University of Nevada, Las Vegas Computer Science 477/677 Fall 2021

Assignment 3: Due Wednesday September 15, 2021 8:30 AM

Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. Do not turn this homework in. Please bring a printed copy to class on September 15, and we will go over the problems.

The exam on September 20 will be based on the problems in the first two homework assignments and on the problems in this assignment.

1. The following function computes the product of two positive integers. Verify that that `p + cd = ab` is a loop invariant for this code, and use the loop invariant to show that `product(a,b)` returns `a*b`.

```
int product(int a, int b)
 {
  assert (a > 0 and b > 0);
  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;
 }
```

2. The following function computes x^b for a real number x and a positive integer b. What is its loop invariant? Hint: Addition is to Multiplication as Multiplication is to exponentiation. The loop invariant is analogous to the loop invariant of the function `product` in the previous problem.

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

```
 return z;
}
```

3. Most of the following recurrences can be solved using either the master theorem or the generalized master theorem, by the anti-derivative method. Use Θ in your answer if appropriate, otherwise O or Ω.

4. $F(n) = F(n-1) + n^2$

5. $F(n) = 3F(n/3) + n$

6. $F(n) = 3F(n/3) + 3F(2n/3) + n^3$

7. $F(n) = F(n/2) + 2F(n/3) + n^2$

8. $F(n) = F(n/2) + 1$

9. $F(n) = F(\sqrt{n}) + 1$ (Hint: Use substitution: $m = \log n$ and $G(m) = F(n)$.)

10. $F(n) = F(\log n) + 1$ (Hint: this was on a previous homework.)

11. $F(n) \leq F(n/5) + F(7n/10) + n$ (Hint: this should look familiar!)

12. $F(n) = F(n - \sqrt{n}) + \sqrt{n}$

13. $F(n) = F(n - \log n) + n \log n$

14. $F(n) = F(n-1) + F(n/2) + 1$

    I cannot solve this one. (Seriously.)[1]

15. Name each of these seven algorithms.

    (a) Pick an element P from a set S, then partition S into two parts: those items which are less than P and those greater than P. Recorsively sort each part, and combine them to form a sorted list.

    (b) Divide a set S into two roughly equal parts. Recursively sort each part, then combine the two sorted parts to obtain a sorting of S.

    (c) Given a sorted set S and an item x, you need to determine whether $x \in S$. Pick one element, say P, out of S. If $x = P$, you are done. If $x < P$, discard all items of S which are greater then P, while if $x > P$, all items of S which are less then P. Keep doing this until you either find x or you have discarded all items of S.

[1] You might wonder what the purpose of this problem is. I want you to realize that the world is not neatly tied up in blue ribbons. Not not every question has an answer, and not every question that has an answer has an answer that anyone can figure out.

(d) Given a set S, create an empty binary search tree T. Insert the items of S into T one at a time. Finally, visit and print the items of T in left-to-right order, also called inorder.

(e) Given a set S, delete the least element of S and print it. Then delete the least remaining element of S and print it. Keep going until you have deleted and printed all elements of S.

(f) All items of S are in a row. Agents run up and down the row, swapping any two adjacent items if the one on the right is less than the one on the left. Keep going until no more swaps are possible.

(g) You are grading a large number of exams, each of which is labeled with the ID of a student. Each ID consists of five numerals, and no two students have the same ID. You separate them into ten piles, based on the last digit of the ID. You then combine the piles in order, and separate them again into piles, based on the second to last digit of the ID. Do this five times. What will you accomplish?