3. A transport company has a number of vehicles which can carry passengers. Each vehicle is classified either as a bus or as a coach. All vehicles have a registration number and have a certain number of seats for the passengers. A bus can have a maximum number of standing passengers, but a coach is not allowed to carry any standing passengers. Some of the coaches are fitted with seat belts, but seat belts are never fitted in a bus.

The transport company currently has a data file, `VEHICLE.dat`, stores the following data:

- `RegNo` is used to uniquely identify a particular vehicle. A typical vehicle registration number comes in the format "SBA1234":
  - S – vehicle class ("S" stands for a private vehicle)
  - BA – alphabetical series ("I" and "O" are not used to avoid confusion with "1" and "0")
  - 1234 – numerical series
- `NoOfSeats` is the maximum number of seats for passengers that each vehicle can carry.
- `VehicleType` is the type of the vehicle and can take one of the two values: '`B`' for a bus and '`C`' for a coach.

`VEHICLE.dat` has the following structure:

<NumberOfRecords>
<RegNo>|<NoOfSeats>|<VehicleType>
<RegNo>|<NoOfSeats>|<VehicleType>
. . . .
. . . .
<RegNo>|<NoOfSeats>|<VehicleType>

NumberOfRecords is the number of records in the file.

Task 3.1
Complete the test case table with the addition of **three** more invalid vehicle registration numbers. The reasons for their invalidity should be different.

The return value is a code as follows:

- 0 – valid registration number
- 1 – the registration number was not 7 characters
- you will use other integer numbers for other invalid cases.

Test Number	`RegNo`	Return value	Explanation of the test case
1	`SBA1234`	`0`	Valid registration number
2			
3			
4			

Evidence 10:
The completed test case table. [3]

Task 3.2
Write program code for a function to validate a registration number. The function header has the format:

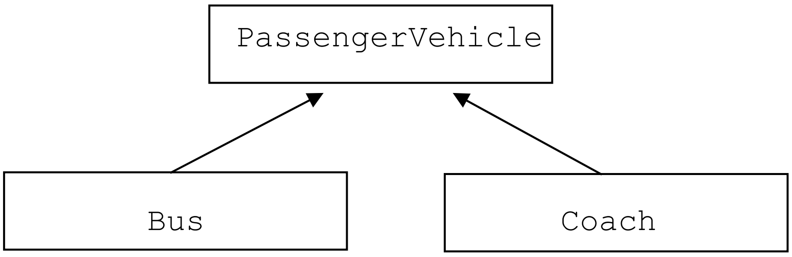
```
FUNCTION ValidateRegNo(ThisRegNo : STRING) RETURNS INTEGER
```

Write a program to:
- Input a registration number by the user
- Validate the input using the function `ValidateRegNo`
- Output a message describing the validity of the input.

Evidence 11:
- Program code for the function `ValidateRegNo` [4]
- **Three** screenshots showing the testing of Test Numbers 2, 3 and 4.. [3]

Additional data now needs to be stored about the vehicle:
- `MaxStanding` is the maximum number of standing passengers that a vehicle classified as of type '`B`' can carry.
- `SeatBeltsFitted` is a field that indicates whether a vehicle of type '`C`' has fitted with seat belts.

The program design to process data about the vehicles is to be implemented with object-oriented programming with the following three classes:



Task 3.3
Write program code for the three classes shown.

Evidence 12:
Program code for the three classes. [6]

Task 3.4
The data in VEHICLE.dat does not currently contain the following additional data:

- MaxStanding
- SeatBeltsFitted

Write code to read a record from VEHICLE.dat and write the updated record to UVEHICLE.dat.

As the data on each vehicle is read it should be displayed and the user should be allowed to input the additional data required.
The user should be prompted to input the data item appropriate to the type of vehicle.

The number of standing passengers is never more than 15.
For a 'C' type vehicle the MaxStanding field should contain a zero value.
The SeatBeltsFitted field should contain an appropriate value.

You are expected to make use of the classes you designed in Task 3.3.

Evidence 13:
Program code for Task 3.4. [10]

Evidence 14:
Screenshot showing the contents of UVEHICLE.dat from running the program. [2]