

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | Car Show | | | | |  | |
|  |  | | | | | | |  |
|  | | | |  |  | | | |
|  | | | | Saif Ahraf Ramadan Hamadah |  | | | |
|  | | | | 4/6/2023ProgrammingAsma Lafi |  | | | |
|  | | |  | | |  | | |

**Part 2 :**

**Task 1 of part 2 :**

In this report for the programming course final project part two, We will describe the three paradigms, Procedural programming paradigm, Object-Oriented programming paradigm, and Event-Driven programming paradigm. In addition to that we will write a system with the three paradigms for the same scenario, we will discuss the different characteristics of each programming paradigm, Also we will explain the debugging process to develop our code and we will discuss some coding standards that we have followed in our system.

**2.1.1** The Programming paradigm is the way or the style of programming in which way we have written a

specific program. The paradigm or the programming paradigm is not a concrete thing but the programming language is a concrete thing so we can’t use the phrase “Programming language paradigm” because the programming paradigm is the way how we write the program but the programming language is a language that we use to develop a specific program, not the way how we write the program. We have many programming paradigms that we can use in writing any program we want, like the procedural programming paradigm, Object-

Oriented programming paradigm, and event-driven programming paradigm. Some paradigms are easy to write with some programming languages but not other programming languages like the object-oriented programming paradigm it is easy to use this programming paradigm with Java, On the other hand, It is harder to use it with C programming language it needs some techniques to use with C, **There are different programming paradigms** **[1].**

The procedural programming paradigm is a linear top-down approach, Which means the execution of the

program starts from the top and then executes line by line from the top to the end of the program. The main component of the procedural programming paradigm is the functions. The functions break down the program into smaller parts called the functions and every function has a specific task to perform when we call the function in the main method by its name, Every function has its local variables that we can use only on the function that we can’t use on other functions and there are global variables that can be accessed by all the functions, There are functions with a return value, on the other hand, we have void functions that don’t have a return value, Also we have functions that take an argument, on the other hand, there are functions don’t take an argument. The program flow can be tracked easily because it is a linear top-down approach but unsuitable for complex and real-world tasks because it focuses on the functions, not the data itself **[1].**

The object-oriented programming paradigm has many features like encapsulation to keep the functions and

data safe from any outside access, the only way to access the data is by using functions like setter and getter

functions. Inheritance is a relationship between different classes where is a class called a parent

class(superclass) where the properties are inherited and other classes called child class(subclass) which

inherited the properties from the superclass. The composition when we have a class is a part from another class

like when we say an engine is a part of the car, we have the engine class as part form the car class we make an object from the engine class (part class) into the car class (main or the default class). Methods, every class has a

specific method to implement when we make an object from the class and call the method. Attributes are the

inputs and the output of the class, Every class has its own attributes that refer to the class. We have a modifier for the attributes and the method, We have the access modifier like public access modifier where we can access the data from anywhere in the project, a default access modifier where we can access the data anywhere in the same package, private access modifier where we can access the data only on the same class, protected access modifier where we can access the data from the same package and sub-class outside the package, Then we have a non-access modifier like the static that related to the class and we don’t need to create an object to access it, only we need a static method to call the static variable and we access it through the class on the object. Constructors We

use the constructor to build the objects, we have a default constructor that the compiler creates from itself if we do not create a constructor, a No-argument constructor that we create but does not take any arguments, and the

parameterized constructor that we create it and it takes arguments. Polymorphism is when we have a class and

we can make it in different shapes, it’s like inheritance but here the sub-classes inherited all the attributes from

the parent class but in different values which makes a different shape from the superclass. Abstraction, The abstraction class is a blueprint for other classes, we can’t access this class directly. But the main components of the object-oriented are classes and objects, where the object is an instance of a class and the class is the template for the object like the car is a class and Audi is an object from the car class. It is easy to develop a project with

this paradigm especially if the project expansion in the future means this paradigm is more manageable than the procedural programming paradigm. It gives us more security for data over the procedural programming paradigm where we can access global data from anywhere, This paradigm is more suitable for complex and real-world tasks as it deals with the projects effectively more than the procedural. To achieve the score of the object-oriented programming paradigm you should follow these practices: Don’t repeat yourself you should never write something twice in different places of the code, Encapsulate the functions and attributes using private to protect your data if there are any changes on the project in the future, Single Responsibility Principle which means every class should have only one function, and Open Closed design which means our project should be closed for any modifications but opened for any extension. The class is the major thing in object-oriented cars and the object is an example of the class like Audi is an object from the class car. Every object has it is own copy of the class methods and attributes as the class is a blueprint for the object, the class contains attributes and methods then the object has all the methods and attributes of the class, We create the objects in the main method using the constructor and then we can access all the methods and attributes of the class through the object**[1]**.

The event-driven programming paradigm is a programming paradigm that we can implement

in any programming language in which the user can control the program flow by the events. The event-

driven paradigm uses one of the other programming paradigms, procedural programming paradigm, or object-oriented programming paradigm, and works with it at the same time. Before the object-oriented, event-driven has been implemented with the procedural paradigm and the flow of the program was depend on the application’s routine requires a highly structured program, and the whole project would be written by the programmer to make sure that the events are handled to manage the flow of the program. After the object-oriented programming paradigm, the event-driven would be implemented with the object-oriented as there is not a visible flow of the program and the routine depends on an event loop that waits for the event to occur, The code of the event loop has been provided by the event-driven environment as it is not visible for the programmer. One of the most applications event-driven is the Graphical User Interface(GUI) like toggles, list boxes, text fields, buttons, and checkboxes. Our task is to create the user interface object in the frame, create the frame, make the design like the size of the frame, behavior on the close operation, layout of the frame, visibility of the frame, and add objects on the frame like a radio button that only we can tick a one choose, put them on a button group, write a code for each object to respond to a specific event, add button(action listener)and write a code for the action when the event occurs. The main components of the event-driven are the event that will occur like a key press when the user presses any button, The event listener that will trigger the action when the event occurs, It’s responsible for detecting the events, Event handling that is responsible for all the event process(event/event listener/action to be performed/event loop), the action to be performed when the event occurs, Some code was written to be implemented when the event occurs, and event loop that waits for another event to happen to make the process again,The event-driven has two libraries for the objects of the GUI, We have the awt library then we have the developed library swing library, It depends on the awt library because the awt library is a part from the swing library, These libraries are responsible to make us able to use the objects of the GUI.**[1].**

**2.1.2** I have developed a car show system like the “open sooq” to search for a specific car for many purposes in the future like sales and rents purposes. I have used the three programming paradigms in the same scenario, In This system we use it to find a specific car we want.

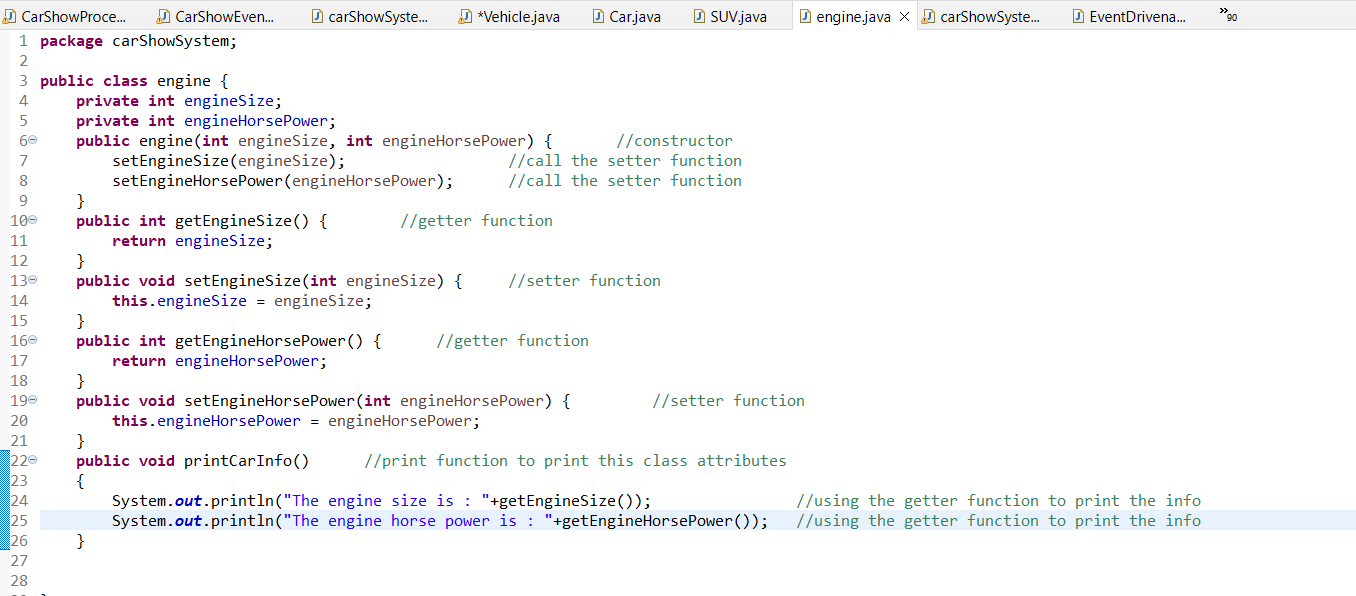
How the object-oriented operates: we have the vehicle class as the superclass, the car class is a sub-class from the vehicle because the car is a vehicle, then we have the sedan class and SUV class, they are different shapes from the car class, and the engine class is for composition because the engine is part from the car. So the customer can put the options of any vehicle, then the car itself, identify the shape of the car or the class of car, and put the engine of that vehicle by creating an object from the car class.

How the procedural operates and the event-driven: They are the same as the object-oriented but the user put the information for the system.

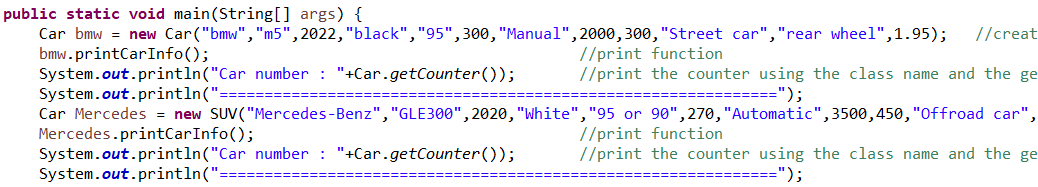
**2.1.3** Now we will compare and contrast the characteristics of each paradigm that we have used in our system with some screenshots from the code :

* **Object-Oriented Programing paradigm features :**

1. We have two main components in the object-oriented paradigm, The first component is the class, any class has methods and attributes, and the class is the blueprint of the object, Here you can see the screenshots of the class :



1. The second main component in the object-oriented is the object, the object is an instance from the class, We create it in the main method using the constructor, Every object has its own copy of the methods and attributes of the class, Here you can see the screenshot of the object :



1. The encapsulation is one of the features that protect the data from any outside access and make a header for the data using setter and getter functions to have the ability to access the data globally through the name of the function, we use the setter to give the value of the attribute and the getter to return the value of the attribute, Here you can see the screenshots of the encapsulation :

