AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

Biblioteka obsługująca system wind

z przedmiotu

Języki programowania obiektowego

Elektronika i telekomunikacja 3 rok

Bartłomiej Kisielewski

piątek 15:00

prowadzący: mgr. Inż. Jakub Zimnol

9.01.2025

1. Opis projektu

Biblioteka zarządza systemem wind, umożliwiając symulację działania zespołu wind w budynku wielopiętrowym. Projekt zawiera również plik testowy do demonstracji funkcjonalności systemu. Główne funkcjonalności to:

- Obsługa wezwań z korytarzy na piętrach
- Obsługa wyboru pięter wewnątrz windy
- Efektywne przypisywanie zadań do wind w systemie

2. Opis klas

Klasa Elevator

Reprezentuje pojedynczą windę w systemie. Oferuje następujące funkcjonalności:

- Ruch windy: moveUp, moveDown, moveToFloor
- Obsługa wezwań na piętra: addTargetFloor
- Dostęp do informacji o stanie windy:
    - Aktualne piętro
    - Kierunek ruchu (up, down, idle)
    - Bufor pięter docelowych

Posiada pola prywatne takie jak:

- Aktualne piętro, maksymalne i minimalne piętro
- Aktualny kierunek ruchu windy
- Bufor pięter docelowych

Jeśli chodzi o istotne metody prywatne to mamy:

- moveUp oraz moveDown odpowiedzialne za ruch windy o jedno piętro w górę / dół

Klasa nie posiada publicznych pól, natomiast posiada metody publiczne, najistotniejsze to:

- Konstruktor Elevator, inicjalizuje nam obiekt windy, ustawiając początkowe piętro, maksymalne jak i minimalne piętro
- Odpowiednie gettery
- moveToFloor – łącząca ze sobą metody prywatne, by przesunąć windę na docelowe piętro
- addTargetFloor, która dodaje piętro do kolejki zadań windy

Klasa System

Zarządza systemem wind i ich współpracą, a oferuje ona:

- Przypisywanie wezwań z korytarzy do najlepszej windy z wykorzystaniem algorytmu najbliższego wezwania.
- Wyświetlanie statusu wszystkich wind w systemie (elevatorStatus).
- Wykonywanie kroków symulacyjnych dla każdej windy (performNextStep).

Pola prywatne:

- elevators to wektor obiektów typu Elevator, reprezentujących windy.
- maxFloors z maksymalną liczbą pięter w systemie

Metoda prywatna:

- findBestElevator wybiera najbardziej dogodną windę do obsługi wezwania

Metody publiczne:

- Konstruktor System tworzy system wind, inicjalizując liczbę wind wraz z max liczbą pięter, zawiera również walidację, by nie było np. sytuacji że system zawiera 0 wind
- Getter potrzebny do pobrania ID danej windy
- elevatorStatus do monitorowania stanu naszego systemu
- handleHallCall przypisuje wezwanie z korytarza do wcześniej wybranej najlepszej windy

Program główny (test.cpp)

Symuluje działanie systemu i umożliwia interakcję użytkownika:

- Wywołanie windy na piętro z korytarza
- Wybór piętra wewnątrz kabiny windy
- Przegląd aktualnego stanu wind

3. Kompilacja i uruchomienie

Korzystamy z Cmake, a jednocześnie mamy w repozytorium .gitignore, który zapobiegnie dodania do repozytorium po zmianach (build będzie jedynie lokalnie). W ogólnym przypadku komenda nie posiadałaby „-G „MinGW Makefiles”, moim przypadku wykorzystuje system windows z MinGW, dlatego w komendzie się znajduje MinGW Makefiles.

- cmake -G "MinGW Makefiles" -B build
- cmake --build build
- ./build/ElevatorSystem

Po uruchomieniu programu, pozwalamy użytkownikowi wybierać opcje z menu, np.:

- Przyzwanie windy na piętro
- Dodawanie pięter do listy celów w kabinie windy
- Przegląd stanu wind w systemie

4. Dodatkowe informacje

Screeny prezentujące działanie

Nieprawidłowe dane – walidacja wpisanych danych przez użytkownika - error

```
PS C:\Users\pc\Desktop\JPO2\System-for-Elevator-service> cmake -G "MinGW Makefiles" -B build
-- The CXX compiler identification is GNU 13.1.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: C:/msys64/ucrt64/bin/cc.exe - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:/msys64/ucrt64/bin/c++.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done (1.8s)
-- Generating done (0.0s)
-- Build files have been written to: C:/Users/pc/Desktop/JPO2/System-for-Elevator-service/build
PS C:\Users\pc\Desktop\JPO2\System-for-Elevator-service> cmake --build build
[ 25%] Building CXX object CMakeFiles/ElevatorSystem.dir/src/elevator.cpp.obj
[ 50%] Building CXX object CMakeFiles/ElevatorSystem.dir/src/system.cpp.obj
[ 75%] Building CXX object CMakeFiles/ElevatorSystem.dir/src/test.cpp.obj
[100%] Linking CXX executable ElevatorSystem.exe
[100%] Built target ElevatorSystem
PS C:\Users\pc\Desktop\JPO2\System-for-Elevator-service> .\build\ElevatorSystem.exe

Set up program:

Enter the number of floors in the building: -2

Enter the number of elevators in the building: 3
Error: Number of elevators must be greater than 0 and number of floors must be at least 1
terminate called after throwing an instance of 'std::invalid_argument'
  what():  Number of elevators must be greater than 0 and number of floors must be at least 1
PS C:\Users\pc\Desktop\JPO2\System-for-Elevator-service>
```

