

Specifikation P-uppgift: Platsbokning på SJ

Beskrivning

- Program som hanterar platsbokning för tåg.
- Användaren ska kunna boka, avboka och skriva ut biljetter. Om användaren väljer att boka får den först välja varifrån den vill åka, sen vart den vill åka och sedan vilken tid.
- Användaren kan boka flera platser samtidigt och välja om de ska vara bredvid varandra eller sprida.
- Data för varje tåg sparas i en egen fil som uppdateras varje gång en plats bokas på det tåget. Tågfilerna läses in vid programmets start.

Algoritm

1. Programmet läser in alla tågfiler.
2. Följande meny visas tills användaren avslutar programmet:
Vad vill du göra?
 - Boka, skriv 'B', på samma rad följt av önskat antal biljetter.
 - Avboka, skriv 'A', på samma rad följt av ett platsnummer.
 - Skriva ut de senaste bokade biljetterna, skriv 'S'.
 - Avsluta, skriv 'Q'
3. Beroende på vilket val användaren gör anropas olika funktioner med olika parametrar. När användaren väljer skriva ut ska en biljett skapas som en ny fil. När användaren bokar platser ska detta sparas till tågfilen.
4. Vid nästa bokning läses filen in igen för att inte riskera dubbelbokning
5. När tiden för en avgång har passerat ska den avgången och alla biljettfiler för den tas bort.

Datastrukturer

Varje plats representeras av en instans av klassen Seat. Varje vagn representeras av ett Carriage-objekt som har en dictionary med alla dess platser. Vagnarna finns i sin tur i en dictionary i varje tåg som representeras av ett Train-objekt. Alla Train-objekt lagras i en dictionary med tåg ID som nyckel för att enkelt kunna få ut data från det. All data om tågen lagras i Json-filer.

Klasser och metoder

```
class Seat:
    """
    Attributes:
    number: seat number
    is_booked: True if booked, else False
    """

    def __init__(self, number, is_booked):
        """
        Creates a new Seat
        :param number: The number of the seat
        :param is_booked: True if booked, else False
        """

    def __str__(self):
```

Mikaela Gärde
migarde@kth.se

```
    """
    Used to get information about the seat to print
    :return: String with seat number
    """

    def book(self):
        """
        Used to mark the seat as booked
        :return: String confirming the booking
        """

    def unbook(self):
        """
        Used to mark the seat as unbooked
        :return: String confirming the unbooking
        """

    def make_dict(self):
        """
        Used to make the information about the seat into a dictionary to
        save to json file
        :return: Dictionary of information about seat
        """

class Carriage:
    """
    Attributes:
    id: Id of the carriage
    seats: Dictionary of all seats in the carriage
    """

    def __init__(self, id, seats):
        """
        Creates a new carriage
        :param id: Id of the carriage
        :param seats: List of all seats in the carriage
        """

    def __str__(self):
        """
        Used to get information about the carriage to print
        :return: String with information about the carriage
        """

    def show(self):
        """
        Used to get a overview of the carriage with all booked seats marked
        to print
        :return: String creating a overview of the carriage
        """

    def booked_seats(self):
        """
        Used to display the number of booked seats in the carriage
        :return: String with number of booked seats
        """

    def make_dict(self):
        """
        Used to make the information about the carriage into a dictionary

```

Mikaela Gärde
migarde@kth.se

```
to save to json file
:return: Dictionary of information about the carriage
"""
```

```
class Train:
    """
    Attributes:
    id: Id of the train
    carriages: Dictionary of all carriages on the train
    time: Time and date when the train departs
    departure: Name of the departure
    Destination: Name of the destination
    """

    def __init__(self, id, carriages, time, departure, destination):
        """
        Creates a new train
        :param id: Id of the train
        :param carriages: List of all the carriages the train consists of
        :param time: Time and date the train departs
        :param departure: Name of the departure
        :param destination: Name of the destination
        """

    def __str__(self):
        """
        Used the get information about the train to print
        :return: String with information about the train
        """

    def get_time(self):
        """
        Used to rewrite the time attribute to a printable string
        :return: String with time and date
        """

    def make_dict(self):
        """
        Makes the information about the train into a dictionary to save to
        json file
        :return: Dictionary of information about the train
        """
```

Funktioner

```
def read_files():
    """
    Used to read all the json-files in the directory "trains" and creates
    Seat -, Carriage - and Train Objects from the data in the files.
    :return: (nothing)
    """

def write_file(train):
    """
    Used to save the data from a train to a json file in the directory
    "trains".
    :return: (nothing)
    """

def show_menu():
```

Mikaela Gärde
migarde@kth.se

```
    """
    Used to print the menu
    :return: (nothing)
    """

def ask_for_choice():
    """
    Used to ask the user for a choice
    :return: Choice as a String
    """

def menu_execute(choice):
    """
    Used to execute the option the user chooses in the menu
    :param choice: a string with the chosen option
    :return: (nothing)
    """

def get_departures():
    """
    Used to get list of all departures from all trains to print
    :return: List of all departures
    """

def get_destinations(departure):
    """
    Used to get list of all destinations from the selected departure
    :param departure: Selected departure
    :return: List of destinations
    """

def get_trains(departure, destination):
    """
    Used to get information about all trains connected to the selected
    departure and destination
    :param departure: Selected departure
    :param destination: Selected destination
    :return:
    """

def book():
    """
    Used to book seats
    :return: String confirming the booking
    """

def unbook():
    """
    Used to unbook a seat
    :return: String confirming the unbooking
    """

def print_ticket():
    """
    Used to print tickets from the booked seats
    :return: String confirming the printing
    """
```