Implementing QuickSort with a Stack

CSE 326: Data Structures

October 1, 2008

Section 3.3 of the textbook gives a very general, but not very simple, way of replacing recursion by use of a stack. Here is a simpler example. Consider the recursive version of QuickSort:

```
procedure QuickSort(int low, int high)
{
    if low <= high
    {
        int pivotLocation = partition(low, high);
        QuickSort(low, pivotLocation-1);
        QuickSort(pivotLocation+1, high);
    }
}
```

The details of the function `partition` aren't important for our purposes. All you need to know is that it rearranges the keys and returns an index i with the property that all the keys at positions `low`, ..., $i-1$ are less than or equal to the key at position i, and all the keys at positions $i+1$, ..., `high` are greater than or equal to the key at position i.

Here is a simple version of this procedure using a stack `S` to implement the recursion. The entries on the stack will be pairs of integers.

```
procedure QuickSort(int low, int high)
{
    int pivotLocation;
    push(<0,-1>, S);  // a marker to identify the bottom of the stack
    while !IsEmptyStack(S)
    {
        while low <= high
        {
            pivotLocation = partition(low, high);
            Push(<pivotLocation+1,high>, S);  // record second recursive call
            high = pivotLocation-1;           // execute first recursive call
        }
        <low,high> = Pop(S);  // fetch next recursive call to execute
    }
}
```