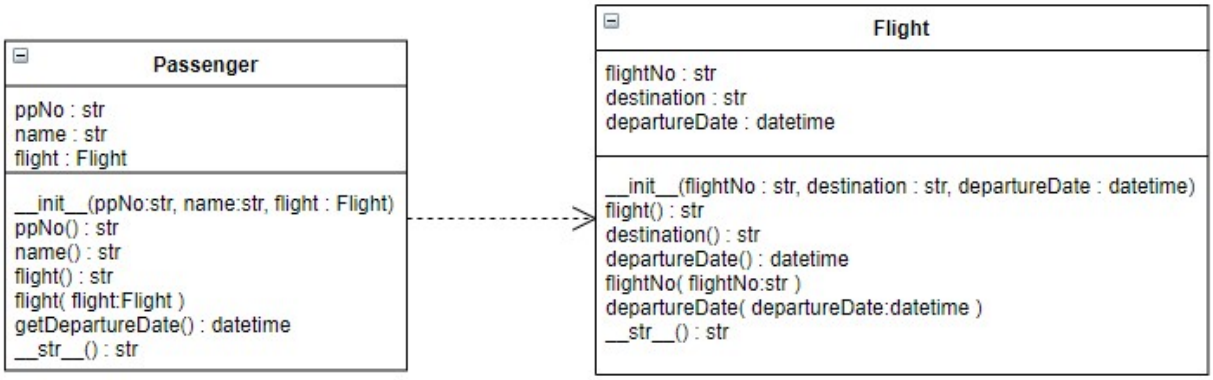
Lab 2 (Composition and Collection)

1. This question models a Passenger taking a Flight. The class diagrams for a Passenger and Flight are as follows:



The code for the Flight class is given as follows:

```
from datetime import datetime
class Flight:
    def __init__(self, flightNo,  destination,  departureDate):
      self._flightNo = flightNo
      self._destination =destination
      self._departureDate = departureDate

    @property
    def flightNo(self):
        return self._flightNo

    @property
    def destination(self):
        return self._destination

    @property
    def departureDate (self):
        return self._departureDate

    @flightNo.setter
    def flightNo(self, flightNo):
        self._flightNo = flightNo

    @departureDate.setter
    def departureDate (self, departureDate):
        self._departureDate = departureDate

    def __str__(self):
        return f'Flight: {self._flightNo} Destination: {self._destination} Departure
Date: {self._departureDate:%d/%m/%Y %H:%M }'
```

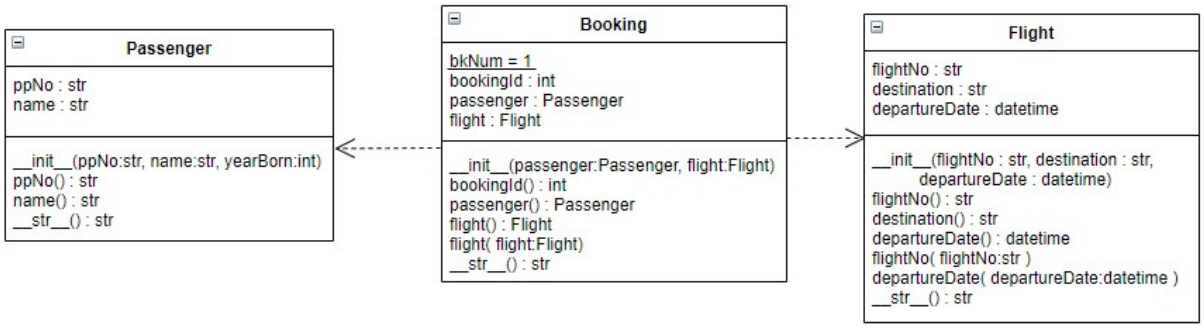
Use composition to write a Passenger class. It has 3 instance variables – ppNo (str), name(str) and flight(Flight).

The constructor has passport no, name and flight as parameters. It has a getter property for ppNo, name, flight and a setter property for flight to assign a different flight. A method getDepartureDate() returns the departureDate of the flight. The __str__() method returns the passenger name and flight details as follows:

```
Name: John   Flight: SQ1   Destination: LA   Departure Date: 25/12/2021 04:15
```

a. Write the Passenger class.
b. Write an application to test the classes above by doing the following:
   i. Create a Flight object f1 – SQ1 to LA on 25/12/2021 0415
   ii. Create a Passenger object p1 John, pp1 taking flight f1
   iii. Print the departure date for p1
   iv. Create a Passenger object p2 also taking flight f1
   v. Print both passengers' information using the str method
   vi. Change the flight object's departure date to 26/12/2021 1525
   vii. Print both passengers' information using the str method.

2. a. Discuss the limitations with the design of the Passenger and Flight class in Q1.
   b. Given the following class diagram.

