SIT102 Introduction to Programming

Distinction Task 9.4: Linked List

Overview

Linked lists are a dynamic data structure used to store multiple values, in a similar way that arrays and vectors also store multiple values. Implementing your own linked list is a great way of exploring the use of pointers and memory management. In this task you will create a custom linked list using pointers and memory management functions.

Submission Details

Download the starter code and watch the Linked List Starter video to get started with this task. Implement the code shown, then make the following changes.

1. Implement an `add_to_start` procedure that will add a value to the start of the linked list.

   ```
   void add_to_start(linked_list &list, int value)
   {
       //...
   }
   ```

   Test this in `main` by adding three values to the start of the list.

2. Implement a `reverse_print_all` procedure that will print out all of the elements in the list in reverse (last to first).

   ```
   void reverse_print_all(const linked_list &list)
   {
       //...
   }
   ```

   Add code to test this in `main`.

3. Implement an `insert_after` procedure that will add a new node after a selected node:

   ```
   void insert_after(node selected_node, int value)
   {
       //...
   }
   ```

   Test this in main and make sure you can add to the middle or end of the list.

4. Implement a `reverse_for_each` that accepts a `visitor` and executes it for each element in the list, starting at the last element and moving backwards through the list.

5. Implement a `reverse_print_all_with_visitor` that will use your `reverse_for_each` and the `print_node` to print the list in reverse.

   Add a test for this in `main`.

6. Use a lambda expression and your `reverse_for_each` to implement a `length` function that will calculate the length of the list:

   ```
   int length(const linked_list &list)
   {
       //...
   }
   ```

   Add a test to main to make sure this also works correctly.

Once you have this all working, grab a screenshot of it in action and submit to OnTrack.