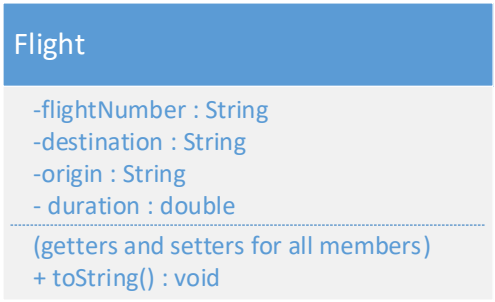
1. Flight control



a. Create the Flight class. Ensure that you add a default and a constructor with all arguments.

b. Then create the following flight objects in a class (with a main method) called **AirTrafficControl**.

Flight number	Destination	Origin	Duration
ESY3214	Belfast	Amsterdam	2.2
BA127	London Heathrow	Cape Town	11.5
AE999	Dublin	New York JFK	7.5
AA678	Dublin	Newwark	8.5

For example :
Flight f1 = new Flight("ESY3214", "Amsterdam", "Belfast", 2.2);

c. Output all details for all flights i.e. use the **toString()**

d. It is possible to hold all the flights in an array…

Flight[] flights = {f1, f2,f3,f4};

Add all the flights to an array.

e. Referring to an element in the array of flights will enable you to access an individual flight and the methods within.

e.g. **System.out.println(flights[0].getFlightNumber());**

Use a FOR loop to print output all the details of all the flights in the array of flights.

```
System.out.println("All flights.................");
// now show all flights using the array
for (int loop = 0; loop < flights.length; loop++) {
    System.out.println(flights[loop].toString());
}
```

f. Use the array to search for the flight with the longest duration. (Output the longest duration)

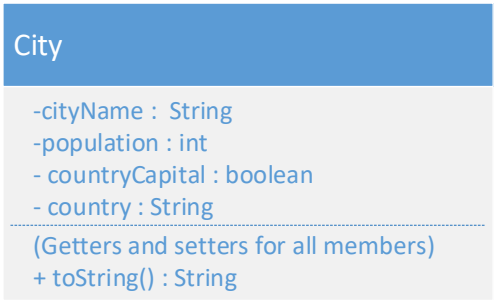
e. Use the array to search for all the flights to Dublin. (Output the flight numbers)

e. Update the Flight class. Add validation to the class to ensure that the ***Duration*** upper limit is 18 hours (inclusive). If this happens then default ***Duration*** to 0 and output an appropriate message to the user. **Hint** : put the validation in the setter for the ***duration*** and call it directly from the constructor.

Test this new feature with the following …

Flight number	Destination	Origin	Duration
AUS12	Beijing	Paris	18.1

2. **City Listings**



a. Create the City class. Ensure that you add a default and a constructor with all arguments. Add validation to ensure that the population cannot be set to less than 0 and more than 50 (default to 1).

b. Then create the following City objects in a class (with a main method) called **CityAdmin**.

cityName	Population (million)	countryCapital	country
Beijing	11	true	China
Madrid	5	true	Spain
Rome	6	true	Italy
New York	8	false	USA
Washington	2	true	USA
Dallas	3	False	USA

c. Store all the Cities in an array

Instead of coding all the following functions in the **main** method it would be better to use a method for each one. Create each method (it may be static) and then invoke each one in turn from the main method (passing the array as a parameter argument each time).

```
public static void main(String[] args) {

    // create the cities
    City c1 = new City("Beijing", 11, true, "China");
    City c2 = new City("Madrid", 5, true, "Spain");
    City c3 = new City("Rome", 6, true, "Italy");
    City c4 = new City("New York", 8, false, "USA");
    City c5 = new City("Washington", 2, true, "USA");
    City c6 = new City("Dallas", 3, false, "USA");

    // add to an array
    City[] cities = { c1, c2, c3, c4, c5, c6 };

    // show all
    displayAll(cities);

    // searching for the smallest city
    smallestCity(cities);

    // capital cities search
    capitalCitySearch(cities);

    // city by country search
    cityByCountry(cities, "USA");


    // cities population average
    averagePopulation(cities);
}

/**
 * Prints all city details to screen
 * @param cities
 */
public static void displayAll(City[] cities) {
    System.out.println("Displaying all cities ");
    for (City city : cities) {
        System.out.println(city.toString());
    }
    System.out.println();
}
```

d. Use an enhanced FOR loop to print out all the details of all the Cities in the array.

e. Use the array to search for the City with the smallest population. (Output the population)

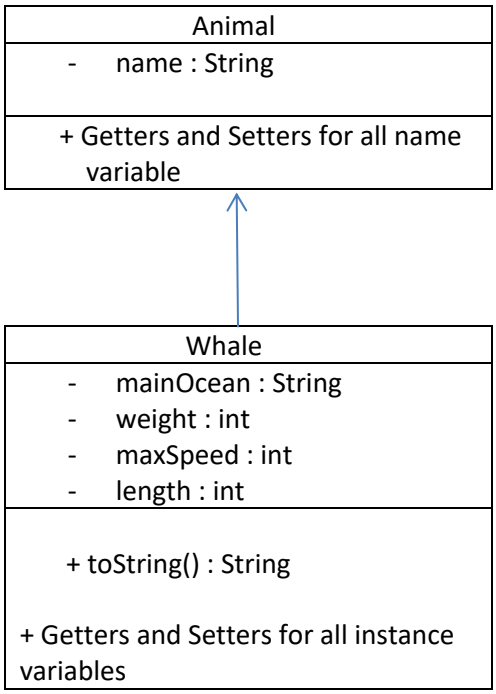
e. Use the array to search for all the Cities that are capitals of their country (Output the names of the cities and the respective country)

f. Use the array to search for all the Cities that are from the USA (Output the names of the cities)

g. Output the average population of all the cities in the array (Output to 2 decimal places).

3. Whales

a. Create a Java program based on the following UML classes. (Ensure default and constructors with full arguments are included in both classes)



B. Create a *WhaleWatcher* class that will have a *main* method to create the following instances of Whales.

Name	**Right**	**Blue**	**Sperm**	**Humpback**
Main ocean	Atlantic	Pacific	Atlantic	Antarctic
Weight (tonnes)	2001	2001	1900	919
Max Speed	21	23	20	13
Length (meters)	16	16	40	13

Again, a lot of functionality required for this class, so use a method for each task…..

- Use an array to store the four Whale objects.
- Use the array to output all the details for each Whale (use the *toString()* method).
- Use the array to output the names of the whales that swim mostly in the `Atlantic`
- Use an array to output the name of the whale(s) that swim fastest
- Output average length of the all whales
- Use an array to output the name of the heaviest whale(s)
- Add appropriate validation