiem. Pomogłaby oszacować optymalną liczbę otwartych kas oraz pracowników dostępnych w oddziale.

2. Opis wszystkich założeń przyjętych przez zespół

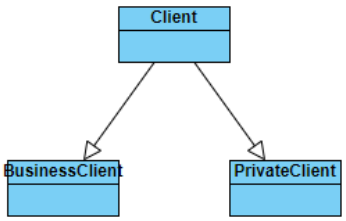
Definicja podmiotów:

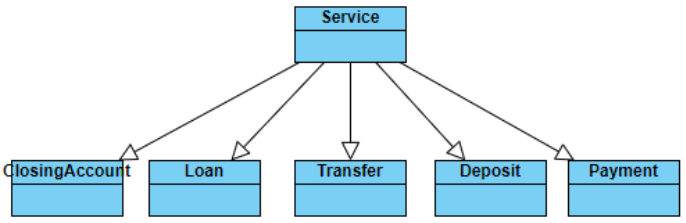
- Klient - osoba z kontem w banku która może wykonać dokładnie jedną z dostępnych akcji. Klient posiada dokładnie 1 konto.
- Kolejka - klienci ustawiają się w kolejce do każdej ze stanowisk i maszyn
- Maszyna - obsługuje klienta bez pracownika; wpłatomat, bankomat, informacja,
- Stanowisko - obsługuje klienta przy pomocy pracownika ludzkiego.
- Worker - osoba lub maszyna która obsługuje klientów.
- Akcje - wpłata pieniędzy, wypłata, pożyczka, depozyt, przelew, zamknięcie konta, informacja. Nie do każdej akcji jest potrzebny pracownik ludzki.

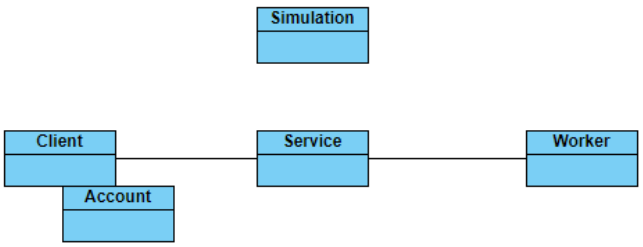
3.Podział na klasy, hierarchia klas i relacja między klasami

Klasy

- ListService - który zawiera listę serwisów
    - Service
        - Transfer
        - Loan
        - Deposit
        - Payment
        - ClosingAccount
- Simulation - symulacja banku
- AllClients - lista klientów
    - Client
        - BusinessClient
            Account
        - PrivateClient
            Account
- Machine
    - Counter
- Random
    - RandomData
    - RandomClients
    - RandomWorkers
- AllWorkers - vector pracowników
    - Worker







4. Instrukcja użytkownika

Istnieją dwie możliwości uruchomienia symulacji- poprzez podanie argumentów w linii poleceń lub poprzez argumenty w pliku tekstowym. W zależności od sposobu podania argumentów symulacji, zmienia się liczba niezbędnych i wymaganych argumentów w linii poleceń.

Argumenty- oba przypadki:

1. Ścieżka do pliku wykonywalnego- bezwzględna lub względna
2. Decyzja o sposobie podania argumentów:
   1 - plik
   2 - linia poleceń

Kolejne argumenty- plik:

3. Ścieżka do pliku tekstowego z argumentami- bezwzględna lub względna (string)
4. Ścieżka do pliku tekstowego do wyników symulacji- bezwzględna lub względna (string)

Kolejne argumenty- linia:

3. Ścieżka do pliku tekstowego do wyników symulacji- bezwzględna lub względna (string)
4. Czas symulacji (liczba iteracji) (liczba całkowita nieujemna)
5. Liczba pracowników (liczba całkowita nieujemna)
6. Liczba klientów prywatnych (liczba całkowita nieujemna)
7. Liczba klientów biznesowych (liczba całkowita nieujemna)

Plik z argumentami- struktura:
Cztery oddzielone od siebie spacjami liczby całkowite nieujemne:

1) czas symulacji (liczba iteracji)
2) liczba pracowników
3) liczba klientów prywatnych
4) liczba klientów biznesowych

5. Wskazanie wykorzystanych elementów biblioteki STL

Wykorzystane elementy biblioteki STL:

