Data Structures and Algorithms

Lab 2 on Lists, Stacks, and Queues in C++

ENSIA 2023-2024

Objectives

- Familiarize with STL (C++ Standard Template Library)
- Understand Vector and List implementation
- Implement basic data structures: List, Stack, and Queue
- Implement operations on data structures: insert, delete, search, etc.
- Calculate the complexity of the different operations on data structures
- Select the appropriate data structure for a given problem

Prerequisites

C++ Classes (1.4), C++ Details (1.5), and Template (1.6) from the course textbook[1].

Exercise 1

Let's suppose you are given a list L, and another list P containing integers sorted in ascending order. The procedure `printLots(L,P)` prints the elements in L that are in the positions specified by P. For example, if $P = \{1, 3, 4, 6\}$ the elements in positions 1, 3, 4, and 6 in L are printed.

1. Write the procedure `printLots(L,P)`. You may only use the public STL list operations.
2. What is the running time of your procedure?

Exercise 2

Let's assume that a singly linked list is implemented with a header node, but no tail node, and that it maintains only a pointer to the header node.

1. Write a class that includes methods to:

[1] Data Structures and Algorithm Analysis in C++, Fourth Edition, Mark Allen Weiss

(a) return the size of the linked list

(b) print the linked list

(c) test if a value x is contained in the linked list

(d) add a value x at the beginning of the list if it is not already contained in the linked list

(e) add a value x at the end of the list if it is not already contained in the linked list

(f) remove the first occurrence of a value x if x is contained in the linked list

2. What is the running time of each method?

Exercise 3

Swap two adjacent elements of a linked list by adjusting only the links (and not the data) using:

1. Singly linked lists
2. Doubly linked lists

Exercise 4

Given two sorted lists $L1$ and $L2$, write a procedure to compute $L1 \cap L2$ using only basic list operations. What is the running time of your procedure?

Exercise 5

Add `insert` and `erase` operations to the class `Vector`. The header `Vector.h` is provided in the page 86 of the course textbook[1]. Here is the prototype of the two functions:

- `iterator insert(iterator pos, const Object &x)` inserts the element x at the position *pos* in the list and returns an iterator to the inserted element.
- `iterator erase(iterator pos)` deletes the element at the position *pos* and returns an iterator to the next position.

Exercise 6

Given a string `str`, write a function to check whether the pairs and the orders of the two brackets (,), [,] are correct. For example, the following expressions:

- [()][()()]() is correct
- [(]) and (() are not correct

Exercise 7

Efficiently implement a *Stack* class using a singly linked list, with no header or tail nodes.

Exercise 8

Efficiently implement a *Queue* class using a singly linked list, with no header or tail nodes.