

Dokumentacja wstępna – Szpital

Oliwier Szypczyn, Andrzej Tokajuk

1. Temat projektu

Tematem projektu jest szpital. Jest on podzielony na oddziały, w których znajdują się sale. Pracują w nim pracownicy, którzy pełnią różne role m. in lekarze, pielęgniarki i ratownicy medyczni. W szpitalu hospitalizowani mogą być pacjenci z różnymi schorzeniami, którzy wymagają zróżnicowanego leczenia.

2. Klasy wykorzystane w projekcie:

- ☐ Symulacja
- ☐ Szpital
- ☐ Miejsce
 - ☐ Oddział
 - ☐ SOR/Recepcja
 - ☒ Dyspozytornia karetek
 - ☐ Sala
 - ☐ Ogólna
 - ☐ Gabinet
- ☒ Człowiek
 - ☒ Pacjent
 - ☒ Lekarz
 - ☒ Pielęgniarka
 - ☒ Ratownik medyczny
- ☒ Karetka
- ☒ Usługa-medyczna
 - ☒ konsultacja
 - ☒ operacja
- ☒ Baza danych-wszystkich usług medycznych
- ☒ Karta-zdrowia
- ☐ Kolejka
- ☒ Generator liczb losowych
- ☒ Generator losowych pacjentów

3. Opis klas:

→ Karta zdrowia:

◆ pola:

- pesel pacjenta - (identyfikator)
- choroby
- zaplanowane usługi medyczne

◆ metody:

- dodawanie i "uleczanie" chorób
- sprawdzanie czy jest dana choroba
- planowanie i kończenie usług
- sprawdzanie czy jest dana usługa

→ Usługa medyczna:

◆ pola:

- ID usługi
- całkowity planowany czas trwania
- aktualny czas wykonania
- lista lekarzy
- lista pielęgniarek
- pacjent
- stan usługi
- bool czy na NFZ?

◆ metody:

- gettery, settery
- operator inkrementacji czasu usługi
- dodanie pracownika
- liczenie kosztu
- sprawdzanie czy jest poprawny personel
- rozpoczęcie usługi
- kontynuacja usługi
- koniec usługi
- zwracanie pacjenta z usługi
- zwracanie personelu z usługi

→ Baza danych usług:

◆ pole:

- lista usług

◆ metody:

- planowanie usługi
- usuwanie usługi zrealizowanej
- wyszukiwanie usługi
- obliczanie kosztu wszystkich usług

→ Kolejka:

◆ pola:

- numerki
- pacjenci
- sala (identyfikator)

◆ metody:

- gettery, settery
- dodawanie pacjentów i usuwanie (queue)
- przydzielanie pacjentom numerków

→ Karetka:

◆ pola:

- ratownicy medyczni
- pacjent
- numer rejestracyjny
- całkowity planowany czas jazdy
- aktualny czas wykonania interwencji
- bool czy jest w akcji czy nie
- stan w którym się znajduje

◆ metody:

- gettery, settery
- jedź na interwencję
- kontynuuj interwencję
- wróć z interwencji
- dodaj personel
- zwróć personel
- zwróć pacjenta

→ Szpital:

◆ Pola

- Nazwa
- Lista "miejsc"
- Pacjenci
- personel
- usługi medyczne

◆ metody:

- manipulacja pacjentami
- manipulacja personelem
- manipulacja usługami

→ Człowiek:

◆ pola:

- PESEL - (identyfikator)
- Imię
- Nazwisko
- Płeć
- Wiek
- czy stan uległ zmianie (globalnie dla symulacji)
 - Pacjent
 - ◆ Karta zdrowia
 - ◆ Stan w którym aktualnie jest
 - Lekarz
 - ◆ Specjalizacja (enum)
 - ◆ stan w którym jest
 - ◆ stawka godzinowa
 - Pielęgniarka
 - ◆ stan w którym jest
 - ◆ stawka godzinowa
 - Ratownik Medyczny
 - ◆ stan w którym jest
 - ◆ stawka godzinowa

◆ metody:

- gettery i settery
- sprawdzenie czy stan uległ zmianie
- zmiana stanu
- operatory porównania

→ Miejsca:

◆ pola:

- Oddziały
 - Nazwa
 - Lekarze
 - Sale
 - ◆ Numer (identyfikator)
 - Ogólna:
 - Liczba łóżek
 - Pacjenci
 - Gabinet:
 - Pracownicy
 - Kolejka
 - pacjent
 - Rodzaj badania
- SOR/Recepcja
 - kolejka
 - pielęgniarki
- Dyspozytornia Karetok
 - Karetki

- Ratownicy medyczni

→ Symulacja:

- ◆ lista pacjentów szpitala
- ◆ data i czas
- ◆ metody
 - zarządzanie stanem ludzi ze względu na ich stan
 - czytanie statusu człowieka

→ Generator liczb losowych

- ◆ generator losowości na podstawie ziarna
 - Losowanie liczby z zakresu
 - Losowanie szansy zdarzenia na podstawie procentowej

→ Generator losowych pacjentów

- ◆ Generator losowych liczb
- ◆ Użyte w programie pesele
- ◆ Baza danych imion męskich
- ◆ Baza danych imion żeńskich
- ◆ Baza danych nazwisk
 - losowanie wszystkich pól i zwrócenie wskaźnika na pacjenta

