TASK PERFORMANCE: Written Part
Name: Jarilla, Angelo Christiene M.
Section: BSCS401
Subject: Design and Analysis of Algorithm

1) Linked list:
A linked list is a type of data structure in which pointers are used to connect its nodes. Every node has two components: the data and a link or reference to the node after it in the sequence. Linked Lists enable flexible memory allocation, which makes it simple to add or delete elements without effecting others, in contrast to arrays, where elements are kept in a continuous block of memory. However, since you have to go through the list from the beginning, looking for an element may take longer.

2) Queues:
Using Queue data can be managed and arranged. Like a line at a ticket counter, a queue operates on the First In, First Out (FIFO) principle, which states that the first element added is the first one deleted. In contrast, a stack adheres to the Last In, First Out (LIFO) principle, which states that when plates are stacked and unstacked, the last item added is the first one removed. A heap is a unique kind of tree-based data structure that is mostly used for effectively managing priorities, including rapidly determining which element is the smallest or largest.

3) Graphs:
A graph is made up of nodes, also known as vertices, which are groups of points connected by edges. Graphs can depict a wide range of relationships found in the real world, including computer networks, social networks, and road maps. They might be undirected, meaning that connections flow both ways, like a two-way street, or directed, meaning that connections have direction, like a one-way street.

4) Trees:
In Trees, there is a primary node which is called a root and, in those roots, there are branches. Trees are a unique kind of graph. Although a tree's nodes can have several children, there is only one path connecting any two of them. Trees are frequently found in organization charts, file systems, and search techniques like binary search trees.

EFFICIENCY TABLE:

Info:	Linkedlist	Queue	Graph	Trees
User time (seconds):	0.03	0.02	0.02	0.02
System time (seconds):	0.00	0.00	0.00	0.00
Percent of CPU this job got:	70%	69%	66%	66%
Elapsed (wall clock) time (h:mm:ss or m:ss):	0:00.04	0:00.05	0:00.05	0:00.04
Average shared text size (kbytes):	0	0	0	0

Average unshared data size (kbytes):	0	0	0	0
Average stack size (kbytes):	0	0	0	0
Average total size (kbytes):	0	0	0	0
Maximum resident set size (kbytes):	10280	10248	10232	10248
Average resident set size (kbytes):	0	0	0	0
Major (requiring I/O) page faults:	0	0	0	0
Minor (reclaiming a frame) page faults:	1125	1120	1120	1135
Voluntary context switches:	40	41	40	41
Involuntary context switches:	0	0	0	0
Swaps:	0	0	0	0
File system inputs:	0	0	0	0
File system outputs:	0	0	0	0
Socket messages sent:	0	0	0	0
Socket messages received:	4096	4096	4096	4096
Signals delivered:	0	0	0	0
Page size (bytes):				
Exit status:	4.0K	4.0K	4.0K	4.0K
File Size (Disk usage of the file):	0.03	0.02	0.02	0.02

ALGORITHM USE CASE

	Linkedlist	**Queue**	**Graph**	**Trees**
Best-fit Use-case	Dynamic memory allocation, frequent insertions/deletions	FIFO processes, task scheduling	Modeling relationships, network problems	Hierarchical data, efficient searching
Example Use-cases	Undo/redo operations, real-time systems	Task scheduling, BFS in graph traversal, request handling	Social networks, GPS routing, web crawlers	File systems, binary search trees, decision trees in machine learning

EXECUTION SPEED:

	LinkedList	**Queue**	**Graph**	**Trees**
real	0m0.056s	0m0.050s	0m0.053s	0m0.056s
user	0m0.013s	0m0.028s	0m0.027s	0m0.019s
sys	0m0.025s	0m0.009s	0m0.010s	0m0.021s