# CS2710 : PDS Lab Tutorial Questions

Tutorial/Prep 6

Sep. 4, 2024

## Information

---

- This document comprises tutorial questions for CS2700, which includes both conceptual/theory questions (relevant for CS2700) and programming questions (relevant for both CS2700 and CS2710; these questions can also be thought of as preparatory/practice programming questions for CS2710 Lab 6, and so also referred to as Prep 6).
- Try to solve programming problems using array/vector and strings only. (Don't use any other data structure). You can use built-in sort() if asked/required.

---

### Conceptual questions (for CS2700)

1. You are to write a **Count()** function that counts the number of times a given integer occurs in a linked list of integers. How will you code the loop in this function below using a *while* loop, and how will you code it using a *for* loop?

```
int Count(Node* head, int searchFor) {
 int count = 0;
 Node* current = head;

 ----------------
 ----------------
 ----------------

 return count:
}
```

2. What is the functionality of the following piece of code?

```
int List::func() {
  Node* temp = tail->prev;
  tail->prev = temp->prev;
  temp->prev->next = tail;
  size--;
  return temp->item;
}
```

3. Consider three stacks with elements as positive numbers. You are given the sum of elements in each stack. Provide an algorithm to find the maximum possible number k such that the sum of the elements of all the three stacks can be reduced to k.
   *Note:* A stack's sum can be reduced (only) by removing some (possibly none or all) of its top elements.

4. Consider an implementation of a stack, where popping from an empty stack causes a "Stack is empty" warning message to be printed. Suppose we start with an initially empty stack S, and perform a total of 25 push operations, 12 top operations, and 10 pop operations (in arbitrary order). 3 of the pop operations generated the warning mentioned above. What is the current size of S?

5. Write a pseudocode to multiply two polynomials (with each sorted by the exponents of their literals) using a linked list. If the polynomials have M and N terms, respectively, what is the time complexity of your program?
   (Hint: $O(M^2N^2)$, $O(M^2N)$, and $O(MN\log(MN))$ running time algorithms are possible, where $M \leq N$.)

6. Recursion is a classic solution to the "$N$-Queens" problem, where $N$ queens need to be placed in a $N \times N$ chess board such that no two queens attack each other. You happened to be stuck in a world where recursion is considered an entangling sin and hence utterly prohibited. How will you code $N$-Queens solution given this predicament? You are allowed and even encouraged to maintain a stack in this world. What would you store in this stack to solve the $N$-queens problem? Trace the contents of your stack as your code finds a solution for $N = 4$.
   (Hint: A stack of integers suffices to solve this problem.)

## Programming questions (for CS2710 Lab 6 preparation/practice, and for CS2700)

7. Write a program that uses stacks to check whether an input expression of parantheses of different types is balanced (e.g., "()([])" is balanced, whereas "([)]" is not).

8. Write a C++ class that implements all basic operations of a circular (singly) linked list. Your program should then read a series of operations input by the user and print the contents of the list after each operation (e.g., "insert(5), insert(6), insert(70), remove(6)", etc.).

9. Write a program to add two polynomials (with each sorted by the exponents of their literals) using a linked list. If the polynomials have M and N terms, respectively, what is the time complexity of your program?

10. The stock span problem in financial analysis helps figure out profitable stocks (e.g., stocks on a 52-day high). For a stock, we have a series of $N$ daily price quotes stored in an array $X$ and we need to calculate the span of the stock's price for all $N$ days. Specifically, the span $S[i]$ of $X[i]$ is the maximum number of consecutive elements $X[j]$ immediately preceding $X[i]$ such that $X[j] \leq X[i]$. Can you figure out a solution that uses stacks to compute all $N$ spans in linear time?