- queue, wykorzystaliśmy je w celu ustawienia klientów w kolejkę do maszyn i kas, analogicznie jak dzieje się w prawdziwym banku. Kiedy maszyna lub kasa się zwolni pierwsza osoba z kolejki podchodzi do wolnego stanowiska, a reszta klientów wciąż czeka w kolejności na swoją kolej.
- list
    - AllClients - przechowuje listę wszystkich klientów banku, dzięki użyciu listy możemy zapewnić że nie będzie dwóch takich samych klientów.
    - ListService - przechowuje wszystkie przeprowadzone transakcje przez klientów banku
- vector
    - AllWorkers - przechowuje vector wszystkich pracowników banku

6. Opis zidentyfikowanych sytuacji wyjątkowych i ich obsługi

Lista wyjątków

- ClientInDatabaseException - zapobiega dodania dwóch instancji tego samego klienta
- NegativeServiceTime - zapobiega negatywnemu czasu obsługi klienta
- NoAvailableWorkerException - jest wywoływane gdy wszyscy pracownicy są zajęci
- NoSuchWorkerException - nie został znaleziony pracownik o danych wartościach
- NoSuchClientInDatabaseException - nie został znaleziony klienta o danych wartościach
- WorkerAlreadyExistsException - stworzony pracownik już istnieje
- negative_account_balance - klienta ma negatywny bilans konta
- NegativeNumberOfClientsException - podana przez użytkownika liczba klientów jest mniejsza niż 0
- NegativeNumberOfWorkersException - podana przez użytkownika liczba pracownikówjest mniejsza niż 0
- NegativeTimeSimulationException - podany przez użytkownika czas symulacji jest mniejszy niż 0
- WrongNumberOfArgumentsException - liczba argumentów w pliku jest różna od 4
- FileDoNotExistException - plik z argumentami lub do przechowywania rezultatu symulacji nie istnieje

7. Opis podziału obowiązków w zespole.

Kacper Murygin:

- klasy Worker, Offer oraz Client (wraz z hierarchią- BusinessClient oraz PrivateClient - implementacja wraz z późniejszymi poprawkami
- klasy allClients, allWorkers- implementacja wraz z późniejszymi poprawkami
- klasa Simulation- implementacja klasy symulacji, dodanie setterów i getterów, konstruktora, rozpoczęcie implementacji metody start() klasy, odpowiedzialnej za przeprowadzenie symulacji, dodanie szkieletu obsługi kolejek, wybierania klienta,
- klasa Counter- implementacja jako klasa pochodna od Machine
- implementacja czytania argumentów z linii poleceń, wraz z enum-em dla lepszej czytelności,
- implementacja czytania argumentów z pliku tekstowego
- implementacja zapisu wyniku symulacji do pliku tekstowego
- dodanie wypisywania w symulacji wartości argumentów wywołania symulacji
- dodanie wypisywania w symulacji rozmiaru kolejek
- implementacja kolejek w symulacji, losowanie kolejki (typ usługi) dla klienta wraz z enum-em dla lepszej czytelności
- dodanie obsługi wyjątków w przypadku braku dostępnego pracownika
- dodanie zwalniania pracownika, jeśli usługa jest zakończona
- implementacja obsługi kilku kas
- poprawki kosmetyczne
- wyjątki- obsługa i implementacja
- dokumentacja
- testy jednostkowe

Maciej Bieńkowski

- Zaplanowanie wyglądu, hierarchii, działania projektu
- Koordynacja pracy, plany odnośnie rozwoju projektu
- klasy Service (wraz z hierarchią- Transfer, Loan, Payment, Deposit, ClosingAccount), Machine, Employee - implementacja wraz z późniejszymi poprawkami
- Klasa listService - implementacja wraz z późniejszymi poprawkami
- Ujednolicenie klasy Worker z klasą Employee
- Dodanie do klasy Client zmiennej Account,
- Pliki randomData, randomWorkers, randomClient - implementacja wraz z późniejszymi poprawkami
- implementacja obsługi klientów przez pracowników ludzkich jak i maszyny i wypisywania co się dzieje na ekranie
- implementacja odblokowywania stanowiska po odpowiednim czasie
- implementacja zapisywania wszystkich transakcji
- poprawki kosmetyczne
- dokumentacja