Poprawne stworzenie systemu, wyświetlenie statusu wind oraz menu dla użytkownika

```
PS C:\Users\pc\Desktop\JPO2\System-for-Elevator-service> .\build\ElevatorSystem.exe

Set up program:

Enter the number of floors in the building: 10

Enter the number of elevators in the building: 3

Simulation started with 3 elevators.

========================================================================================== [counter: 0]
Elevator 0:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:
Elevator 1:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:
Elevator 2:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:

========================================================================================== [counter: 1]

Choose an option:
1. Call elevator
2. Pick floor in the elevator
3. Perform next step (movement)
4. Exit
Enter your choice:
```

Opcja 1 – przyzwanie windy z piętra

```
Choose an option:
1. Call elevator
2. Pick floor in the elevator
3. Perform next step (movement)
4. Exit
Enter your choice: 1
Enter the floor to call from (0-10): 7
Enter the direction (1 for up, 2 for down): 1
Added target floor: 7
Call assigned to elevator 0.

========================================================================================== [counter: 2]
Elevator 0:
        Current floor: 0 || Direction: Idle || Next Target: 7 || Floors in queue: 7
Elevator 1:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:
Elevator 2:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:

========================================================================================== [counter: 3]
```

```
Choose an option:
1. Call elevator
2. Pick floor in the elevator
3. Perform next step (movement)
4. Exit
Enter your choice: 1
Enter the floor to call from (0-10): 2
Enter the direction (1 for up, 2 for down): 2
Added target floor: 2
Call assigned to elevator 1.

========================================================================================== [counter: 12]
Elevator 0:
        Current floor: 4 || Direction: Up || Next Target: 7 || Floors in queue: 7
Elevator 1:
        Current floor: 0 || Direction: Idle || Next Target: 2 || Floors in queue: 2
Elevator 2:
        Current floor: 0 || Direction: Idle || Next Target: 0 || Floors in queue:

========================================================================================== [counter: 13]
```

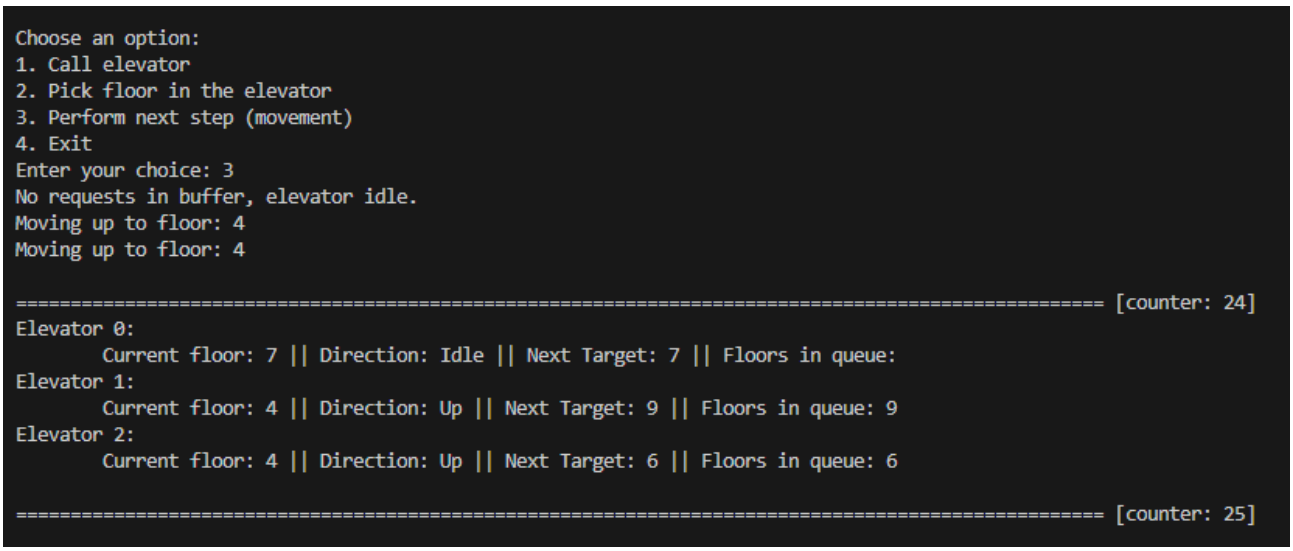
Opcja 2 – wybór piętra w windzie

```
Enter your choice: 2
Pick elevator ID (0-2): 1
Pick target floor (0-10): 9
Added target floor: 9
Floor 9 added to queue for elevator 1

========================================================================================== [counter: 16]
Elevator 0:
        Current floor: 4 || Direction: Up || Next Target: 7 || Floors in queue: 7
Elevator 1:
        Current floor: 0 || Direction: Idle || Next Target: 2 || Floors in queue: 2 9
Elevator 2:
        Current floor: 0 || Direction: Idle || Next Target: 6 || Floors in queue: 6

========================================================================================== [counter: 17]
```

Opcja 3 – krokowanie



Opcja 4 – zakończenie symulacji