

# Coursework II [CST-2550]

## *Contents:*

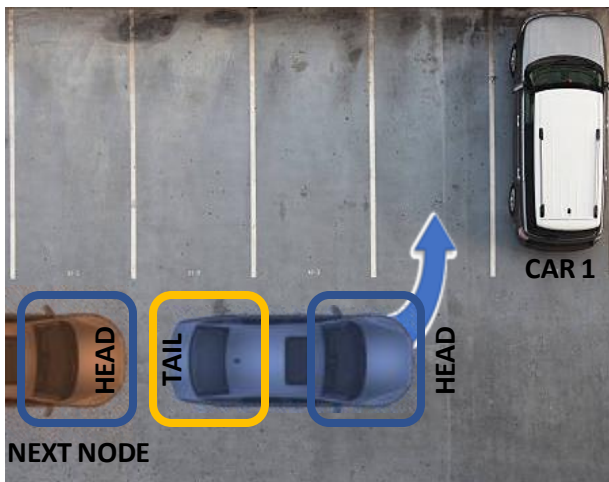
<b>Introduction</b> .....	1
<b>Design:</b> <i>Data Structure Selection</i> .....	1
<b>Data Structure:</b> <i>Analysis</i> .....	1
<b>Conclusion:</b> <i>Summary, Limitations &amp; Prospective ideas</i> .....	2
<b>References</b> .....	4

## ❖ Introduction

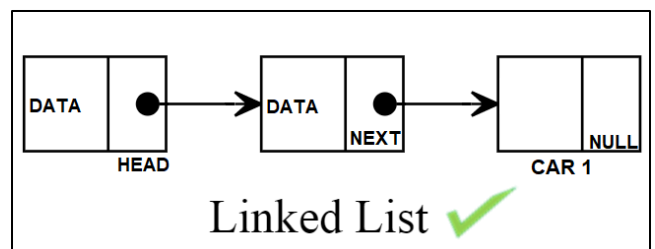
- Our main objective is to create a Car parking System which will take vehicle information such as Vehicle number plate, date, time, parking charge etc.
- To fulfill the task objective, we are required to select a specific type of data structure
- The **task objective** is to use a data structure to store information of vehicles such as cars and to also calculate the parking charge by further calculation of time difference of when the vehicle was entered and exited. The charge rates and vehicle information is provided, and we are to program accordingly to make sure we get the respective output.
- I have selected my data structure to be linked list. I will explain in this document of how I came to select the data structure starting in the Design justifying my reasons for my choice of selection. It will include analysis of the algorithm and what way we can best achieve the functionality.
- After that will come testing where I performed testing to check if the program performs as intended and to find out what flaws or errors it may have.
- Finally, I end it with the conclusion, where I discuss the critical reflection of what my program has and the limitations I had to go through while programming it. Further changes and updates to a similar task in the future will be enlisted after that.

## ❖ Design: Selection and Analysis

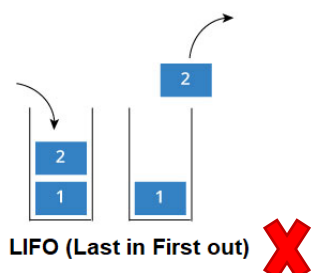
- Primarily made an illustration of the scenario to better depict what I had understood what the algorithm requires.



- From the depiction I understood the following:
  - Cars enter and exit parking lot in a line (like a queue)
  - Each car can be seen to have a head and tail to which it can connect to the next node



- In my initial analysis, it realized that there are many inconsistencies and using Stack (LIFO) or Queue (FIFO) as the data structure would not be ideal in this scenario.
- My justification is that if there would be any element that would want to “exit” the parking system, it would not be possible using LIFO as the only one to exit can be the last vehicle. Similarly for FIFO if any car in the middle or end would like to exit before the first car, it would be difficult to implement.
- Therefore, selecting Singly Linked list for this felt perfect as it obeys the flow of vehicles and allows them to exit in order.



## ❖ Analysis of algorithm

The requirements for achieving the objective of this task are listed below:

- The program for the parking system should be able to add items (Vehicles Number Plate information), collect information such as date and times of entry and exit.
- The program should be able to display all the vehicles or show the report of all the vehicles present in the parking system at their respective timings.
- The program should also be able to display the parking charge by calculating the difference in the exit and entry of the vehicle's parking time.

