Tutorial 2: Lists and Hashing

Please note that the questions on hashing will need to be delayed for a later tutorial. It is given here so that you have the chance to look at it in time for the next ADS Thread tutorial.

(1) (a) Give pseudo-code for an algorithm that reverses a singly linked list. The algorithm should run in \(\Theta(n)\) time for any input that has \(n\) items with the exception of the empty list when it runs in time \(\Theta(1)\).

(2) (a) Consider the expression

\[
 a \ b - c \times d \ a + /
\]

given in reverse Polish notation (i.e., we write the arguments of a function first and then the name of the function, so we write \(3 4 \times \) rather than \(3 - 4\)). This has the advantage that there is no need for brackets. Use a stack to produce an equivalent form of this expression in standard bracketed form.

(3) Let \(H\) be a bucket-array hash table with hash function \(h(n) = (2n + 5) \mod 11\). Draw the contents of \(H\) after inserting the following keys:

\[
\]

(4) This question is a little harder than the others but does represent a realistic scenario in the use of hashing. In fact the solutions are quite straightforward.

Suppose we wish to maintain a dictionary of everyday English words using a hash table, with collisions resolved by chaining. One possible hash function is

\[
 h(w) = value(w) \mod m,
\]

where \(m\) is the size of the hash table and \(value(w)\) is the integer value of the character string \(w\) interpreted as a radix-128 integer, i.e., if \(w = ws_0s_1\ldots s_{n-1}\), then

\[
 value(w) = \sum_{k=0}^{n-1} ord(s_i) \times 128^{n-1-k},
\]

where \(ord(s_i)\) is the ASCII value of \(s_i\).

(a) For any word \(w\), the hash value \(h(w)\) is less than the table size \(m\) and so can be represented using the same number of bits \(b\), say) as \(m\) itself. The quantity \(value(w)\), however, will typically be so large as to cause integer overflow, even for short words \(w\). Suggest a way of computing \(h\) which, for any input string \(w\), requires as storage only \(b\)-bit integers, where \(b\) exceeds \(b\) by only a small additive constant.

(b) If a table size of around 50,000 were required, we might choose \(m = 2^b\), which is convenient from the point of view of performing ‘mod’ operations. Explain why this is a poor choice in the current application, and suggest an alternative which is likely to improve the performance of the hash function.

\[
inf2b\ algorithms\ and\ data\ structures\ informatics\ 2b\ 2023\ ch1\ section2
\]

\[
inf2b\ algorithms\ and\ data\ structures\ informatics\ 2b\ 2023\ ch1\ section2
\]