

## Lab3 Q1

A.) It can't be compiled because the title is private, it should either be changed to public, or we'd need to use a 'getter/setter method' (function defined inside of the class) Why inside of the class?  
Private attributes can only be accessed or modified inside of the class.

B.) The "Publisher" is not protected nor private so we can set it without problem in main. It's not good to always use public for security reasons.  
Setter/getter functions can control the way people can access the variables.

C.)

Why the displayInfo works:

-The displayInfo is a public function so it can be called in main, and it can access the author because it's defined inside of the class, private attributes can only be accessed and modified inside of the classes themselves.

What if the DispInfo was private?

-We couldn't call it in main as is.

D.)

Private: Can only be accessed within the class

Pros: Completely controlled how the data can be accessed. Very safe.

Cons: Less flexible

Protected: Can be accessed in the class or in child classes

Pros: Hides the data (like private) but more flexible

Cons: Potentially exploitable in child classes.

Public: Can be accessed anywhere.

Great for calling setter/getters for example.

Pros: Very easy to use and Access and no restrictions.

Cons: All the data is public, and no control over how it's used.

E.

- Classes are a great way of grouping up multiple different data types into a single Group which makes it way more manageable. Classes are great whenever you want to reuse and manage a specific Group of variables, for example our book situation: Title, pagecount, author, price, Publisher etc.

- ⊗ 'std::string Book::title' is private within this context gcc [Ln 52, Col 11]
- ⊗ member "Book::title" (declared at line 37) is inaccessible C/C++(265) [Ln 52, Col 11]
- ⊗ 'std::string Book::title' is private within this context gcc [Ln 53, Col 32]
- ⊗ member "Book::title" (declared at line 37) is inaccessible C/C++(265) [Ln 53, Col 32]

Author: Peter

PS C:\Users\Papu\C\_plusplus25>

## Lab3 Q2

A.) the Program can be compiled because we use setters and getters to Access Private variables.

As stated previously, Private variables CAN be changed WITHIN the class itself.

The setters and getters are public, and inside of the class- so they may Access the Private data.

B.) setters are good for modifying Private variables,

We could even add some rules to determine that for example age has to be between 1-120 etc. Or that something gets flagged when grade is F (failing) and so on.

Getters are good for accessing attributes in a read only fashion.

C.) String Group = "2025 group" is an identifier that we didn't make a getter/ setter for- so it's a default for all students entered into the class at the moment.

Probably the class name should've been Student25 or something to make it clearer that it's not for just any students.

Name: Papu Saroma

Age: 25

Grade: B

Group: 2025 group

PS C:\Users\Papu\Cplusplus25>

A.) Explain what is constructor ? Why do we need to use constructor

- Constructor is something that gets automatically called when a new object in the class is created. It has the same name as the class. It should include all the needed variables for the class.

B.) Destructor gets automatically called when an object gets out of scope or is deleted manually as in our example. if we used `Car Car4("brand xy",2023);` to create a car on the stack, it would get called automatically instead of the manual deletion. Destructor should have the same name but a tilde in front of it (`~Car`)

Destructors free memory

C.) Syntax in code\*

Can we return a value from them?

No, the debugger immediately tells us that it's not the job of them to return anything, they just make or delete an object in the class.

D.) New allocates memory on the heap, while objects on the stack get cleaned when out of scope. I tried this in the code and objects on the stack call the destructor automatically.

E.) We could just use setters instead of a constructor I guess, so it shouldn't be a problem to not have one. Deleting the constructor got the compiler to tell me that there wasn't a constructor and that was a problem as I was trying to call one in the code.

F.) If you don't define a destructor, the compiler doesn't say that it's a problem.

I assume the compiler still deletes the objects, but as I don't have the `cout` telling me that it does, I won't know that they were deleted.

Car Brand1 from 2020 created.

Brand: Brand1, Year: 2020

Car Brand1 destroyed.

Car Brand2 from 2021 created.

Brand: Brand2, Year: 2021

Car Brand3 from 2022 created.

Brand: Brand3, Year: 2022

Car Brand3 destroyed.

Brand: Brand2, Year: 2021

Car Brand2 destroyed.


Car Stack1 from 2023 created.

Brand: Stack1, Year: 2023

Car Stack2 from 2024 created.

Car Stack2 destroyed.

Car Stack1 destroyed.

PS C:\Users\Papu\C\_plusplus25> 

A.) Inheritance is when you make a child class that inherits some variables from the parent class. It lets you reuse the code and add more things on top of it.

A child class can only Access Private and protected members of the parent class.

Privated parts of the parent class can Still be accessed through setter/getter methods.

B.) In this context I'm not completely sure why we'd have a default constructor- but we could make the object without knowing all of the parameters which is nice.

C.) It doesn't compile without the constructor.

D.) Picture provided of the printout



Starting engine of Vehicle

Brand: Toyota, Year: 2021

Number of doors: 4

Car engine is starting!

Brand: Default-Option, Year: 0

Number of doors: 0

Battery Capacity: 0 kWh

Electric engine is starting... Silent but powerful!

Brand: Tesla, Year: 2077

Number of doors: 4

Battery Capacity: 88 kWh

Electric engine is starting... Silent but powerful!

PS C:\Users\Papu\C\_plusplus25>

Brand: Nissan, Year: 2022

Number of doors: 4

Battery Capacity: 40 kWh

Electric engine is starting... Silent but powerful!