**Algorithm (Pseudo code) :**

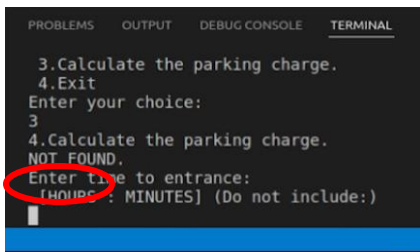
- Create display, insert, search and initialize other prerequisites (list.cpp)
- Create function to load read and write to any file specified by the user (in main.cpp)
- Create user interface for user to select options
- Create switch case to cycle through all the options
- Create constructors in node.h (for initializing nodes) and list.h (initializing the elements such as Date, Vehicle no.) header files

## ❖ Testing

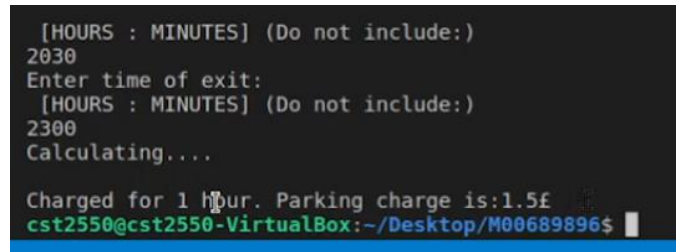
*Statement of testing approach used*

- *Validation Testing:* For validation testing we confirmed that the user is able to add a new vehicle and its data such as time and date, can calculate the parking charge by knowing the difference of times of exit and entry. It also has user validation implemented.
- *Defect Testing:* When entering the time when asked for the user, if ':' is entered with the numbers, it will not perform correctly therefore requiring the user to enter '13:30' as '1330'.
- *Unit Testing:* When testing each function and parts of the code, due to an unsuitable choice of data type certain parts don't work properly, one example would be the one given in the prior test.
- *Debugging:* The code has been through a large process of debugging before being finally being uploaded as the requirement for the coursework is to make sure the program is able to perform and achieve the task in the University lab environment (for e.g. should be able to run on Linux via terminal).

### • Test cases:



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
3.Calculate the parking charge.
4.Exit
Enter your choice:
3
4.Calculate the parking charge.
NOT FOUND.
Enter time to entrance:
[HOURS : MINUTES] (Do not include:)
```



```
[HOURS : MINUTES] (Do not include:)
2030
Enter time of exit:
[HOURS : MINUTES] (Do not include:)
2300
Calculating...

Charged for 1 hour. Parking charge is:1.5f
cst2550@cst2550-VirtualBox:~/Desktop/M00689896$
```

- When trying to calculate parking charge it will display "NOTFOUND" unintentionally
- Example of defect testing as described in detail above

## ❖ Conclusion

### *Summary of the work done*

- *For this scenario I chose linked list as the data structure to implement in my program. I revised all the information provided and started by creating an algorithm and then working about it. Soon I had to debug a lot of errors to make sure the program was performing as intended. Soon after initiated the testing phase and confirmed everything was performing properly.*

### *Limitations and critical reflection*

- *I should have better time management and implementation*
- *Would have resulted better if not working within a virtual machine due to technical issues*
- *Should have made accurate implementations rather than alternate solutions*

### *Change of approach on similar task in future*

- *There were a short number of mistakes made in the making of this program, some of which I wish to avoid in the future implementation of this program.*
- *Will try to reduce the length of the code and make better use of functions*
- *Will try to possibly consider other options such as different data structures to check if they are suitable for the requirements as the current selection was implemented not as perfectly.*
- *There were some mistakes such as bracket errors which costed with a lot of time, therefore to code more clearly and indent better.*

## ❖ References (Harvard format)

- *cjamesm (2012) 'Answer to "How to loop back to the beginning of a switch statement after a case"', Stack Overflow. Available at: <https://stackoverflow.com/a/13089413> (Accessed: 6 May 2022).*
- *How to read CSV file in C++? - Java2Blog (2021). Available at: <https://java2blog.com/read-csv-file-in-cpp/> (Accessed: 6 May 2022).*
- *Input/output with files - C++ Tutorials (no date). Available at: <https://www.cplusplus.com/doc/tutorial/files/> (Accessed: 20 April 2022).*
- *Read data from csv file and store it to linked list || Search data from linked list || in C++ - YouTube (no date). Available at: <https://www.youtube.com/> (Accessed: 6 May 2022).*