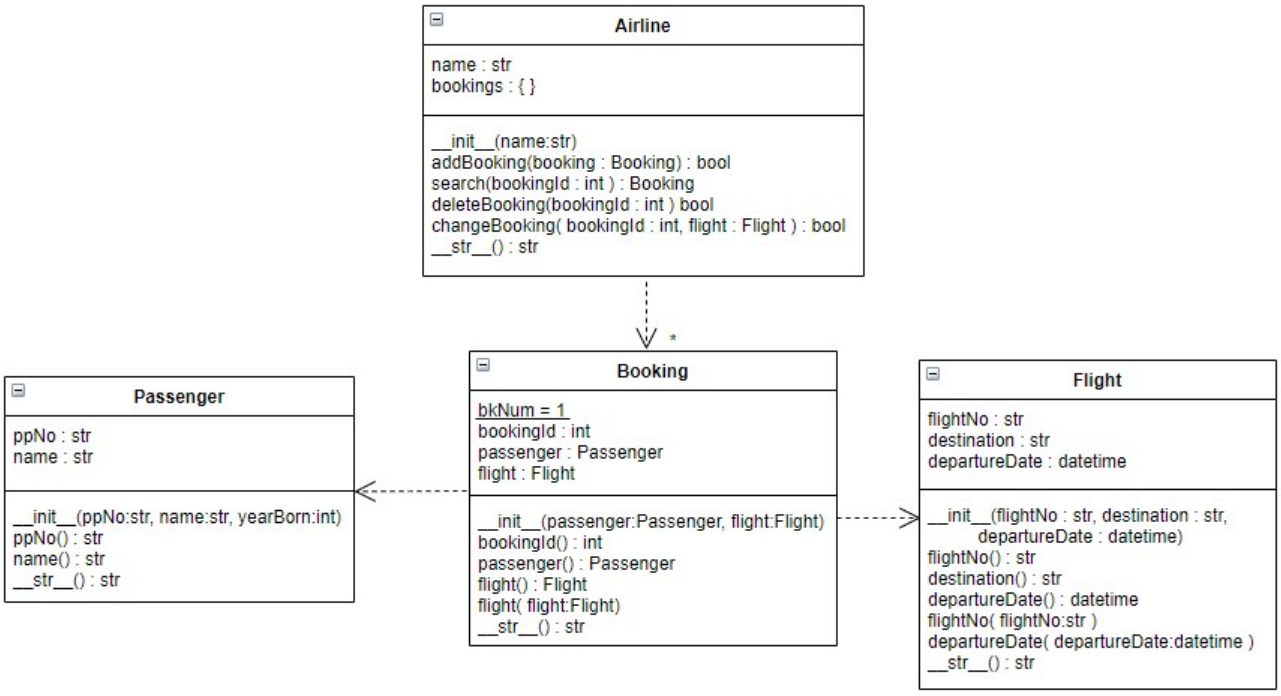


A Booking class is introduced to store Passenger and Flight booking. The class has 3 instance variables, the booking id which starts from the class variable bkNum=1 and increments by 1 for every new booking, a passenger and flight. It has properties for booking id, passenger and flight, and flight setter to allow change of flight. The str() method displays the booking details as follows:

```
Booking id: 1
Passport No: PP1   Name: John
Flight: SQ1   Destination: LA   Departure Date: 25/12/2021 04:15
```

Write the Booking class.
Test the class with the following statements:
i. Create an empty list to store booking objects.
ii. Create 2 Passenger objects p1 and p2.
iii. Create 2 Flight objects f1 and f2.
iv. Create 2 booking objects for the passengers and flights created and add to the list, p1 taking f1 and p2 taking f2.
v. Display the details of the bookings from the list.
vi. Make changes such that both passengers are taking the same f1 flight.
vii. Display the details of the booking.
viii. Change the flight departure date for the flight that both passengers are taking.
ix. Display the details of the booking again.

3. Given the following class diagram:



The bookings are managed by an Airline class which consists of 2 instance variables, the airline name and a dictionary collection of bookings. The class has the following methods:

- addBooking(booking) The method has a Booking object as parameter and adds the booking to the dictionary. The key is the booking id and the value the Booking object. The method returns True if the Booking object is added successfully and False otherwise.
- Search(bookingId) Given a booking id, the method returns the Booking object if found, and None otherwise.
- deleteBooking(bookingId) Given a booking id, the method removes the Booking from the dictionary if found. Return True if the remove is successful and False otherwise.
- changeBooking(bookingId, flight) Given a booking id and a flight, the method replaces the booking with another flight. Return True if the change is successful and False otherwise.
- __str__() returns a string of all the bookings in the following format:

```
Booking id: 1
Passport No: PP1   Name: John
Flight: SQ1   Destination: LA   Departure Date: 25/12/2021 04:15

Booking id: 2
Passport No: PP2   Name: Peter
Flight: SQ2   Destination: New York   Departure Date: 1/12/2021 01:10
```

Test the Airline class with the following statements:

a. Create an Airline object.
b. Add several bookings. To do this, create 3 Passenger objects, 3 Flight objects, 3 Booking objects and add the bookings. For each booking, print the booking id.
c. Search for a booking based on the booking id and display the details if found.
d. Change one of the bookings to another flight.
e. Remove one booking.
f. Print all the bookings.