CS 61B Fall 2021

More Asymptotics

Exam Prep Discussion 8: October 11, 2021

1 Asymptotics is Fun!

(a) Using the function `g` defined below, what is the runtime of the following function calls? Write each answer in terms of `N`.

```
1  void g(int N, int x) {
2      if (N == 0) {
3          return;
4      }
5      for (int i = 1; i <= x; i++) {
6          g(N - 1, i);
7      }
8  }
```

`g(N, 1)`: $\Theta(\quad)$
`g(N, 2)`: $\Theta(\quad)$

(b) Suppose we change line 6 to `g(N - 1, x)` and change the stopping condition in the for loop to `i <= f(x)` where `f` returns a random number between 1 and `x`, inclusive. For the following function calls, find the tightest Ω and big O bounds.

```
1  void g(int N, int x) {
2      if (N == 0) {
3          return;
4      }
5      for (int i = 1; i <= f(x); i++) {
6          g(N - 1, x);
7      }
8  }
```

`g(N, 2)`: $\Omega(\quad)$, $O(\quad)$
`g(N, N)`: $\Omega(\quad)$, $O(\quad)$

2 Flip Flop

Suppose we have the `flip` function as defined below. Assume the method `unknown` returns a random `integer` between 1 and `N`, exclusive, and runs in constant time. For each definition of the `flop` method below, give the best and worst case runtime of `flip` in $\Theta(.)$ notation as a function of `N`.

```
1  public static void flip(int N) {
2      if (N <= 100) {
3          return;
4      }
5      int stop = unknown(N);
6      for (int i = 1; i < N; i++) {
7          if (i == stop) {
8              flop(i, N);
9              return;
10         }
11     }
12 }
```

(a)
```
public static void flop(int i, int N) {
    flip(N - i);
}
```

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

(b)
```
public static void flop(int i, int N) {
    int minimum = Math.min(i, N - i);
    flip(minimum);
    flip(minimum);
}
```

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

(c)
```
public static void flop(int i, int N) {
    flip(i);
    flip(N - i);
}
```

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

3 Prime Factors

Determine the best and worst case runtime of `prime_factors` in $\Theta(.)$ notation as a function of `N`.

```
1  int prime_factors(int N) {
2      int factor = 2;
3      int count = 0;
4      while (factor * factor <= N) {
5          while (N % factor == 0) {
6              System.out.println(factor);
7              count += 1;
8              N = N / factor;
9          }
10         factor += 1;
11     }
12     return count;
13 }
```

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$