Theory Problems 2

Linear Data Structures

Problem 1: Matching Parenthesis

A common problem for compilers and text editors is to determine whether a set of parenthesis is balanced and properly nested. For example, the string (((x))(y))(z) is properly formed, but)(x)(and (y)) are not.

a) Write an algorithm that returns true is a string contains properly nested and balanced parenthesis and false otherwise. The string may contain other characters which should be ignored. Optionally, the algorithm can report an error message with the position of the offending parenthesis. Your algorithm should use a constant amount of memory, regardless of the length of the string.

b) Write another algorithm for the same problem, but this time the algorithm may use O(n) memory but must use a stack

Problem 2: Palindromes

A palindrome is a string which is the same forwards and in reverse, e.g. ‘rotator’. Write an algorithm to test if a given string is a palindrome.

Problem 3: Dictionary

Using one of the ADTs discussed in the lecture, implement the dictionary data structure in pseudocode or Java.

What is the number of steps in O() notation to execute the following operations?

- Add 100 items to the dictionary and carry out *n* searches
- Add *n* items to the dictionary and carry out 100 searches

Dictionary Data Structure

Organization

Elements are pair (*k*,*v*) of key and value
Random access by key value
Unordered

Operations

`Insert(k,v)` key-value pair **`(k,v)`** added to dictionary. If **`k`** exists the value is replaced.

`Delete(k)` remove key-value pair **`(k,v)`** from the dictionary

`Search(k)` return the value **`v`** corresponding to **`k`**

Problem 4: Hashed dictionary

A hash function is a function which maps data onto fixed-size values (e.g. numbers in a fixed range). Consider a hash function Hash(x: T) which takes data of type T and produces an integer in the range 0-255. This function takes O(1) time.

How could you use this hash function to implement a faster dictionary?

A collision is when the hash function gives the same value for two different input values, i.e. Hash(x)==Hash(y), x≠y. Explain what problem this causes for your dictionary and propose a solution.