University of Nevada, Las Vegas Computer Science 477/677 Spring 2020

Answers to Assignment 3: Due Thursday February 20, 2020

1. Give the asymptotic time complexity in terms of n, using Θ, O, or Ω, whichever is most appropriate.

   (a) $F(n) \geq F(n - \sqrt{n}) + n^2$
   $F(n) - \sqrt{n} \geq n^2$
   $\dfrac{F(n) - \sqrt{n}}{\sqrt{n}} \geq n^{3/2}$
   $F'(n) = \Omega(n^{3/2})$
   $F(n) = \Omega(n^{5/2}$

   (b) $H(n) < H(n/3) + H(n/4) + 2H(n/5) + n$
   Since $\frac{1}{3} + \frac{1}{4} + 2 \cdot \frac{1}{5} < 1$
   $H(n) = O(n)$

   (c) $G(n) = 3(G(2n/3) + G(n/3)) + 5n^2$
   Since $3\left(\frac{2}{3}\right)^3 + 3\left(\frac{1}{3}\right)^3 = 1$ and $3 > 2$
   $G(n) = \Theta(n^3)$

2. For each of these recursive subprograms, write a recurrence for the time complexity, then solve that recurrence.

   (a)
```
void george(int n)
 {if(n > 0)
   {for(int i = 0; i < n; i++) cout << "hello" << endl;
    george(n/2); george(n/3); george(n/6);}}
```
   $G(n) = G\left(\dfrac{n}{2}\right) + G\left(\dfrac{n}{3}\right) + G\left(\dfrac{n}{6}\right) + n$
   Since $\frac{1}{2} + \frac{1}{3} + \frac{1}{6} = 1$
   $G(n) = \Theta(n \log n)$

   (b)
```
void martha(int n)
 {if (n > 1)
  {martha(n-1); martha(n-2):}}
```
   Hint: Look at problem 0.3 on page 9 of your textbook.

   $M(n) = M(n-1) + M(n-2) + 1$
   We start by assuming that M is an exponential function. We assume $M(n) = c^n$ for some constant c.
   Substituting: $c^n = c^{n-1} + c^{n-2}$
   Dividing both sides by c^{n-2} we obtain $c^2 = c + 1$ which is a quadratic equation. Thus $c = \dfrac{1+\sqrt{5}}{2}$. Hence
   $M(n) = \Theta\left(\left(\dfrac{1+\sqrt{5}}{2}\right)^n\right)$ (Note that $\dfrac{1+\sqrt{5}}{2}$ is the golden ratio.)

3. The following function computes ab for positive integers a and b. The loop invariant for this code is $p + cd = ab$.

```
int product(int a, int b)
 {
  int c = a;
  int d = b;
  int p = 0;
  while(d > 0)
   {
    if(d % 2) p = p+c;
    c = c+c;
    d = d/2;
   }
  return p;
 }
```

The following function computes x^b for a real number x and a positive integer b. What is its loop invariant?

Answer: $zy^d = x^b$

```
float power(float x, int b)
 {
  float y = x;
  int d = b;
  float z = 1.0;
  // loop invariant holds here
  while(d > 0)
   {
  // loop invariant holds here
    if(d % 2) z = z*y;
    y = y*y;
    d = d/2;
  // loop invariant holds here
   }
  // loop invariant holds here
  return z;
 }
```

4. Consider the following program, where n is a given constant.

```
int A[n];
int B[n];
void getA()
 {
  for(int i = 0; i < n; i++) cin >> A[i];
 }
int main()
 {
  getA();
  int s = 0;
  int i = 0;
  // initialially: numcredit = 2*n;
  // Loop invariant: numcredit  =  2*n-2*i+s.  Holds here since s = i = 0.
  while(i < n) // beginning of outer loop
   {
    // numcredit = 2*n-2i+s Not yet paid for this iteration of the outer loop
    while(s > 0 and B[s-1] > A[i]) // beginning of inner loop
     {
      s--;
      // we chage one credit for this iteration of the inner loop.
      // and numcredit 2n-2i+s is decremented to pay for it.
     }
    B[s] = A[i];
    s++;
    i++;
    // We charge one credit for this iteration of the outer loop.
    // Since i and s are both incremented, numcredit = 2n-2i+s
    // is decreased by 1, which pays for the iteration.
   } // end of outer loop
  for(int j = 0; j < s; j++) cout << B[j];
  cout << endl;
  return 1;
 }
```

(a) If n = 10 and the input stream is 0 6 9 8 1 3 2 5 4 7 what is the output? Ans: 0 1 2 4 7

(b) The inner and outer loops are both linearly bounded, and thus the time complexity of the code is $O(n^2)$. But, it is not $\Theta(n^2)$. Use amortization to prove that the time complexity is $\Theta(n)$.

We will charge one credit each time the outer loop iterates, and one credit each time the inner loop iterates. We initially allocate 2n credits, and use one credit for each iteration. We then need to have at least zero credits remaining at the end. The loop invariant is that numcredit = 2n-2i+s, initially 2n. Eventually numcredit is s, which is at least 0. The number of iterations of both loops together is thus at most 2n. The input and output loops are each $\Theta(n)$, hence the entire code is $\Theta(n)$.