

Data structure used

- 1) **ArrayDeque** – gets floor size from the user also helps in processing floor size the order in which they received. 3->1->2 (FIFO Order)
- 2) **Stack**- acts as blueprint. It has floor size in expected order using which floors will be assembled Ex: 3->2->1 (LIFO order)
- 3) **Linked List**: Arranges floors using the blueprint and displaying floors on the particular day (Elements will be inserted at the front of the list and will be sorted using Merge sort technique)

Algorithm:

- 1) Store the given floors in an ArrayDeque .
- 2) Initialize stack with necessary floor size values to create blueprint.
- 3) Create empty linked list.
- 4) Traverse the queue and perform the following operations
 - i. Poll the first element in the queue and insert it at the front of the Linked List.
 - ii. Sort the Linked List using Merge sort implementation.
 - iii. Get the maximum floor size from the stack
 - iv. Compare the first element of the List with the top of the stack.
 - If both matches then print the element, pop it from the stack and delete it from the Linked List. Repeat this step until TOS does not match with the first element in the list.
 - If they do not match then print nothing for the day.

	Queue	Stack	Linked List	Comments	Output
Before processing	4->5->2->1->3 (5 is the first element)	5->4->3->2->1 (TOS=5)	null		
Day 1 (Size given : 4)	5->2->1->3 (Poll 4)	5->4->3->2->1 (TOS=5)	4->null (Insert 4 at the front and sort the list in descending order)	TOS(5) is not equal to head(4) Print nothing.	Day : 1
Day 2 (Size given : 5)	2->1->3 (Poll 5)	5->4->3->2->1 (TOS=5)	5->4->null (Insert 5 at the front and sort the list in descending order)	TOS(5) is equal to head(5) Pop – 5 from stack Delete - 5 from Linked List Print 5	Day: 2 5

	2->1->3	4->3->2->1 (TOS=4)	4->null	TOS(4) is equal to head(4) Pop – 4 from stack Delete – 4 from Linked List Print 4	Day: 2 5 4
Day 3 (Size given : 2)	1->3 (Poll 2)	3->2->1 (TOS=3)	2-> null (Insert 2 at the front and sort the list in descending order)	TOS(3) is not equal to head(2) Print nothing.	Day : 3
Day 4 (Size given : 1)	3 (Poll 1)	3->2->1 (TOS=3)	2-> 1->null (Insert 1 at the front and sort the list in descending order)	TOS(3) is not equal to head(1) Print nothing.	Day : 4
Day 5 (Size given : 3)	Empty (Poll 3)	3->2->1 (TOS=3)	3->2-> 1->null (Insert 3 at the front and sort the list in descending order)	TOS(3) is not equal to head(3) Print 3.	Day : 5 3
	Empty	2->1 (TOS=2)	2-> 1->null	TOS(2) is not equal to head(2) Print 2.	Day : 5 3 2
	Empty	1 (TOS=1)	1->null	TOS(1) is not equal to head(1) Print 1.	Day : 5 3 2 1