Student Id 20210741

Question1: List – Search and Delete Records

Central Hospital would like to have a program implemented to keep track of the hospital patients. It should have a record of the current hospital patients, allow the search of patients by patient ID and have an option to delete a patient's record after the patient is released from the hospital.

Analyse the data structure used in the task. Was the data structure fit for the task? Explain your answer.
The List was a great data structure for this task, especially with the built in functions that allow you to search and manipulate the List elements with ease.

Advantages:

Contains built in functions such as Find() to search through elements to find a match.

dynamic size:
the List can grow in size as needed doubling its memory capacity each time it is exceeded. For example, capacity will increment exponentially 2, 4, 8, 16 and so on when the count of items in the list exceeds the current capacity as a new item is being added.

Can be indexed.

Disadvantages:

Occupies more memory than a list.

Slightly slower to use than arrays as they still use arrays under the hood and still needs to make an entire copy of the array when the list changes in size.

Is there another data structure that could be used in place of array list to complete the task?
A Linked List could achieve this task very effectively. Especially on large sets of patients where list elements often need to be removed. The use of the inbuilt functions Contains(), Add() and Remove() would make this easy to manage.

Using Arrays could be done for this task but it would be more prone to bugs as you would have to build in your own functions to search through the array and manipulate the data, this would also be time more time consuming to set up.

Question 2: Stack – Validate an expression

Create a program that will help to validate an expression to see if it has the correct number of matching brackets. It should ask the user to enter an expression to validate and the type of bracket characters to match.

Analyse the data structure used in the task. Was the data structure fit for the task? Explain your answer.
The Stack works exceptionally well for this task, once I had solved the problem I was thinking to myself that I would have trouble imagining a more suitable data structure for the exercise. The way that the

stack keeps all the elements in LIFO order is perfect for checking that the Brackets are in the correct order as well as evenly balanced.

Advantages:
very simple implementation and effective for executing the task with built in functionality provides ease of coding.

LIFO (Last in first out) is perfect for checking placement and balance (keeps data sorted).

Fast to execute.

Capacity automatically increases and shrinks as elements are added.

Keeps all elements in order.

Disadvantages:

Limited use for extension due to lack of ability to index or search through stack.

Is there another data structure that could be used in place of Stack to complete the task?
Although I'm sure you could design an alternative way to solve this problem, to provide the functionality within another data structure that the stack inherently carries would be a time-consuming exercise that would be prone to bugs.

Question 3: Queue Scenario

Analyse the data structure used in the task. Was the data structure fit for the task? Explain your answer.
Create a program that simulates queueing in a shop. Customers are seen in the order of their arrival to the shop. Sales Assistant is seeing a customer every 5 seconds, a new customer arrives to the shop every 3 seconds.

Advantages:
Fifo (First in First Out) keeps the order of customers sorted naturally by priority.

Good built in functionality for the limited scope of this project.

Disadvantages:

Difficult to extend due to lack of index ability.

Is there another data structure that could be used in place of Queue to complete the task?
A doubly Linked List could be used for this task. Although it would likely make a it little more complex, this would allow for extension by providing a way to access Customers who aren't necessarily at the front of the line to be manipulated.