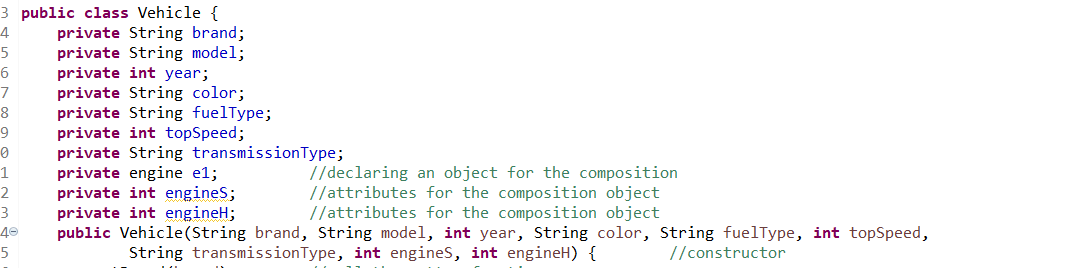


**Here is how we have accessed the attribute globally in the main method using the setter function:**

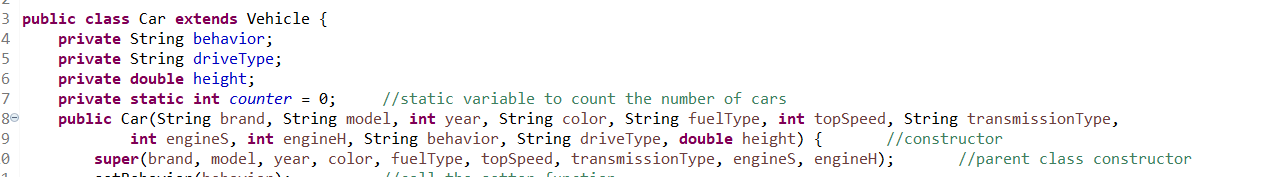


1. Inheritance is when we have a parent class and a sub-class that inherited the attributes and the methods from the parent class, we use the word (extends) on the sub-class then the parent class name, then we adjust the constructor of the sub-class and add the attributes of the parent class then we add the super constructor on the sub-class constructor, also we can access the methods of the superclass on the sub-class using (super . the method name), Here you can see the screenshots of the inheritance :

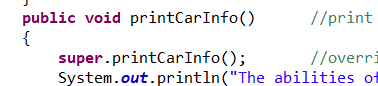
**Superclass:**



**Sub-class:**

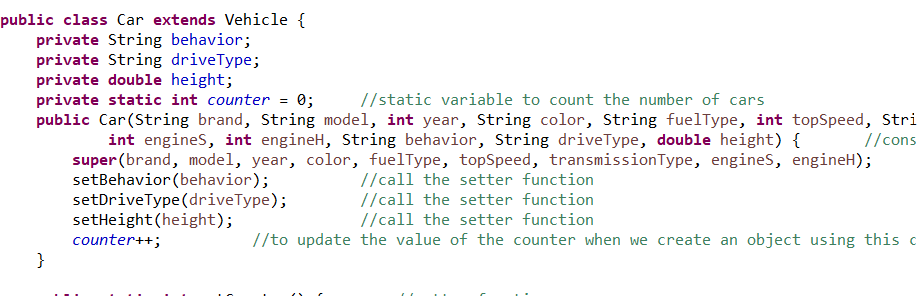
****

**Access the super function from the sub-class :**

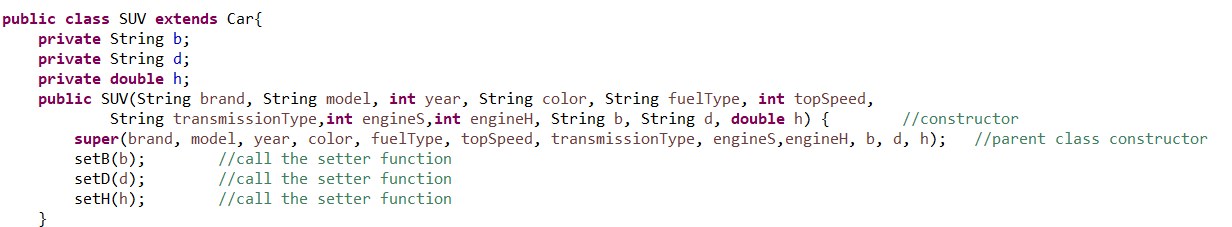


1. Polymorphism is one of the inheritance applications, But here the sub-class uses all the attributes of the superclass without any special attributes for the sub-class, In polymorphism we make different shapes from the superclass in the sub-class, which means the sub-classes are the same as the superclass but in a different shape(different value of the attributes), Also we can create an object from the superclass using the constructor of the sub-class, Here you can see the screenshots of the polymorphism :

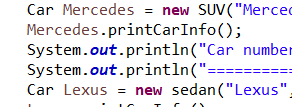
**Here is the superclass (note: there is an inheritance for another thing in the code):**



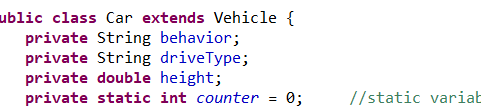
**Here is the sub-class that is a different shape from the car class with the same attributes:**



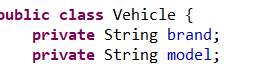
**Here we have created an object from the car using the SUV constructor :**



1. The attributes, Every class has its own attributes that are special for it, Here you can see the screenshots of the attributes :



**The private access modifier that only within the same class can access the data, and we use getter and setter if we need to get access from outside the class :**

****

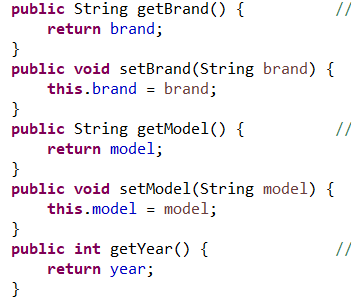
**The static non-access modifier that is related to the class not to an object and can be accessed anywhere using the class and static method:**

****

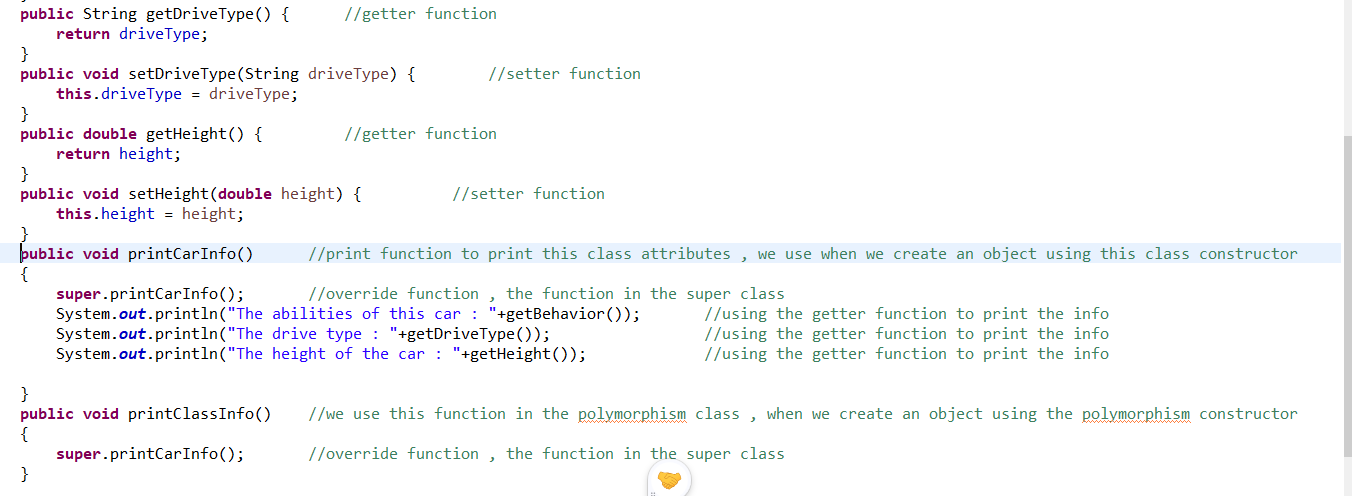
**Here is how we have accessed the attribute through the class, not the object :**

****

**Here is the public modifier that we can access the function from anywhere using the object :**

****

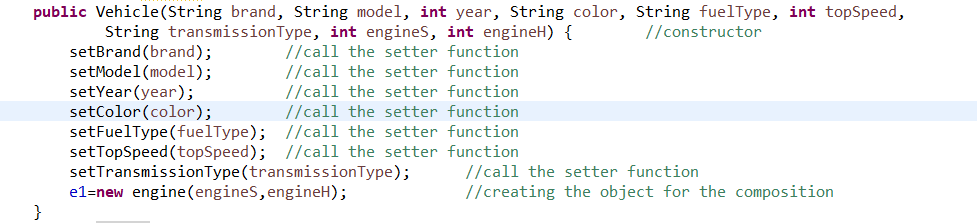
1. The Methods, Every function has its methods with public or private……. Modifiers, Like the setter and getter functions to set and get the private attributes and print function to print the information of the class, We use the name of the function to implement the function on the object of the class :



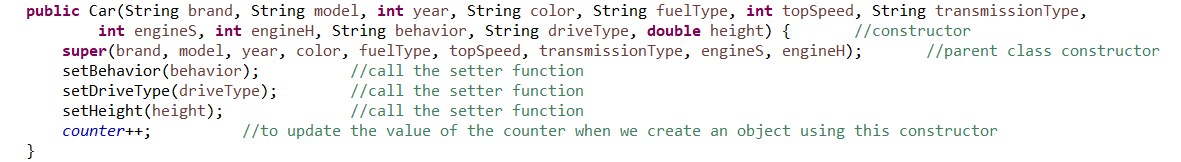
**Every function has a header (modifier + return value + name of the function+(arguments) ) and the function body that contains the task of the function(the function is responsible for one task only) :**

****

1. Constructors, we have three types of constructors, The parameterized constructor that takes arguments, the default constructor that the compiler creates it with default values for the attributes, and no argument constructor that doesn’t take arguments. Every class has a constructor, We use the constructor to create the objects from the class, Create the attributes (give them values), and create an array and objects from other classes (composition), Here are screenshots for the constructors :( **we use it with public access modifier to create an object from the main class**).



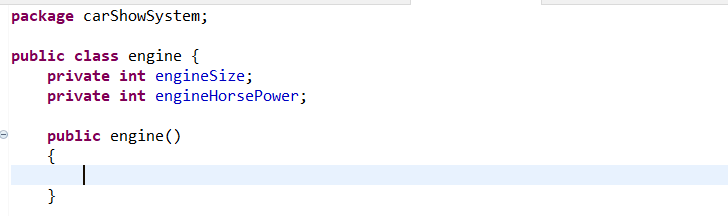
**Here is the super constructor in the sub-class :**



**Here is how we call the constructor on the main to create a new object from the car class :**



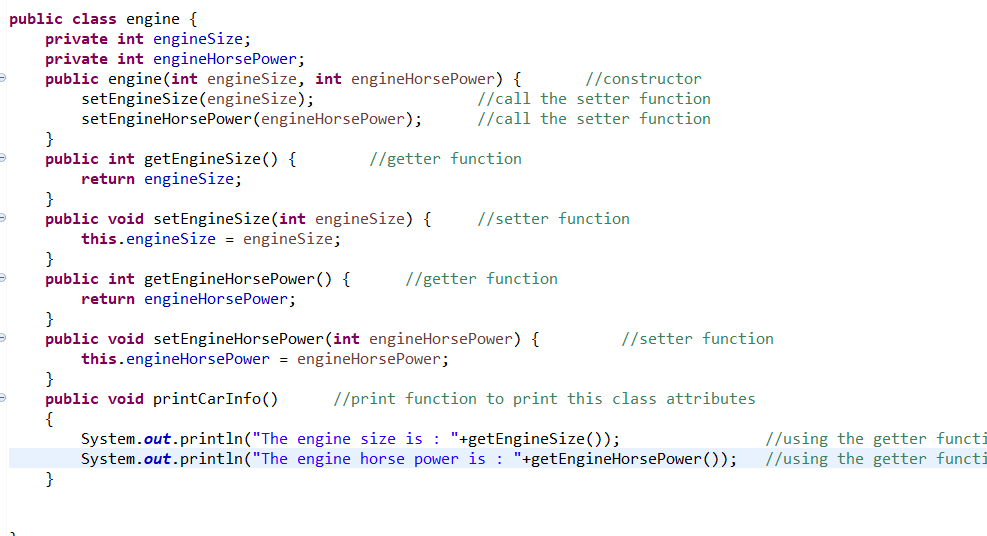
**Here is the no-argument constructor**:



