1. Usestaticarray.c
   1. Submission: Give it a try and submit screenshot

A screenshot of a computer program

AI-generated content may be incorrect.

1. mallocprinter.c
   1. Understand how malloc works but nothing to do or submit
2. Dynamically-allocating.c
   1. Modify code and explain approach and why it solves the problem

I approached this by thinking about how to let the user keep entering numbers without worrying about running out of space. What I did first was set aside some memory for a fixed number of integers. I then used a while loop to keep asking the user for numbers. When the allocated space was full, I used a temporary variable called temp to create a new block of memory with 10 extra spots, and then I updated the main pointer to point to this new memory. This way, even if something went wrong, the original memory would still be safe. I remembered this method from my CSCI 1933 Data Structures and Algorithms class, where we learned about resizing arrays in Java. Doing this works because if you wanted to resize the array you could copy the original data from the array and put it into the temporary array and then assign the temporary array data to the original array variables.

1. Simple-dynamic-example.c
   1. Screenshot of program running for submission

A computer screen with white text

AI-generated content may be incorrect.

1. Python-lists-are-objects-not-arrays.py
   1. Define what an object is in terms of the Object-Oriented Programming paradigm.

In object-oriented programming, an object is an instance of a class that bundles data and behavior into one entity. For example, a Python list is an object that not only holds a collection of elements but also provides methods to manipulate that collection—such as appending, removing, and sorting items. This encapsulation of state and behavior makes Python lists a perfect embodiment of OOP principles, much like how CoPilot understands code by recognizing the structure and interactions of objects.

1. Linkedlist.c
   1. Write a short paragraph explaining how does using a linked list solve some of the same problems were are trying to overcome when using dynamic memory for submission

When you use arrays, memory is set in one block and if you need to resize it, the program must find a new spot for the data and copy it over, which can take time and cause problems, especially with large data. On the other hand, a LinkedList handles memory differently. It stores data in individual nodes where each node points to the next one. Because of this memory doesn’t need to be moved or copied when data is added or removed. The program just updates the pointers between the nodes. This way, adding, deleting or resizing data doesn’t involve moving large chunks of memory making it a more efficient solution for dynamic memory management.

1. Managing-allocation-with-struct.c
   1. Nothing to submit, just review code