C++ task 1:

- Source code (.cpp or .txt)
- Please **do not** distribute the source code to multiple files!
- **Source code, which has been used from a web source, has to be mentioned as a comment!**

Initial situation:

You want to create a program for the Federal Motor Transport Authority to store the data of vehicles.

Write a C++ program based on object-oriented principles that covers at least the following situation.

The following data of each vehicle is stored in the system:

- Number plate
- Year of first registration
- Year of general inspection

The current year is 2022. If the difference between the current year and the year of the general inspection is greater than 3, then the general inspection is due.

In addition, there should be the following three subclasses, which inherit the data members from the vehicle class. The following derived classes have additional information:

Car:

- Horse power
- Emission class [0-low-emission, 1-normal, 2-diesel]

Motorbike:

- Horse Power
- Maximum speed

Truck:

- Number of axles
- Payload in ton

Each derived class should contain be a member function *getTaxLiability*, which returns the following:

Calculation of the tax liability according to the following formulas:

  - Car: (Horse Power + 99) / 100 * 10 EURO * (emission class + 1)
  - Motorbike: (Horse Power + 59) / 100 * 20 EURO * (maximum speed / 150)
  - Truck: payload * 50 EURO per ton

When the program starts, the "database" of the vehicle management system is empty, the data must first be entered.

The data must only be available during runtime (no permanent storage necessary!).

Task:

Vehicle management for a maximum of 100 vehicles (can be either a Car, Motorbike or Truck)

Create a program with a menu that covers the following situation:

- Creating a new vehicle

- Search for a vehicle based on the entered license plate number and return all relevant vehicle data (including the liability and the information of the derived class)

- Listing of all stored vehicle data and the liability

- Listing of all vehicles, which general inspection is due

- Program exit

If necessary (creation, search, and so on), corresponding messages are to be displayed.
Permanent storage of the data is not required. The data must only be available at the runtime of the program.

Use comments in the source code to document your program.

C++ task 2

- Source code (.cpp or .txt)
- Please **do not** distribute the source code to multiple files!
- **Source code, which has been used from a web source, has to be mentioned as a comment!**

Initial situation:

The company TravelSea is planning to introduce a new booking and information system for its ferries.
The following aspects must be taken into account here:

- Since the company is expanding rapidly, the number of ships for which the booking system is to be designed must be kept variable.
- Each ship has a unique name, a ticket price and a loading capacity design that is independent of the other ships. The loading capacity may not be exceeded, meaning there can't be more people on the ship then the maximum capacity.
- When occupying a ship with passengers, a distinction should be made between the passengers. The following passenger data is required for the booking system:
  - First name
  - Last name
  - Year of birth
  - Student (yes/no) -> if yes, then the matriculation number should be stored. A student can only be within the age of 18 and 64.
- The calculation of the ferry tarif per person should also be based on different criteria:
  - Passengers between the age 18 and 64 pay the regular ticket price
  - Passengers of the age less than 13 are free of charge
  - Passengers of the age greater than 64 receive a discount of 20%
  - Students and passengers between the age 14 and 17 receive a discount of 25%

Task:

Create a program with a menu that covers the following situation:

• Create a new ship - the recording of the data has to be done via istream (>>) operator overloading and the new ship will be stored in a linked list

• Output of all ships - the output of the data has to be done via ostream (<<) operator overloading: e.g.
- Arielle, 50 Euro p. P., 100 out of 1200 passengers on board, total sales: 12000 Euro
- Black Thunder, 75 Euro p. P., 450 out of 700 passengers on board, total sales: 45500 Euro
- Little Nemo, 20 Euro p. P., 10 out of 75 passengers on board, total sales: 625.75 Euro

• Manage ship - the user can search for a specific ship. If a ship has been found, the user has two options:

- *Add passenger:* Create a new passenger, which is then on board of the ship.
- *Manage ticket price:* The ticket price can be increased or decreased. The price can't be less than 0 Euro. The increase and decrease should be done by overloading the operator -- and ++, which increases/reduces the ticket price by 1 Euro.

• Program exit

Create the program dynamically using pointers. If necessary (modification, creation, and so on), corresponding messages are to be displayed.
Permanent storage of the data is not required. The data must only be available at the runtime of the program.

Use comments in the source code to document your program.