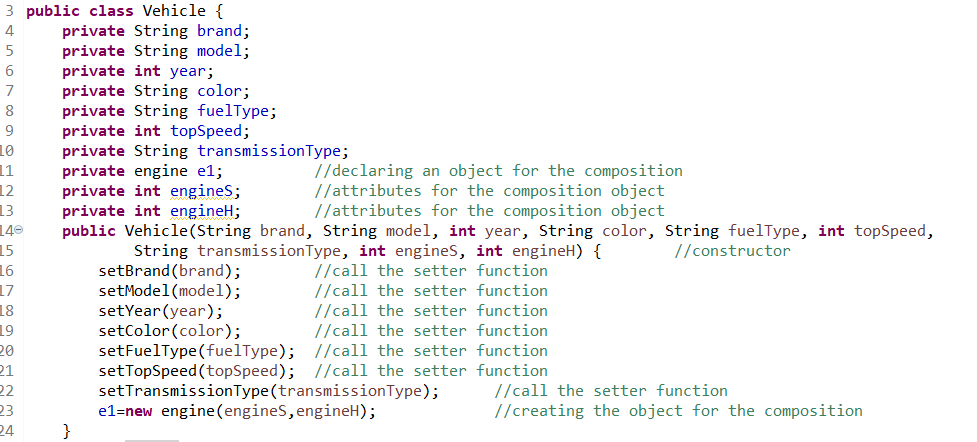
**And if we do not make a constructor, the compiler will make a default constructor.**

1. Composition, When we have a class that is a part from another class like an engine is a part of vehicle, We use composition, We declare an object from the part class on the essential class then we create the object on the constructor of the essential class, and then we need to send an object to the essential class for the part class, Then we can access the methods of the part class on the essential class using the name of the object that we have created (name of object . name of the function). Here are screenshots for the composition:

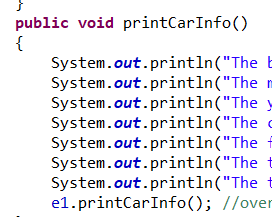
**Here is the part class (engine is part of the vehicle):**



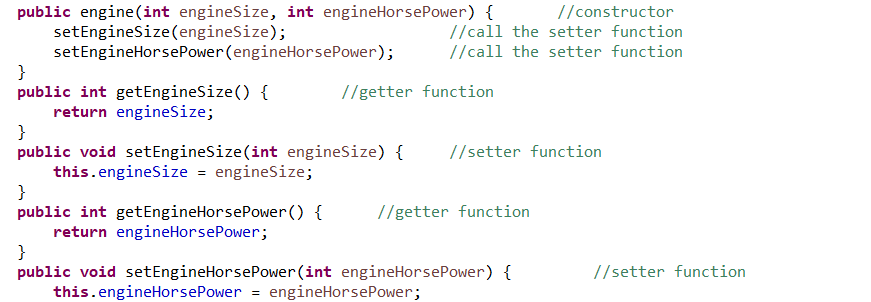
**Here is the composition of the essential class (vehicle class):**

****

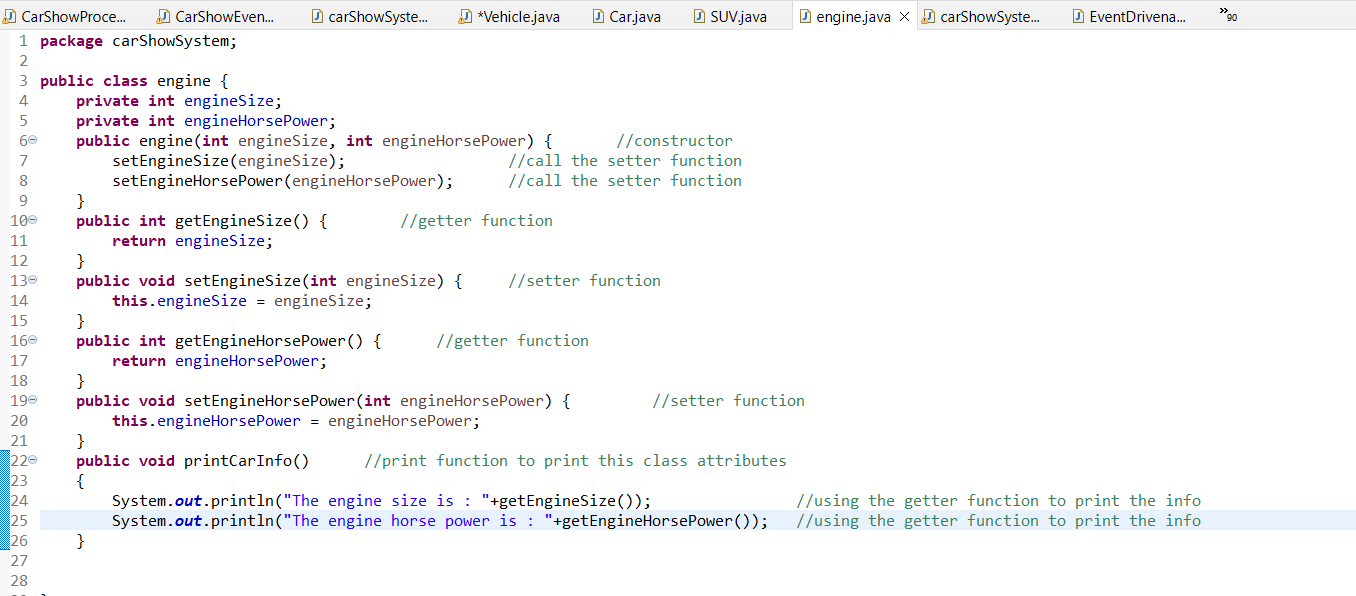
**Here we can access the print method for the engine on the vehicle class through the name of the object because the method is related to the object:**

****

1. Don’t Repeat yourself: if we have something that we have written we can’t write it again twice, we should put it on a function, Like here we call the setter function because we don’t want to write (This . x = x) twice, the first one in the constructor and the second one in the setter function so we call the setter function:

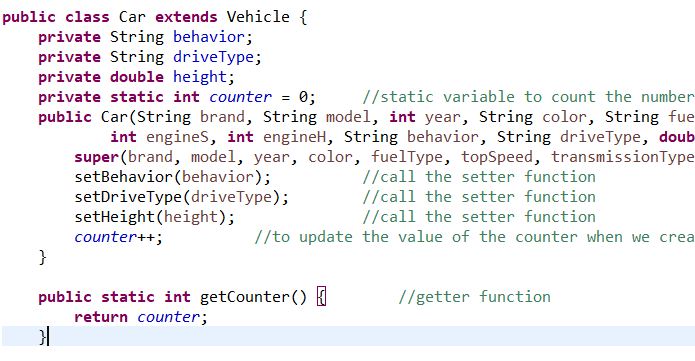
****

1. Single Responsibility Principle, Every class should only have one responsibility to implement it, like here the engine class, is only responsible for the engine and its attributes and methods:

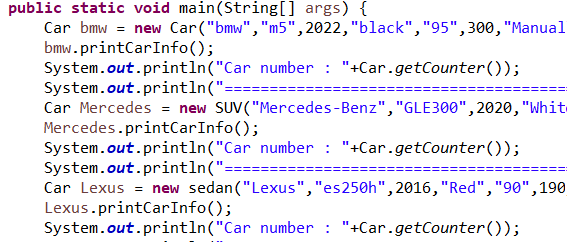
****

1. Abstraction**,** The abstraction class is a blueprint for other classes, we can’t access this class directly, like when we access the class from another class, here we don’t access the class directly:

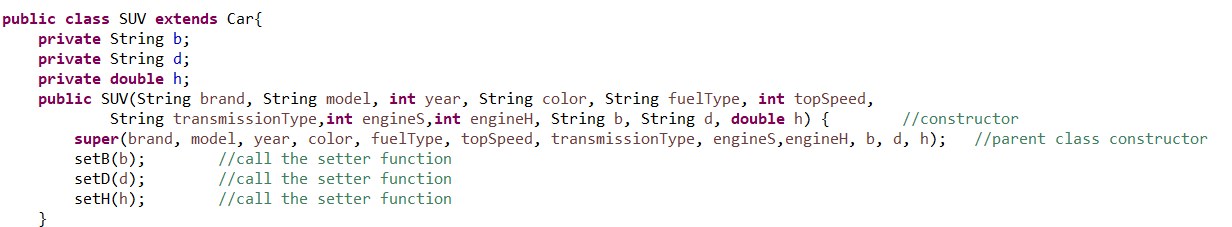
**Here is the vehicle class, we will make access to it through the car class :**

****

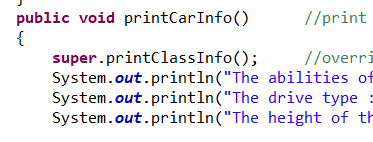
**Here we have made access for the vehicle class but not direct access, We have used the car class then the car class made access for the vehicle class, So we don’t create an object from this class direct :**

****

1. Open-closed design, When we have a class and then we want to create another shape from it so we want make an expansion but we don’t want to modify the code, Here we create class extends from the class that we want to make the shape from it, Here the SUV is a shape from the car so we make SUV class that extend car class without making any modification on the code of the car class and then we can access the methods and the attributes of the car class here in the SUV class :

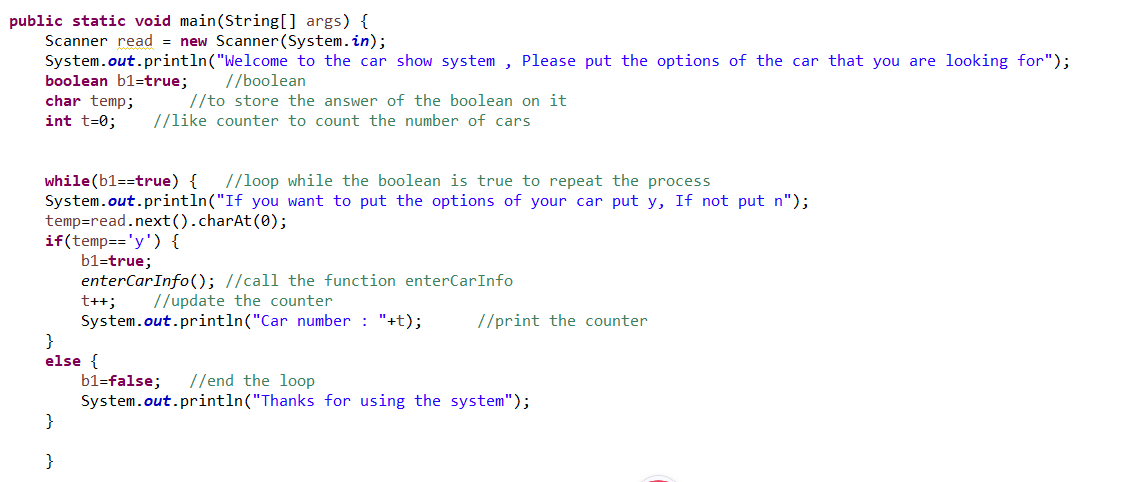
****

**Here we can access the function of the car class in the SUV class:**

****

* **Procedural programming paradigm:**

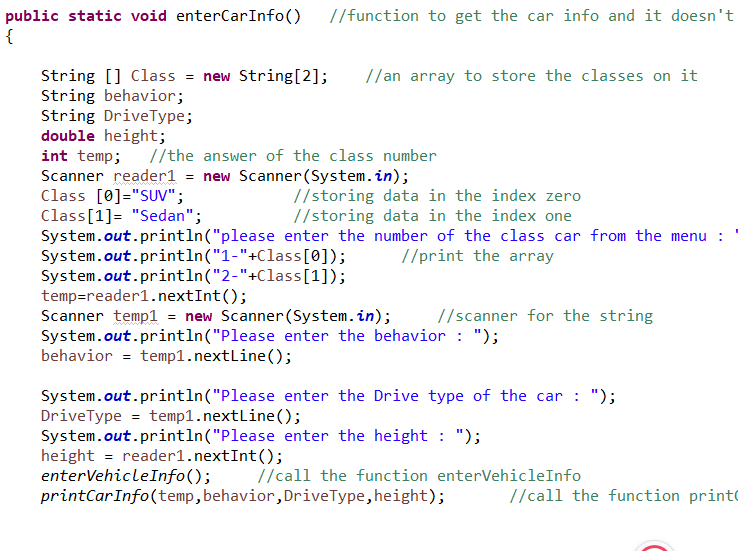
1. Linear top-down: The procedural is a linear top-down approach, It starts the execution of the code from the top and then line by line in order to the end of the code, if there is a call for a function it goes and implements the function from the top to the end of the function then return to the main function and go over of the implementation from its stopped when we called the function, Here are screenshots for the linear top-down process:



