Hints on calculator

Evan M Drake
drakeev@butte.edu

20/11/2025

1 What is this

This sheet is designed to give you a couple of different perspectives for each sub-problem of the calculator assignment. Remember that these hints are designed to guide your thoughts on the problem and not solve the problem, thus on their own they should not be that helpful. Here I will break the calculator into several sub-problems that we can work through individually. First is stack memory, this is the process of using a stack as your main source of memory. Second is inference, this is the process of logically narrowing possibilities until an obvious answer occurs. Third is recursive processes vs iterative ones.

2 Stack memory

Stacks are incredibly useful due to their simple structure and lack of operations. A stack is nothing but a one dimensional data structure which can be filled or emptied from a single direction. That is one can think of a stack in terms of a stack of books, one can only place a book atop the stack or remove a book from the top else the stack topples. Given this simplicity it can be unintuitive how a stack is utilized for processes requiring more than one element at a time. The answer is of course to organize the stack such that the appearance of some element ϵ_1 implies the next element is some ϵ_2. In practice this is done using a sort of method of inference (to be discussed below). Once we have this organized stack where we can approximate the order of elements, we simply need to preserve that order while we work. That is if your stack presence in a simple repeated order such as $\{\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_1, \epsilon_2, \epsilon_3, ...\}$. The most difficult part of working with that stack can be preserving the order of those elements. It is worth thinking about in depth how to preserve this order.

3 Inference

Inference is the art of logically describe a mechanism or sequence with such detail that one can predict (with certainty) what the next element of a sequence is. When dealing with stacks this process is imperative. It is worth drawing a picture for each possible state of a mechanism in order to grantee ones inference rules. For the example of a stack above if one pops ϵ_1 from the stack (given one has preserved order) one can be certain ϵ_2 is now at the top.

4 Recursion vs Iteration

It is worth noting the choice is almost always preference (in computer science). For instance I prefer recursion because it keeps a stack of context on every recursive step. There are those that prefer iteration b/c it implies that stack of context does not exist and thus can not over flow. It is not always a matter of better or worse but rather making that choice. The calculator can be solved using recursion in a somewhat elegant method by defining a choice method and then calling operator dependant methods. The calculator can be solved using iteration by poping the stack repeatedly and analyzing the resulting collection of results. The difference between these methods is an understanding that using either you must still have the same concept of correct inference of your stack elements.