University of Nevada, Las Vegas Computer Science 477/677 Spring 2024

Assignment 6: Due Saturday April 5 2025 11:59 PM

Upload your homework to Canvas.

Name:_____

You are permitted to work in groups, get help from others, read books, and use the internet. Your answers must be written in a pdf file and uploaded to canvas, by midnight February 16th. Your file must not be unnecessarily long. If you have any questions, or you are having trouble uploading the assignment you may email the grader, Sebrina Wallace, at `wallace4@unlv.nevada.edu`. You may also send me email to ask questions.

1. Fill in the blanks.

   (a) True or false: Open hashing uses open addressing. _____.

   (b) When two data have the same hash value, that is called a _____.

   (c) A _____ hash function gives a 1-1 correspondence between the data and the indices of the hash table.

   (d) In closed hashing, if a collision occurs, one of the data uses a _____ sequence to search for an unused index.

   (e) A connected acyclic graph (not digraph) with 25 vertices must have _____ edges.

   (f) In open hashing, the data which share a hash value must be stored in a _____. (Choose one of these answers: **search structure**, **priority queue**, **virtual array**, **directed graph**.)

   (g) In _____ hashing, each datum has more than one possible hash value.

   (h) An optimal binary prefix code for a given weighted alphabet can be computed using _____ algorithm.

   (i) In an unweighted directed graph, the shortest path between two given vertices can be found by _____-first search. (Choose one of these answers: **Depth**, **Breadth**.)

   (j) Binary search tree sort (or simply *tree sort*) is a fast implementation of _____ sort. (Choose of these answers: **selection**, **bubble**, **insertion**, **quick**.)

   (k) A _____ order of a directed graph G is an ordering of the vertices of G such that vertex x must be come earlier than vertex y in the ordering if there is an arc from x to y,

   (l) The subproblems of a dynamic program must be worked in _____ order.

2. Write the asymptotic time complexity for each code fragment, using Θ notation.

(a)
```
for (int i=1; i < n; i++)
    for (int j=i; j > 0; j--)
```

(b)
```
for (int i=1; i < n; i=2*i)
    for (int j=i; j < n; j++)
```

(c)
```
for (int i=1; i < n; i++)
    for (int j=1; j < i; j = j*2)
```

(d)
```
for (int i=1; i < n; i++)
    for (int j=i; j < n; j = j*2)
```

(e)
```
for (int i=2; i < n; i = i*i)
```

(f)
```
for (int i=1; i*i < n; i++)
```

3. Give an asymptotic solution to each of these recurrences, using the Bentley-Blostein-Saxe method, otherwise known as the master theorem. Some of them may require substitution.

(a) $F(n) = 2F(n/2) + n$

(b) $F(n) = 4F(n/2) + n^3$

(c) $F(n) = 4F(n/2) + n^2$

(d) $F(n) = 4F(n/2) + n$

(e) $T(n) = 7T(n/7) + n$

(f) $T(n) = 9T(n/3) + n^2$

(g) $T(n) = 8T(n/2) + n^3$

(h) $T(n) = T(\sqrt{n}) + 1$ Use substitution: $m = \log n$.

(i) $T(n) = 2T(n-1) + 1$ Use substitution: $n = \log m$, *i.e.* $m = 2^n$.

4. Give an asymptotic solution to each of these recurrences using the Akra-Brazzi method, otherwise known as the generalized master theorem.

   (a) $F(n) = 2F(n/4) + F(n/2) + 1$

   (b) $F(n) = 2F(n/4) + F(n/2) + n$

   (c) $F(n) = 2F(n/4) + F(n/2) + n^2$

   (d) $F(n) = F(3n/5) + F(4n/5) + n^2$

   (e) $F(n) = F(n/3) + 5F(2n/3) + 1$

5. Give an asymptotic solution to each these recurrences, using the anti-derivative method.

   (a) $F(n) = F(n - \log n) + \log n$

   (b) $G(n) = G(n-1) + n^c$ where $c \geq 1$ is a constant.

   (c) $K(n) = K(n - \sqrt{n}) + n$

6. What is the asymptotic complexity of the function `martha(n)` given below, in terms of n? Write a recurrence and solve. (I mean the actual value of `martha(n)`, not the time to compute it.)

```
int martha(int n)
 {
  assert(n >= 0);
  if(n < 1) return 0;
  else return 2*martha(n/2) + n;
 }
```

7. What is the asymptotic time complexity of the above code which computes `martha(n)`? Assume that each addition or multiplication takes constant time.

   If you actually need the value of **martha(n)** and not other values of **martha**, the above recursive code is rather inefficient. Describe a faster method. What is its asymptotic time complexity? Assume that each addition or multiplication takes constant time.

8. What is the asymptotic complexity of the function `george(n)` given below, in terms of n? Write a recurrence and solve. (I mean the actual value of `george(n)`, not the time to compute it.)

```
int george(int n)
 {
  assert(n >= 0);
  if(x <= 1) return 1;
  else return george(3*n/5) + george(4*n/5) + n*n;
 }
```

9. What is the asymptotic time complexity of the above code which computes `george(n)`? Assume that each operation takes constant time.

10. The following table gives two possible hash values for each of a set of 8 data. Can you construct a closed hash table of size 8 which contains all the data?

    If so, construct the table. Otherwise, convince me that it's impossible.

Abe	1	4
Bob	7	3
Cec	5	6
Dan	5	7
Eve	1	6
Fay	2	3
Hal	4	8
Ida	8	2

1	
2	
3	
4	
5	
6	
7	
8	