As you can see here the code starts from the top and then goes down, and then you can see there is a call for the function ”enterCarInfo” It goes for the function and implement it then returns to the main function from where it’s stopped when we have called the function:



**The function:**

****

1. The procedural depends on the functions to make the code more efficient and well-ordered, Every function has a single task to perform, With local attributes for every function, And the essential thing is the function header, Here are screenshots for the functions:

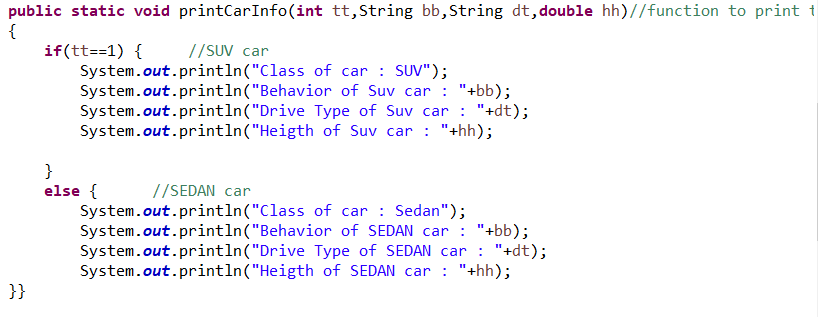
As you can see here this the function header we have the modifiers(public static) these are for the access of the function, Then we have the return type of the function, This return type is void which means this function doesn’t return any value, After that we have the name of the function to differentiate the function and to call by its name if we want to implement it anywhere in the code, And then we have the arguments that the function takes to implement its task and get the expected result from it, These arguments are local for this function and we can’t use them outside the function scope:



Here is a screenshot of the call of the function by its name and sending it the attributes that the function needs:



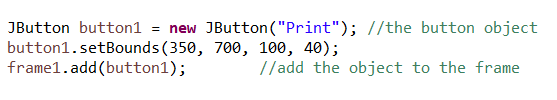
Here is a screenshot of the single functionality of the function that the function is responsible to implement:

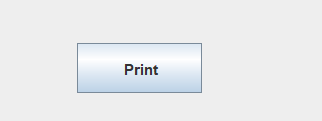


**This function is responsible for printing the car information only depending on the attributes that we have sent to the function when we called it.**

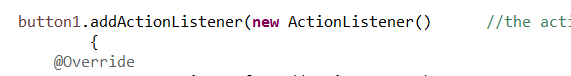
* **Event-Driven programming paradigm:**

1. Events, an Example of an event from my system, This is a button code, When the user press this button, That is what we call the event:

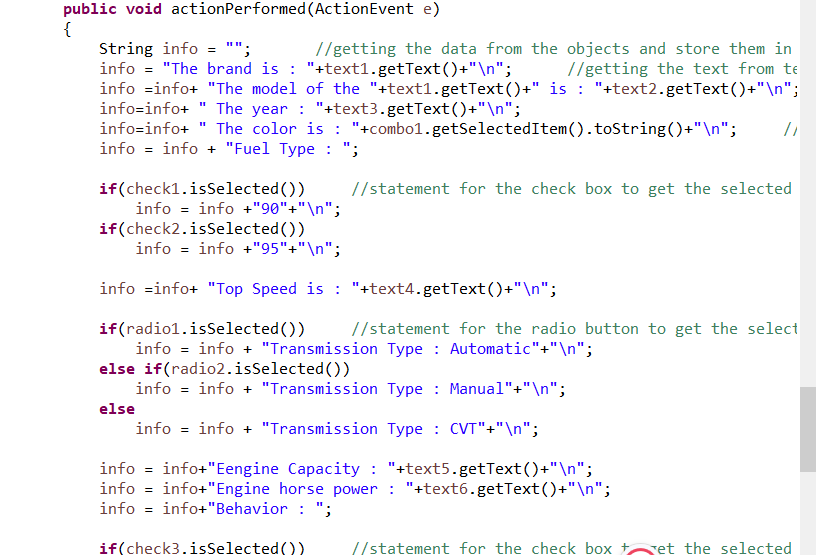


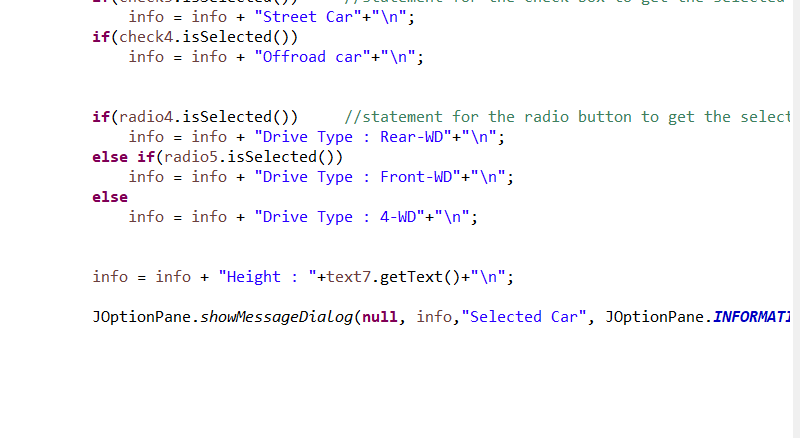
****

1. Action listener is the component that responds to the event to trigger the action:

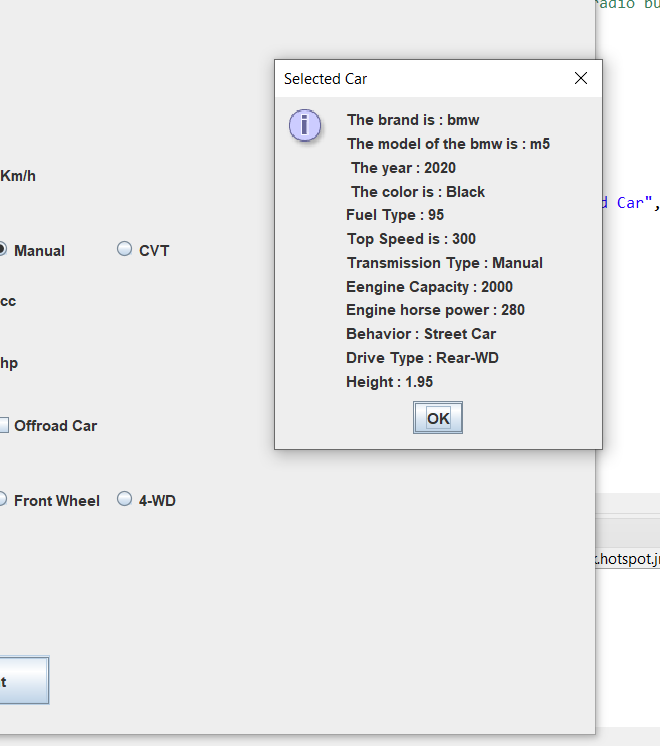


1. Action to be performed, This is the code that we write to perform when the action happens:

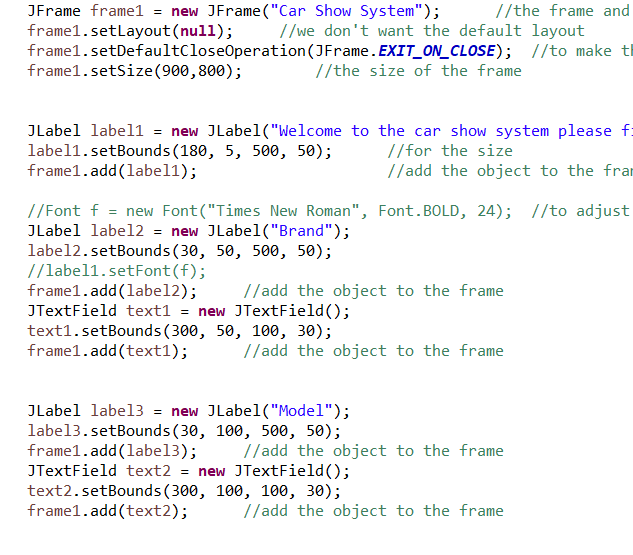


****

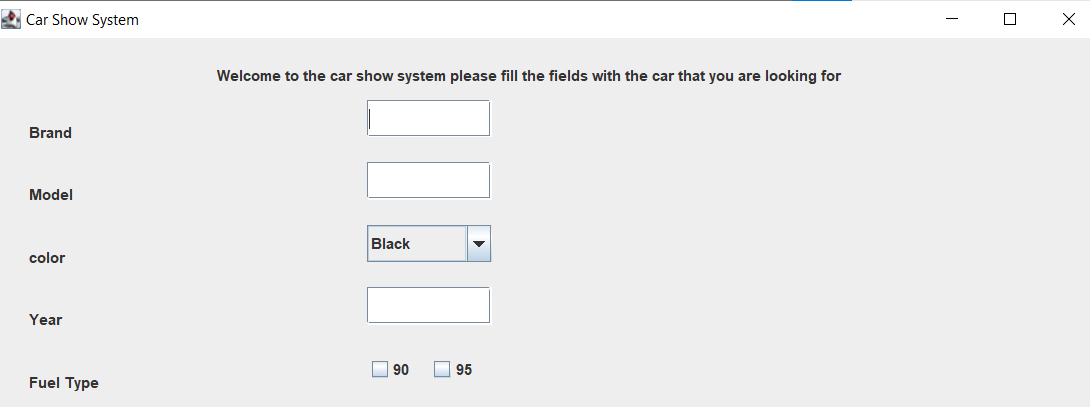
**And here is the action that happened:**

****

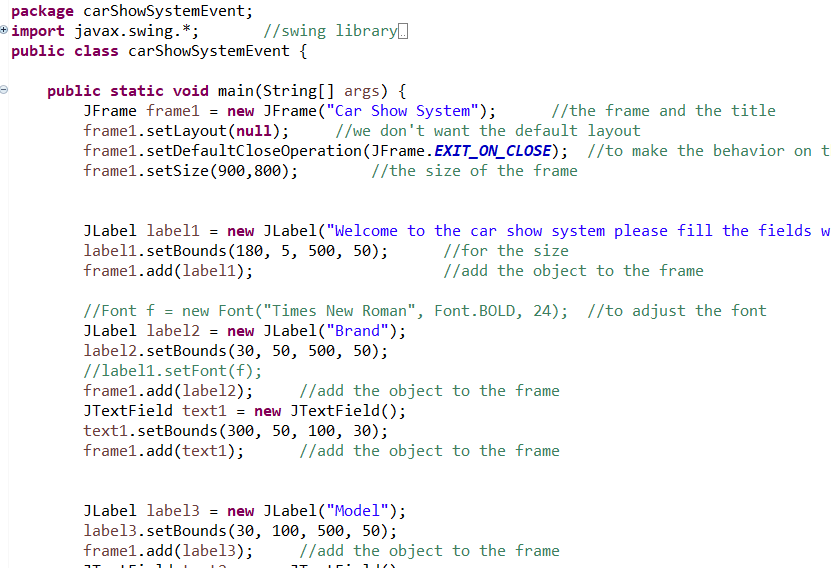
1. As we know the most application of the Event-Driven is the graphical user interface, We have many examples(objects) of the GUI that we add to the frame (JFrame) like:



