ROBOTIC ASSEMBLY LINE SIMULATOR

Name: Kaarthick S

Roll number: ME24B1056

Date:13/04/2025

Key Objective:

1. To design a system using data structures
2. To Showcase how the parts flow
3. To plant robots in a highly efficient manner

Design explanations:

1)Part delivery System: It defines the parts arriving via conveyor belt that means it must have a FIFO principle Thus, using queue is recommended.

2)Robotic arm Task manager: Since the robotic arm pick them up and arranges the part in a LIFO manner the data structure stack must be used to gain more efficiency.

3)Assembly storage unit: all of the prototypes are sent to a garage which is continuous in storing the data (here the vehicles) hence the usage of arrays is considered more optimal.

4)Repair and Upgrade tracker:

a) Defective prototypes: The defective prototypes are tracked by a singly linked list as their size is not predefined and every defective must be easily accessible.

b) Repaired prototypes: the repaired prototypes must be verified for multiple times hence TO and FRO communication within the prototypes is need Thus, using doubly linked is suitable.

c)High-priority prototypes: The highly prioritized orders must be completed as soon as possible hence connecting the ends makes them easier to revisit Thus, using circular linked is better.

How did selecting these data structure helped me solve the code more efficiently?

The data structures used for each property is selected so carefully that it gives the minimum possible number of iterations in the loops used in the code and simultaneously giving all the possible ease of accessibility with minimum possible number of lines in the code with proper indentations.

Variable used in the C-code (based on the data structures):

1. Queue:
2. Front – used for dequeuing
3. Rear – used for enqueuing
4. Parts [6][10] – used for storing the data
5. Stack:
   1. Top – used for pushing and popping the elements.
   2. Parts [6][10] - used for storing the data.
6. Arrays:
   1. Garage [8]-To store the prototypes contiguously
7. Single Linked List:
   1. Data - To store the data
   2. Next – A node pointer to store the next node’s address
8. Doubly Linked List:
   1. Data – To store the data
   2. Next – A node pointer to store the next node’s address
   3. Prev – A node pointer to store the previous node’s address
9. Circular Linked List: Similar to Single Linked List.

What does the Code do?

* At first the parts arrive via conveyor belt and gets stored in a queue
* The robotic arm picks the parts and arrange them in a stack
* Then the robotic arm assembles the prototype vehicle
* The prototype vehicle is then moved to the garage
* Once every prototype is moved to the garage the malfunctioning ones are identified and sent back to the factory
* Then the corresponding garage is emptied and new prototypes are added to it
* For the V.I.P clients the corresponding prototype are taken and upgraded based on their needs
* The prototypes and the supplied to multiple rounds of upgrade check for the V.I.Ps

Explanations of some of the functions used in the code:

1. Enqueue:

the element of the array corresponding to the rear variables value is updated to the input and the rear element is incremented.

1. Dequeue:

The element of the array corresponding to the front variables value is returned and then the front element is incremented

1. Push:

The top element is first incremented and then the given data in inserted in the array corresponding to the top element.

1. Pop:

The element of the array corresponding to the top variables value is returned and then the top element is decremented

Screenshot of the Output of the C-code:

