Quiz 5: Recursive runtimes

CS 14 - Data Structures

April 9, 2013

Questions:

1. How many times do we call the addition operator + in the following algorithm, in terms of the size of the input n?

```
int fib(int n)
{
  if (n==0) return 1;
  if (n==1) return 1;
  return fib(n-1)+fib(n-2);
}
```

   Find a more efficient version of this algorithm that uses time $O(n)$.

2. What is the run time of this function, in terms of the size of the exponent n?

```
int pow(int base, int exponenent)
{
  return powHelper(base, exponenet, 1);
}

int powHelper(int base, int exponent, int accumulator)
{
  if(exponent == 0) {
    return accumulator;
  } else if(exponent % 2 == 0) {
    return fastExp(base * base, exponent/2, accumulator);
  } else {
    return fastExp(base, exponent-1, base * accumulator);
  }
}
```

3. What is the run time of this sorting function, in terms of the size of the array n?

```
/* A is the array
   n is the size of the array
   i is the current index
*/
void sort(int A[], int n)
{
  sortHelper(A, 0, n);
}

void sortHelper(int A[], int i, int n)
{
  int j, small, temp;
  /* basis is when i = n-1, in which case */
  /* the function returns without changing A */
  /* induction follows */
  if (i < n-1) {
    small = i;
    for (j = i+1; j < n; j++) {
      if (A[j] < A[small]) {
        small = j;
      }
    }
    temp = A[small];
    A[small] = A[i];
    A[i] = temp;
    sort(A, i+1, n);
  }
}
```