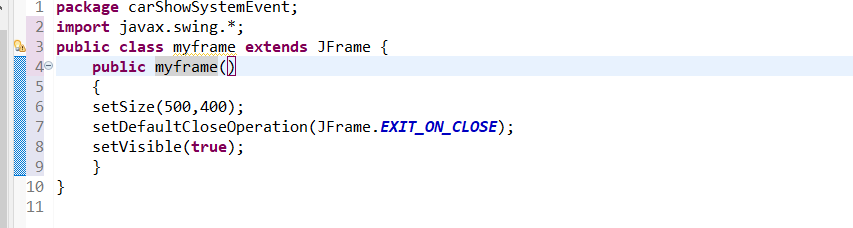
**Labels, Text field, Combo box, and Check box:**

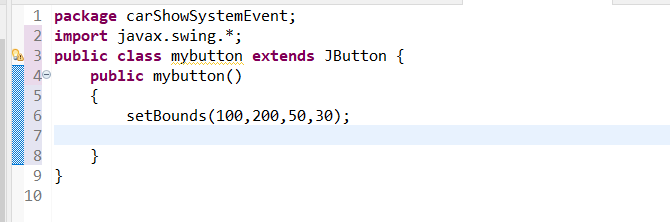


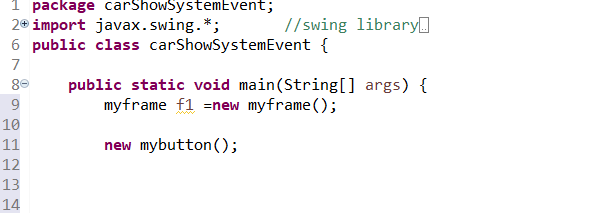
1. We can write the Event-Driven with procedural or object-oriented, Here I have written the code with the procedural paradigm:



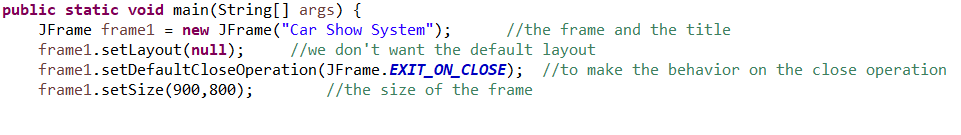
**The objects in the event-driven are classes that we don’t need to know how we create but we make objects from these classes and these classes we consider them as objects also, Here is the event-driven using the object-oriented:**

****

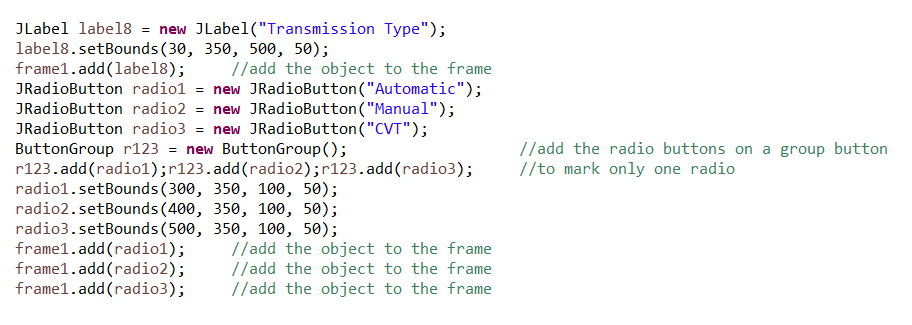
****

****

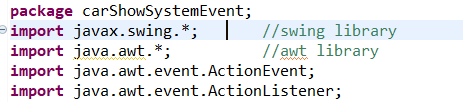
1. Here you can see how we make the design of the object (size, behavior):



7)Here is a screenshot for the button group to be able to tick only one choice:



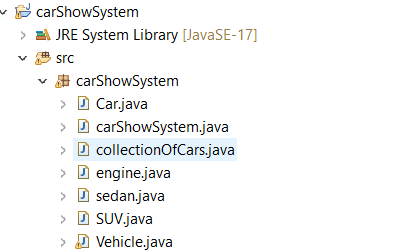
1. The event-driven has two libraries for the objects of the GUI, We have the awt library then we have the developed library swing library, It depends on the awt library because the awt library is a part from the swing library, Here is a screenshot how we imported the two libraries to use the objects that are stored in these libraries**:**

****

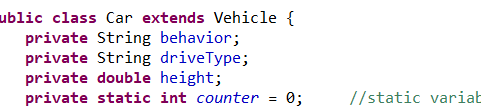
**2.1.4** In this course I have developed a car show system with the three programming paradigms, in this part I will critically evaluate the code that I have developed in terms of the code structure and the characteristics of each paradigm with some screenshots.

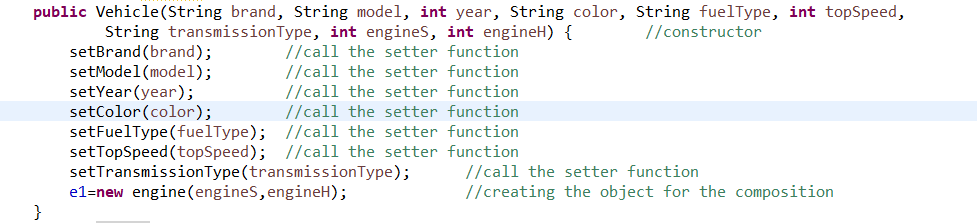
* **Object-Oriented programming paradigm:**

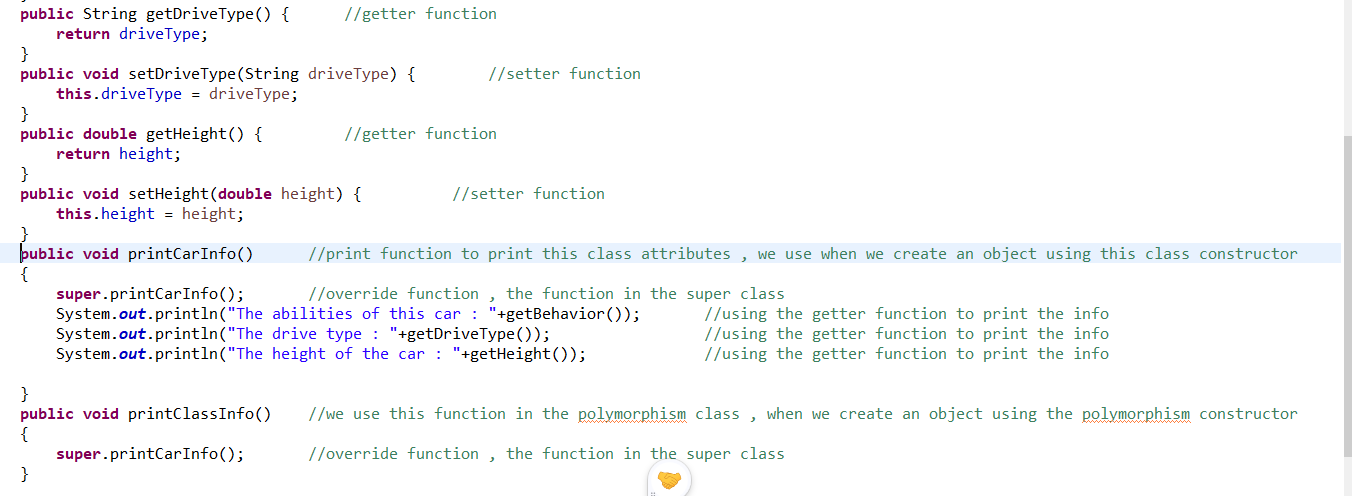
First, I started writing the system with this paradigm as it is the longest paradigm because there are many features that we implement with this paradigm, I have created seven classes with the main class, Two classes for the inheritance, Two classes for the polymorphism, One class for composition, The main class, And class for the array list to add the objects that we create to this array list.



As every class needs attributes and methods, I have written the attributes for each class with private access modifier and setter & getter functions, Then I have written the methods of each class with public access modifier to access the functions anywhere in the code, And then I have created the constructor for each class.

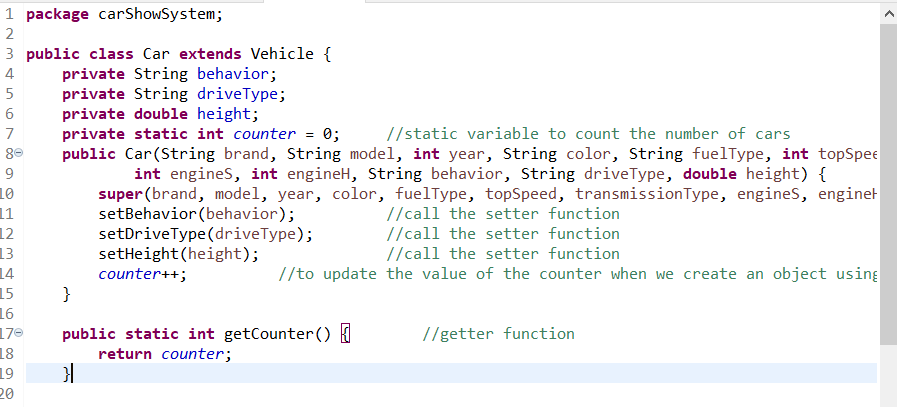


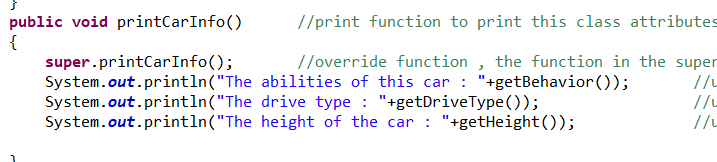
****

****

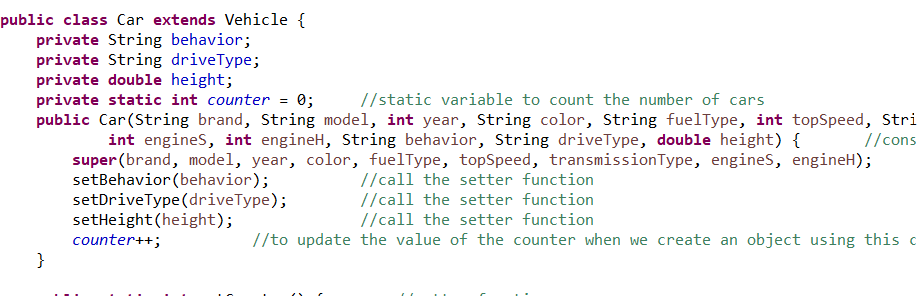
Then I made the inheritance between the vehicle and car class as the car is a vehicle, I extended the car class and modified the constructor of the car class, and add the super constructor to the car constructor, Then I made an override for the function of the vehicle class into the car class.

