Question 1

```
    7 |include <string> char title[50];
      |include <iostream>     ^~~~~
antti@AntinJauhoMylly~ > ^K
antti@AntinJauhoMylly~ > g++ Documents/Cpp2025/lab3/q1/main.cpp
Documents/Cpp2025/lab3/q1/main.cpp: In function 'int main()':
Documents/Cpp2025/lab3/q1/main.cpp:22:15: error: 'char Book::title [50]' is private within this context
   22 |         book1.title ="test";
      |               ^~~~~pages;
Documents/Cpp2025/lab3/q1/main.cpp:7:22: note: declared private here
    7 |       protected char title[50];
      |                  float p^~~~~
Documents/Cpp2025/lab3/q1/main.cpp:22:21: error: incompatible types in assignment of 'const char [5]' to 'char [50]'
   22 |         book1.title ="test";50];
      |         ~~~~~~~~~~~~^~~~~~~~thorInfo(){
Documents/Cpp2025/lab3/q1/main.cpp:23:28: error: 'char Book::title [50]' is private within this context
   23 |         std::cout << book1.title << std::endl;
      |                            ^~~~~
Documents/Cpp2025/lab3/q1/main.cpp:7:22: note: declared private here
    7 |               char title[50];
      |                    ^~~~~
      |nt main(void){
antti@AntinJauhoMylly~ >
E   13:         book1.title = "test";     ■■ Array type 'char[50]' is not assignable
E   14:         std::cout << book1.title << std::endl;     ■ 'title' is a private member of 'Book'
    15 }
main.cpp                                                                    7,6-20        All
```

a) This could be fixed by either making a function for changing the variable inside the class or by making the variable public.
b) The program compiles, because the publisher variable is public. It's not always the best to use public, because private and protected both protect the variable so it cannot change.
c) output:
test
Peter (end of output)
The function works because it's a member of the class Book and it's public.
If the function is private it cannot be accessed from outside the class.

```
book1.displayAuthorInfo();      ■ 'displayAuthorInfo' is a private member of 'Book'
```

d) The access modifiers protect the variables etc. so that they are not changed by things going on outside the class, which is important for enforcing the class rules.
e) Classes are for "modeling" more complex things with primitive data types and defining how they interact with each other and the outside, creating a new kind of variable.

Question 2
output:

```
Antti
1
F
name: Antti age: 1 grade: F
```

a) Program can be compiled, because instead of trying to access the private variables directly setter and getter functions are used.
b) Private attributes can either be accessed with functions or with the constructor.
c) group (string) = "2025 group" is a private variable of type std::string and with the predetermined value of "2025 group"

Question 3
output:

```
Car Toyota from 1987 created
brand: Toyota year: 1987
Car Toyota destroyed
```

a) The constructor is a member function that is run when a new object is made it takes values and based on them assigns the necessary variables.
b) The destructor is the opposite of a constructor and gets run when delete is used.
c) constructor: Car(std::string n, int y) : brand(n), year(y) {}
destructor: ~Car(){}
They cannot return values because they are called automatically and there is nothing to return the value to.
d) When an object is created on the stack it is tied to the context it was created in and gets automatically deleted when the function stack “collapses”. If the object is created in the heap it can be easily accessed by different parts of the program and since it isn’t part of the stack doesn’t get automatically deleted so delete has to be used.
e) If no constructor is defined nothing gets run when the object is created and private attributes will have to be assigned values with other means.
f) If no destructor is defined the object is still deleted by delete, but nothing else happens.

Question 4

```
Starting engine of Vehicle
brand: wroom, year: 2000
brand: Toyota, year: 1987, number of doors: 2
Car engine is starting!
brand: Skoda, year: 2024, number of doors: 4, battery capacity: 0
Electric engine is starting... silent but powerful
brand: Skoda, year: 2024, number of doors: 4, battery capacity: 1000
Electric engine is starting... silent but powerful
```

a) Inheritance allows multiple sub-classes to have the same parent so they all share some of the same attributes and methods if they are public or protected, private attributes cannot be accessed without setters or getters.
b) Constructor overloading allows for multiple ways of constructing the object so that for example default values can be assigned if not provided.
c) If the base class doesn’t have a default constructor it will complain about it as follows

```
■ Constructor for 'Car' must explicitly initialize the base class 'Vehicle' which does not have a default constructor
```

d)
brand: Nissan, year: 2022, number of doors: 4, battery capacity: 40
Electric engine is starting... silent but powerful

First the object eCar is created and the constructor called with the required arguments. Then the member funcitons of eCar are called to print out the attributes and to start the engine.