[Link do murala z reprezentacją UML](#)

4. Opis założeń:

- szpital jest tworzony na podstawie pliku w formacie json i jego stan (liczba sal, personelu itd.) nie będzie zmieniany w trakcie działania programu
- w argumencie wywołania programu będzie podany okres działania symulacji (liczba iteracji), plik do odczytu szpitala i do zapisu przebiegu symulacji
- symulacja będzie iterować co 15 minut
- symulacja będzie losowała czy i ile w danym takcie ma przyjść do recepcji/przyjechać karetką pacjentów
- symulacja będzie posiadać dostęp do listy wszystkich pacjentów/pracowników i za każdym wykonaniem pętli będzie czytać odpowiednie pola i przemieszczać pacjentów/pracowników po szpitalu oraz generować opisy (tylko osoby których stan uległ zmianie)
- Symulacja i klasy w których jakaś czynność będzie korzystać z liczb losowych będzie korzystać z klasy -> generator liczb losowych

5. Podział obowiązków:

1. Oliwier

- Człowiek -> Pacjent/Lekarz/Pielęgniarka/Ratownik
- Dyspozytornia karetek
- Karetka
- Karta zdrowia
- Usługa medyczna -> Konsultacja/Operacja
- Generator liczb losowych
- Generator losowych pacjentów

2. Andrzej

- Sala
- Oddział
- Szpital
- Recepcja
- Kolejka

Symulacja i obsługa plików zostaje na koniec i jeszcze nie wiemy, kto konkretnie zajmie się czym aby całość mogła działać sprawnie. Najpierw chcemy skupić się na fizycznych elementach szpitala

6. Realizacja symulacji:

1. Pobranie za pomocą klasy do plików szpitala i jego statycznych pól z pliku json:

- Nazwa
- Personel medyczny - dodany do bazy danych ludzi
- Miejsca
- Bez pacjentów
- Bez usług

2. Zaczynamy symulację:

- Losujemy czy w danej iteracji ma przyjść do recepcji pacjent: 75%
 - Pacjent jeśli przyjdzie do recepcji to:
 - Ma losowane z pewnej puli dane (klasa pomocnicza)
 - karta zdrowia jest generowana z jakąś chorobą i bez zaplanowanych usług
 - Umieszczamy go w kolejce do recepcji
 - Generujemy wiadomość o tym że pacjent przyszedł do szpitala
- Losujemy czy w danej iteracji ma przyjechać karetką pacjent: 25%
 - Wysyłamy karetkę po pacjenta
 - Generujemy pacjenta tak jak wyżej
 - Generujemy, że karetka wyjechała na interwencję
- Przechodzimy po karetkach i szukamy zakończonych kursów
 - Jeśli jakiś się zakończył to przenosimy pacjenta do kolejki na recepcję ale na początek
 - Generujemy wiadomość, że karetka zakończyła kurs
- Przechodzimy po pacjentach z sali ogólnej:
 - Jeśli pacjent nie ma chorób to może wyjść ze szpitala

- Generujemy wiadomość, że pacjent opuścił szpital
 - Jeśli pacjent ma zaplanowaną operację to wysyłamy go do gabinetu do kolejki do operacji
 - Jeśli pacjent ma zaplanowaną konsultację to wysyłamy go do gabinetu do kolejki do konsultacji
- Przechodzimy po kolejce do gabinetów:
 - Gabinet operacyjny:
 - Jeśli w gabinecie nie odbywa się operacja to bierzemy pierwszą osobę z kolejki i dodajemy do operacji (rozpoczynamy operację)
 - a. Wiadomość o rozpoczęciu operacji
 - Jeśli w gabinecie jest już operacja to kontynuujemy operację
 - Jeśli operacja się kończy to z szansą 60% leczymy chorobę
 - Jeśli choroba wyleczona to usuwamy mu zaplanowaną operację
 - Wysyłamy do sali ogólnej
 - a. Wiadomość o zakończonej operacji z informacją o powodzeniu
 - Gabinet konsultacyjny:
 - Jeśli w gabinecie nie odbywa się konsultacja to bierzemy pierwszą osobę z kolejki i dodajemy do konsultacji (rozpoczynamy konsultację)
 - a. Generujemy wiadomość o rozpoczęciu konsultacji
 - Jeśli w gabinecie jest już konsultacja to ją kontynuujemy
 - Jeśli konsultacja się skończy to z szansą 30% dodajemy nową chorobę
 - Do wszystkich chorób pacjenta planujemy mu operację
 - Wysyłamy do ogólnej
 - a. Generujemy wiadomość, że konsultacja się zakończyła i o tym czy zostały wykryte choroby
- Przechodzimy po kolejce do recepcji:
 - Bierzemy pierwszą osobę z kolejki
 - Planujemy mu z szansą 50/50 konsultację lub operację
 - Wysyłamy do sali ogólnej
 - Generujemy wiadomość, że wysyłamy pacjenta gdzieś (wynik wizyty w recepcji)
- Na koniec każdej iteracji wyświetlamy i zapisujemy do pliku wszystkie wiadomości
- Czyścimy wektor wiadomości i strumień
- Inkrementujemy czas
- Czekamy na pozwolenie od użytkownika, że można zacząć kolejną iterację