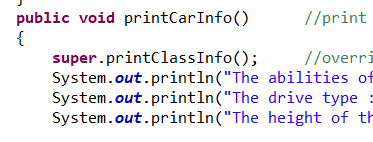




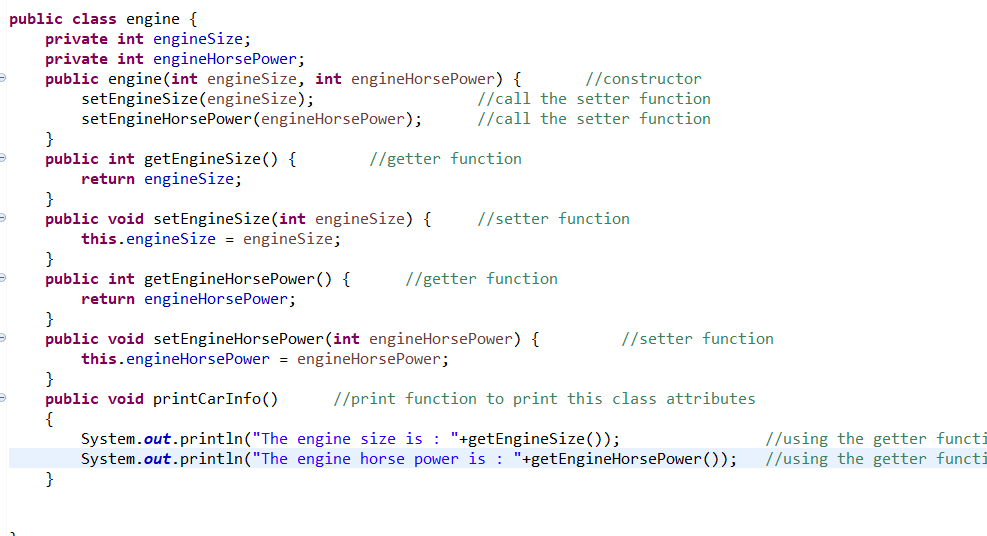


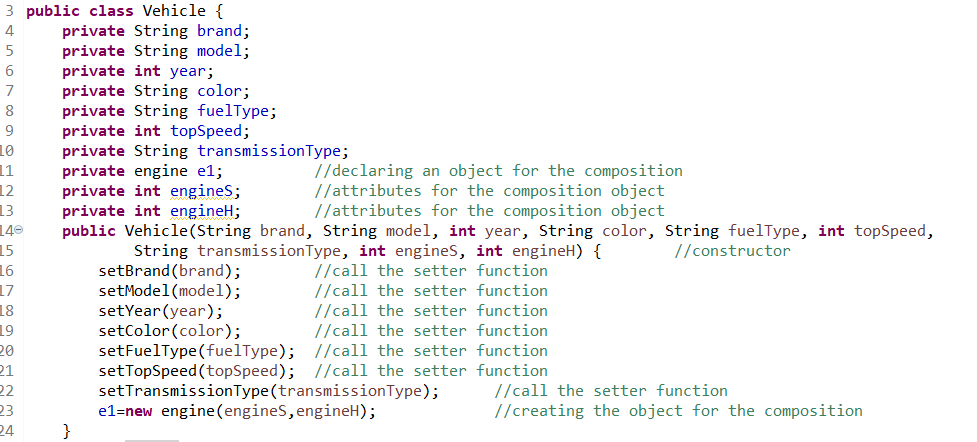
Then I wanted to use the polymorphism between the SUV class, Sedan class, and the car class, As these classes, are polymorphisms from the car class, So these classes inherited all the attributes and methods of the car class without any special attributes for them.

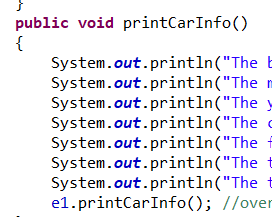




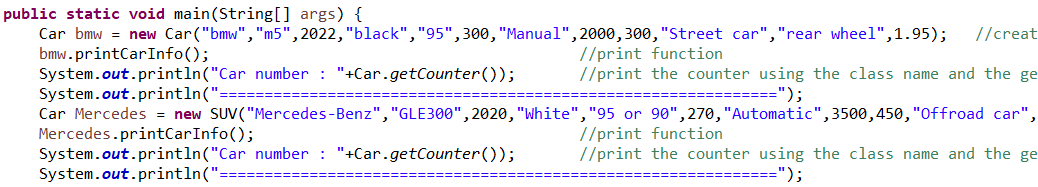
Then I did the composition between the engine class and the vehicle class, As the engine is a part of the vehicle, I declared an object from the engine in the vehicle class then I created the object in the constructor of the vehicle class, Then we can access all the methods of the engine class in the vehicle class using the object that we have created.



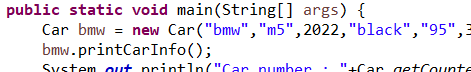




After we have finished creating the classes using the features of the object-oriented, we want to make objects from these classes, So first I created an object from the car class as it inherited the vehicle class, So we accessed the vehicle class but not directly, Then I made an object from the car class using the SUV constructor then the Sedan constructor as there is a polymorphism between these classes, So we have implemented the open closed design through the polymorphism.



In every object, I have accessed the method of the class (name of the object . name of the method).



Then I used the static non-access modifier in the car class to count the objects or the number of cars and

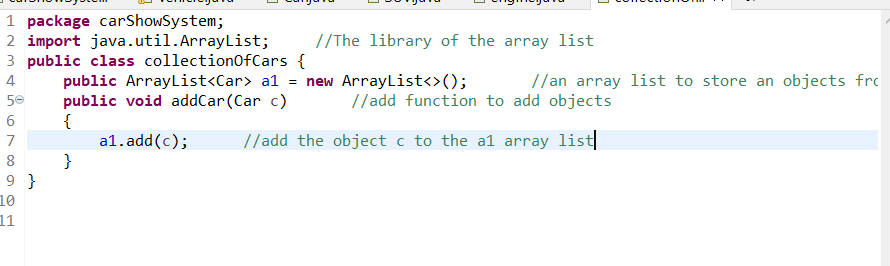
We have accessed this attribute through the class name, not the object name.

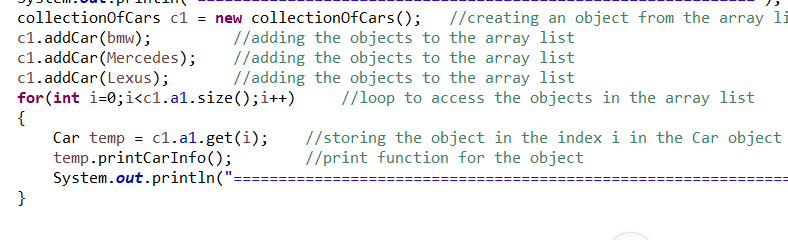






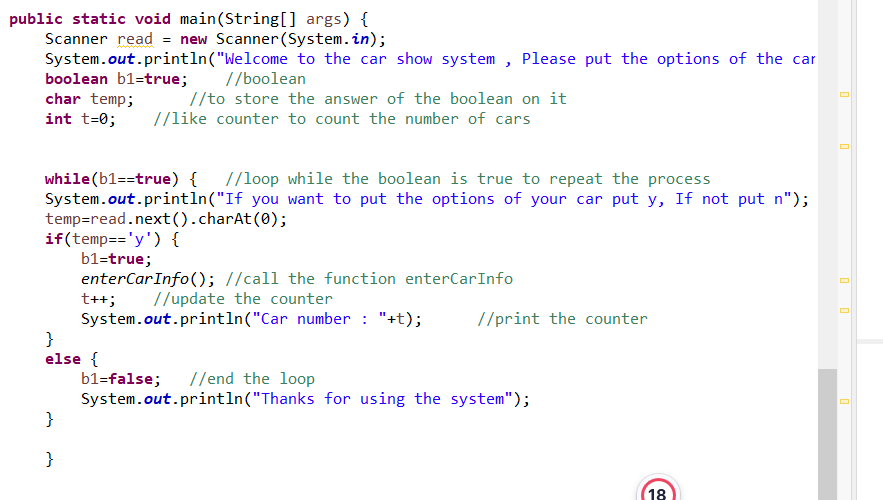
Finally, we want to add these objects to the array list, We have made an object from the array list class, Then we accessed the function add car through the name of the object, And we sent the objects that we have created to the function to store them in the array list, and then we have made a loop to get the objects that we have stored in the array list.

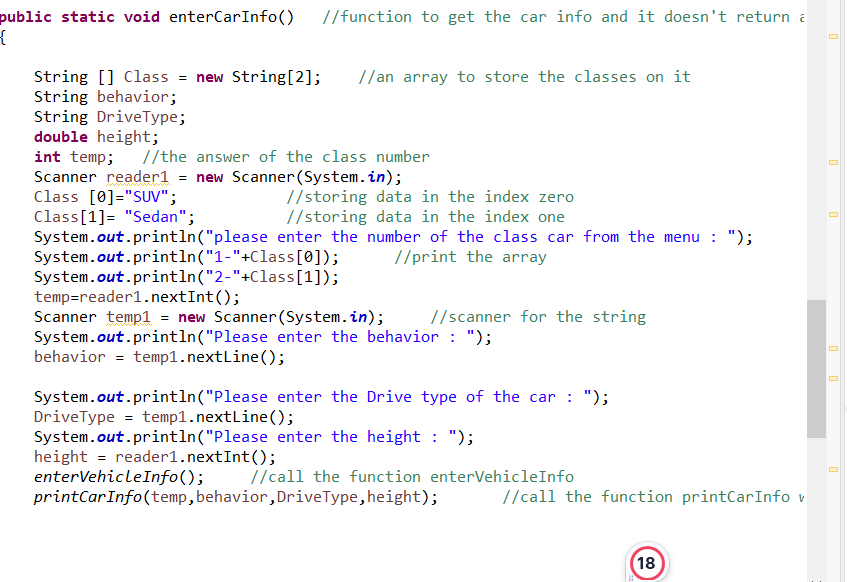




* **Procedural Programming paradigm:**

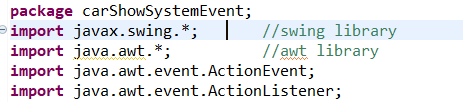
After I wrote the code with the object-oriented, I started writing the code with the procedural, I made many functions, Every function has a single responsibility, And I made the user give the options to the program, And in the main method, I have made a Boolean in a loop to repeat the program again if the user wants to enter the information of the car, and I have called function and in this function, I have called other function and so on. The program starts from the top to the down line by line until it finds a call for a function, it goes to the function and implements it then get back to the main method from where it stopped.

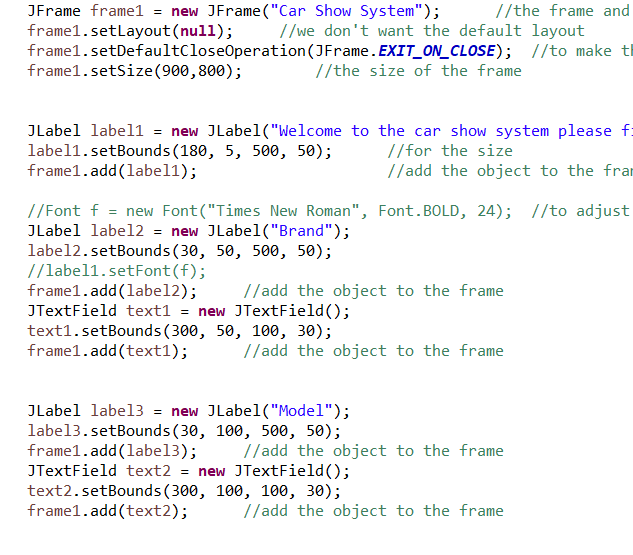


****

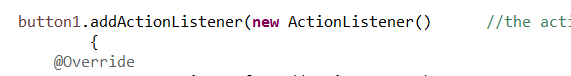
* **Event-Driven programming paradigm**:

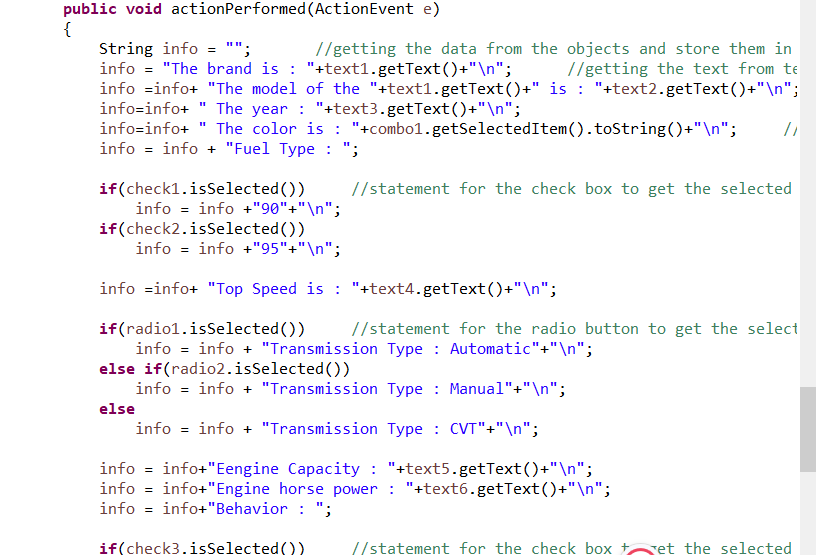
First, we imported the libraries that we want to create objects on the GUI interface, then we created the frame object from the frame class and make the design for it, then we added objects to the frame like text fields, labels, combo box, check box, button, and radio button, and we made the design for all the objects





