List and Iterator ADTs

Khalid Hourani

February 23, 2017

1 Reversing an Array

Reversing an array of n elements involves just swapping elements at certain indices; the element at index 0 must be swapped with the element at index $n-1$, index 1 with index $n-2$, and so on. If n is even, we swap indices i and $n-1-i$ until we reach index $\frac{n}{2}$. If n is odd, we do the same until we reach index $\frac{n-1}{2}$. Since $\lfloor \frac{n}{2} \rfloor$ is given by

$$\left\lfloor \frac{n}{2} \right\rfloor = \begin{cases} \frac{n}{2} & n \text{ is even} \\ \frac{n-1}{2} & n \text{ is odd} \end{cases}$$

we only need to iterate from index 0 to $\lfloor \frac{n}{2} \rfloor$, regardless of the parity of n. Luckily, $\lfloor \frac{n}{2} \rfloor$ is the same as integer division of n by 2. Thus, we have the following implementation in C++:

```
void reverse(int* list, int n)
{
        for (int i = 0; i < n / 2; i++)
        {
            swap(list[i], list[n - 1 - i]);
        }
}
```

2 Randomly Permuting and Array

To randomly permute an array, we use the **Fisher-Yates Shuffle** to permute a 0-indexed array:

1. Iterate through the array, letting i represent the current index. Begin at 0 and end at $n-2$.
2. Choose a random integer j such that $i \leq j \leq n-1$.
3. Swap the i^{th} and j^{th} elements of the array.

In C++, this looks like:

```
void permute(int* list, int n)
{
        for (int i = 0; i < n - 1; i++)
        {
                int j = randint(i, n - 1); //choose a random integer between i and n - 1
                swap(list[i], list[j]);
        }
}
```

3 Circularly Rotating an Array

To rotate an array of n elements circularly by d units, swap the elements at indices 0 and d, then the elements at indices 0 and $2d$, then 0 and $3d$, and so on, noting that the element at index $kd > n$ has index $kd \bmod n$. Repeat until you've performed n swaps. In C++, this looks like:

```
void rotate(int* list, int n, int d)
{
        d = d % n; //if d < 0 or d >= n, rotating by d is the same as rotating by d mod n
        if (d != 0) //rotating by 0 makes no change to the array
        {
                for (int i = 1; i < n + 1, i++)
                {
                        int j = (i * d) % n; //index d, 2d, ...
                        swap(list[0], list[j]);
                }
        }
}
```