**Then we created the action listener and the action to be performed when the event occurs**





**Task 2 part 2:**

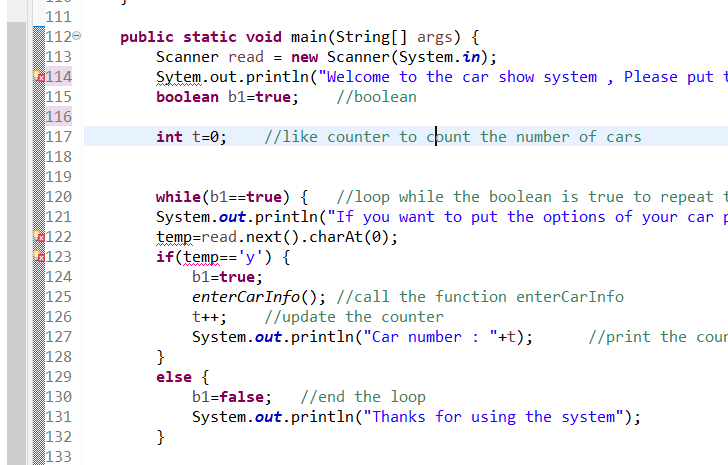
**2.2.1** The debugging process that I have followed in writing my system that to review my code carefully line by line to know where is the error, If it is a syntax error, it is easy to solve this error with this debugging process because the IDE tells us where the error occurred in any line, and tells us what is the error, If it is a logical error, it is bit harder to know the error with this debugging process, but how I use it? Before I execute the code, I write the expected result from my code on paper, Then I run the code and see the result, If it is the expected result, There is not a logical error in my code, If it is not as the expected result, I return back to the code line by line to find where is the error? If I don’t find it, I try to guess in any part is the code. Then I try to make it as a comment then execute the code to know if the error is in that part or not, When I find the logical error I modify it to give me the expected result.

If the error is a run time error, I determine what is the exception, then locate the error line and try to solve it to make the computer able to do the task.

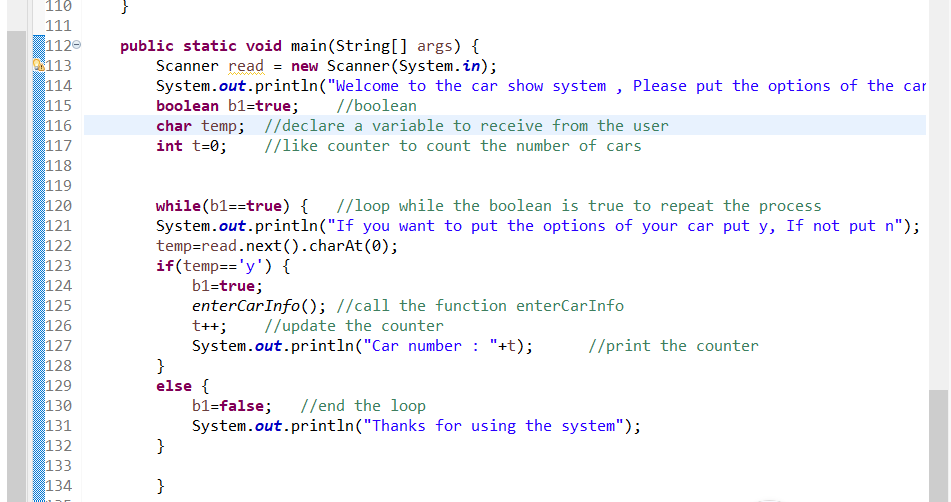
* **The debugging process that I have followed:**

1. **Check for the Syntax error and solve them.**
2. **Run the code and if there is any run time error, Determine what is the exception and locate the error line, Then I solve the error depending on the exception.**
3. **After there is no Syntax and run time error, I check the result of my code, If it is expected, My code is correct, If it is not expected, I review the code line by line, then I try to guess in any part the error is, When I find it, I solve it, Then compare the result again.**

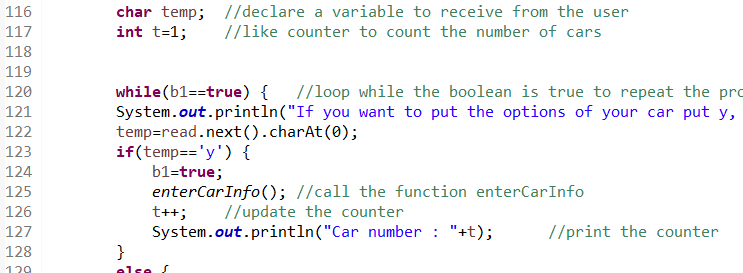
* **Syntax error and how I solved it**:

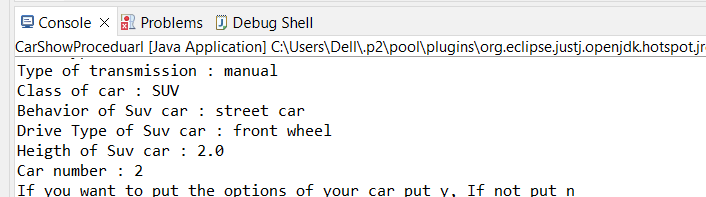


Here is my system, I faced two syntax errors, The first one we miss the letter ‘s’ in the word (System), and the second syntax error is that we didn’t declare the variable (temp). I added the letter ‘s’ to the word (system) and I declared the variable, Here is a screenshot after we solved the syntax errors, We notice that the error mark disappeared on line 114, line 122, and line123:

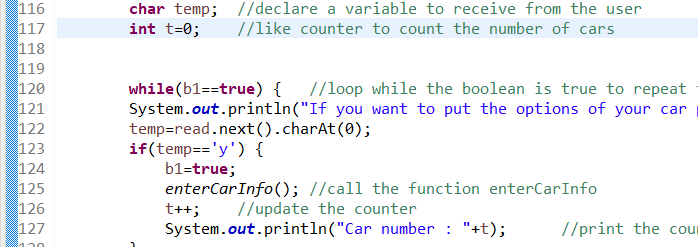


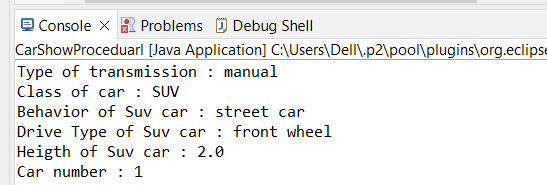
* **Logical error and how I solved it:**





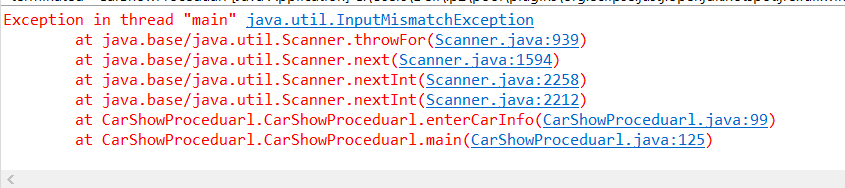
Here we have a logical error, The variable (t) counts the number of cars that the user is looking for when the user looked for car number one, the program prints the car as number two, Because we declared the variable as the starting value is one, And it is updated to two, So I have returned to that variable and declared the variable as the starting value is zero, And it is updated to one and so on, Here is a screenshot for the program after we solved the logical error:



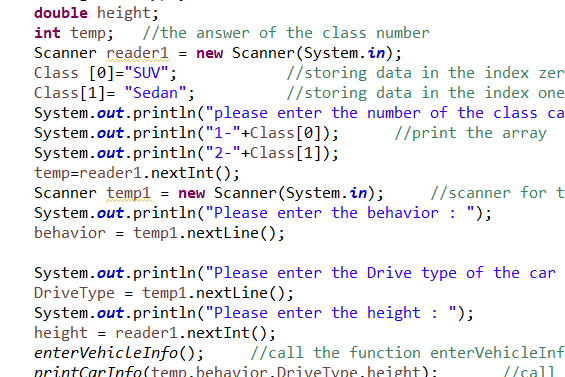


* **Run-Time error and how I solved it:**

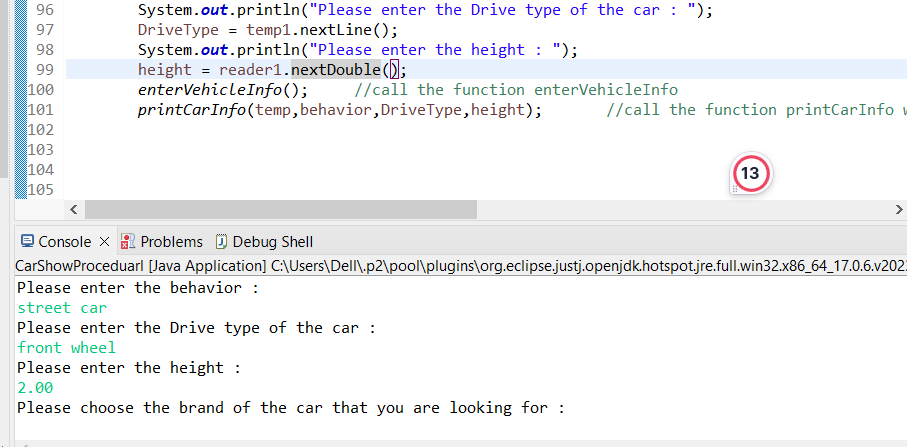
****

****

Here we have this run time error when the user entered the height, So we go back to the code where we made the user put the value of the height



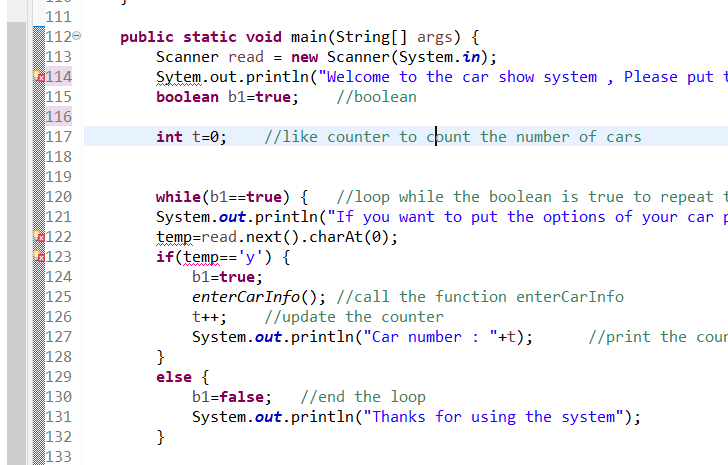
Here we found that the height is double, and the computer created to read an integer from the user, So I modified it to read a double, here is a screenshot after solving the problem:



**How we used the available facilities on IDE to solve each error:**

* **Syntax error:**

Actually, the IDE tells us where is the error located, and what is the error, So we can solve the syntax error easily without using any techniques to find the error and solve it.





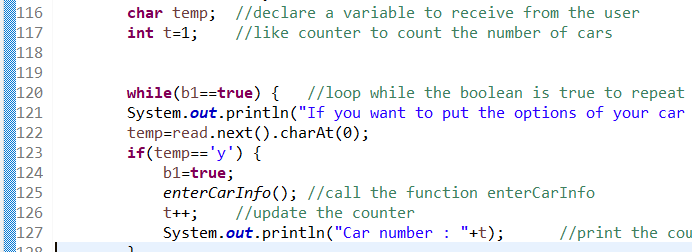
So we solve the problem manually.

* **Logical error:**

There are some facilities available on the IDE help us to find the logical error and solve it, One of these facilities is the breakpoint, We use the breakpoint to find the logical error and solve it, The principle of the breakpoint is to put a breakpoint on the line that you want to trace, then you debug the code not run it. Debugging the code means that the code runs until it arrives at the breakpoint, then the code stops execution, so we can see the execution of the code line by line **[1].**

**How we use the breakpoints to find the logical error in our system:**

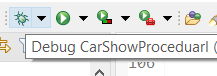
1. Here I want to trace line 126, to know why the system considers the first car as car number two not car number one



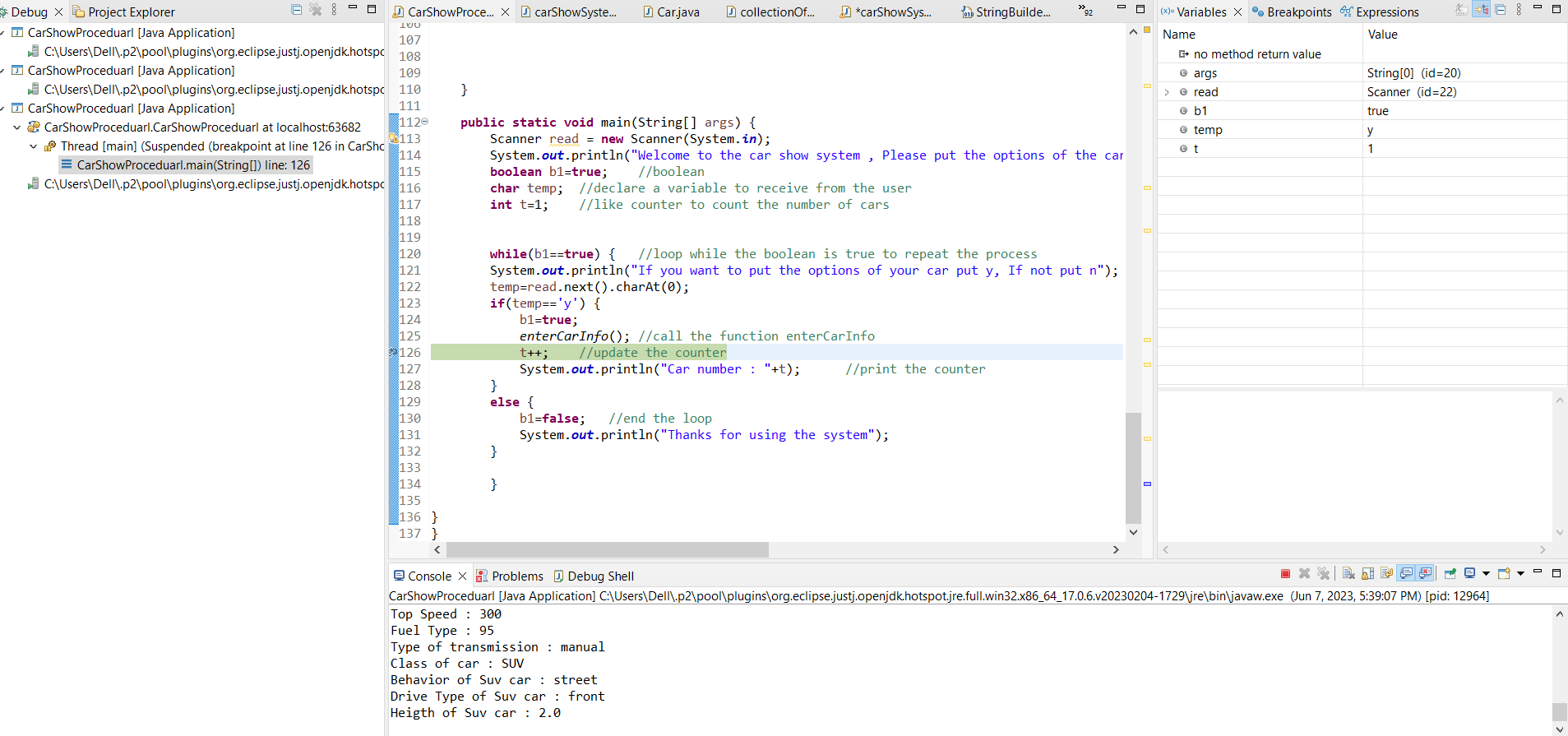
1. So we put a breakpoint on this line:



1. Then we debug the code:



1. The code will run normally until it arrives the breakpoint then the code stops running:



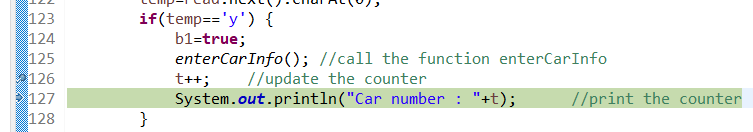
On the right we will find the value of the variable before it is updated, we find that is the value is one before it is updated and we need it zero not one before the updating:



We have some toggles that we have on the IDE like step over, To execute the selected line and goes to the next line:

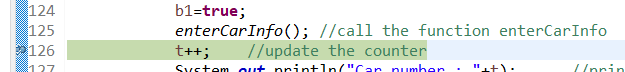


So we press this toggle to execute the line, to see what the value of the variable will be after the updating:





Here we find the logical error, So we will modify the starting value to be zero, not one.





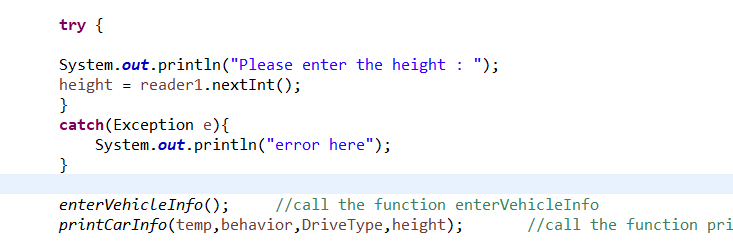


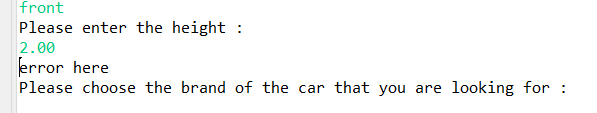


* **Run-time error**:

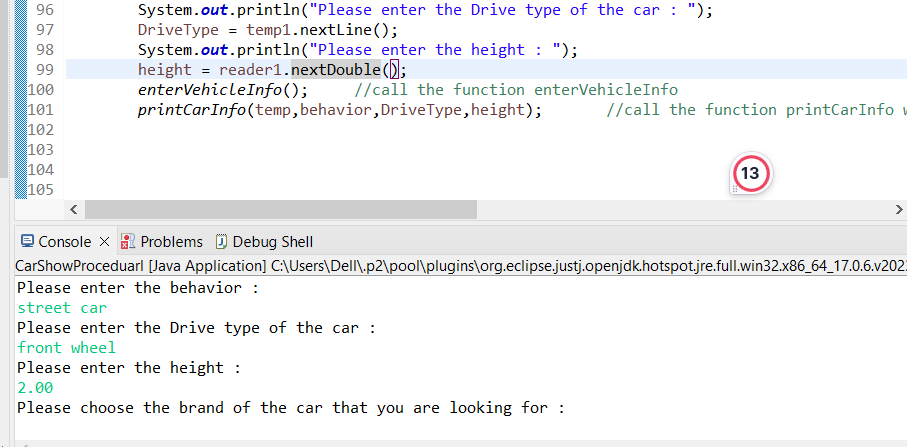


To find the run-time error, We will use the try/catch to find this error, we put the try on the block that we expect the error is from, The try will test this block, Then we put a catch with the type of the exception, if the try find the exception, and the exception in the catch matches with the exception that the try found, The catch will be executed





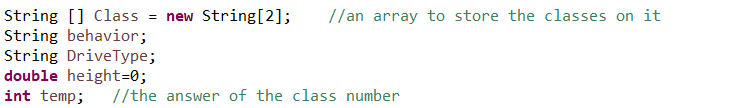
Here the error is found and the catch implemented, Then the code continued, So we will find the problem and solve it



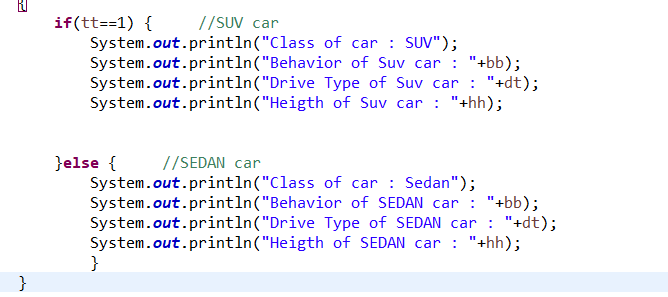
**2.2.2** The debugging process helps to develop more secure and robust applications through many things like exception handling when we use the try, we test a block of code, then when the error is found, the catch implements, then the rest of the code continues, That means we are building a code that is reliable and giving us a result even if there are errors or something challenging to execute the code, That is one of the scenarios how the debugging process helps us to develop a more robust application. When the code is more correct and has a less number of errors that means our code is more secure, The debugging process helps us to find the error and to solve it, So we are decreasing the number of errors, which means we are building a code with fewer errors to reach for an efficient code that is secure to use it in any scenario **[2].**

**2.2.3** The coding standards that I have used in my code:

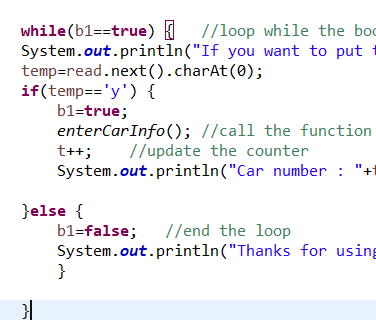
1. Declarations: in Java, it is preferred to declare one variable per line because it enhances the commenting, and not to declare two different types at the same line. I have declared every variable in one line as you can see in the screenshot:



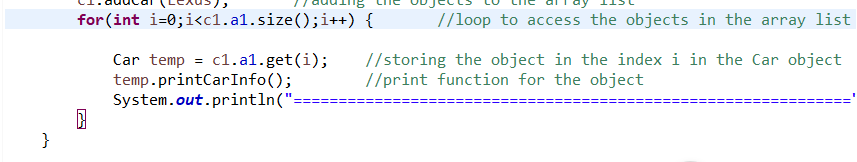
1. if, if-else, if else-if else Statements: In Java, There is a form for the if statement (braces/conditions/statements), I have followed this form in my code when I used the if statement as you can see in the screenshot:



1. While statement: in Java, There is a form for the while statement (braces/condition/statement), I have followed this form in my code when I used the while statement as you can see in the screenshot:



1. For statement: in Java, There is a form for the for statement (braces/initialization/condition/update/statement), I have followed this form in my code when I used the for statement as you can see in the screenshot:



1. Statements: In Java, we write every statement on a single line, I have used this standard in my code as you can see In the screenshot:





* The coding standards are important for the readability of the software, as there are many developers working on the same software so those standards make it easier for the new developer to understand the new code after the previous developer. The standards are also important for maintenance as it improves the process of maintenance, When someone wants to adjust the code, He can understand it and adjust it easily without needing to return to any developer who adjusted the software before, And the coding standards help the developers to write a less complex code because those standards are constant so they decrease the complexity and the errors of the software **[1] [3].**
* It is necessary to follow the coding standards individually to improve and encourage your skills to become a professional developer as you improve yourself by writing constant software, robust application with few errors and bugs, and high-quality software. Also, the coding standards are necessary for the team, as those standards make the code easier to understand, so the members of the team don’t need a lot of time to understand the code and adjust it especially if they are new members so the process of understanding the code becomes easier with constant coding standards **[4].**

**References list :**

**[1]: Lecture notes(slides).**

**[2]:** [How can the debugging process be used to help develop more secure, robust applications? - Quora](https://www.quora.com/How-can-the-debugging-process-be-used-to-help-develop-more-secure-robust-applications)

**[3]:** [Coding standards: what are they, and why do you need them (codacy.com)](https://blog.codacy.com/coding-standards-what-are-they-and-why-do-you-need-them/)

**[4]:** [Why coding standards matter to your team and business - Plesk](https://www.plesk.com/blog/tips-easy-reading/coding-standards-matter/)