CS530 - Analysis of Algorithms. Spring Semester 1991. Instructor: Andrei Toom.

THE COMPREHENSIVE LIST OF HOMEWORKS, MIDTERMS AND ADDITIONAL PROBLEMS

HW 1. (Problem 1.3-2.) Write pseudocode for $\text{MERGE}(A, p, q, r)$.

HW 2. (Problem 2.2-4.) Prove that $lg(n!) = \Theta(n \lg n)$ and that $n! = o(n^n)$.

HW 3. (Problem 3.2-1.) Show that $\sum_{k=0}^{n} 1/k^2$ is bounded above by a constant.

HW 4. (Problem 4.2-3.) Draw the recursion tree for $T(n) = 4T(\lfloor n/2 \rfloor) + n$, and provide tight asymptotic bounds on its solution.

HW 5. (Problem 5.1-3.) Prove the principle of inclusion and exclusion: $|A_1 \cup A_2 \cup \cdots \cup A_n| = |A_1| + |A_2| + \cdots + |A_n| - |A_1 \cap A_2| - |A_1 \cap A_3| - \cdots + |A_1 \cap A_2 \cap A_3| + \cdots + (-1)^{n-1} |A_1 \cap A_2 \cap \cdots \cap A_n|$.

HW 6. (Problem 6.1-13.) Use Stirling's approximation to prove that $\binom{2n}{n} = \frac{2^{2n}}{\sqrt{\pi n}}(1 + O(1/n))$.

HW 7. (Problem 7.4-2.) What is the running time of HEAPSORT on an array A of length n that is already sorted in increasing order ? What about decreasing order ?

HW 8. (Problem 8.3-2.) During the running of the procedure RANDOMIZED-QUICKSORT, how many calls are made to the random-number generator RANDOM in the worst case ? How does the answer change in the best case ?

HW 9. (Problem 9.3-2.) Which of the following sorting algorithms are stable: INSERTION-SORT, MERGE-SORT, HEAP-SORT, and QUICK-SORT ? Give a simple scheme that makes any sorting algorithm stable. How much additional time and space does your scheme entail ?

HW 10. (Problem 10.3-8.) Let $X[1..n]$ and $Y[1..n]$ be two arrays, each containing n numbers already in sorted order. Give an $O(\lg n)$-time algorithm to find the median of all $2n$ elements in arrays X and Y.

HW 11. (Problem 11.4-2.) Write an $O(n)$-time procedure that, given an n-node binary tree, prints out the key of each node in the tree.

HW-12. (Problem 12.4-2.) Write pseudocode for HASH-DELETE as outlined in the text, and modify HASH-INSERT and HASH-SEARCH to incorporate the special value DELETED.

HW 13. (Problem 13.4-1.) Describe a binary search tree on n nodes such that the average depth of a node in the tree is $\Theta(\lg n)$ but the height if the tree is $\omega(\lg n)$. How large can the height of an n-node binary search tree be if the average depth of a node is $\Theta(\lg n)$?

HW 14. (Problem 14.2-5.) Show that any arbitrary n-node tree can be transformed into any other arbitrary n-node tree using $O(n)$ rotations. (*Hint:* First show that at most $n - 1$ right rotations suffice to transform any tree into a right-going chain.)

HW 15. (Problem 15.1-5.) Given an element x in a n-node order-statistic tree and a natural number i, how can the i-th siccessor of x in the linear order of the tree be determined in $O(\lg n)$ time ?

HW 16. (Problem 16.3-5.) Give an $O(n^2)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers. Write the pseudocode.

HW 17. (Problem 17.3-2.) What is an optimal Huffman code for the following set of frequencies, based on the first 8 Fibonacci numbers ?

a:1 b:1 c:2 d:3 e:5 f:8 g:13 h:21

Can you generalize your answer to find the optimal code when the frequencies are the first n Fibonacci numbers ?

HW 18. (Problem 18.1-3.) A sequence of n operations is performed on a data structure. The i-th operation costs i if i is an exact power of 2, and 1 otherwise. Determine the amortized cost per operation.

HW 19. (Problem 19.2-6.) Suppose that B-TREE-SEARCH is implemented to use binary search rather than linear search within each node. Show that this makes the CPU time required $O(\lg n)$, independently of how t might be chosen as a function of n.

HW 20. (Problem 23.4-3.) Give an algorithm that determines whether or not a given undirected graph $G = (V, E)$ contains a cycle. (It is especially valued if your algorithm will run in $O(V)$ time, independent of $|E|$.)

HW 21. (Problem 24.2-2.) Suppose that the graph $G = (V, E)$ is represented as an adjacency matrix. Give an implementation of Prim's algorithm for this case that runs in $O(V^2)$ time. Write a pseudocode.

HW 22. (Problem 25.2-2.) Give an example of a directed graph with negative-weight edges for which Dijkstra's algorithm produces incorrect answers. Why doesn't the proof of Theorem 25.10 go through when negative-weight edges are allowed ?

HW 23. (Problem 26.2-5.) How can the output of the Floyd-Warshall algorithm be used to detect the presence of a negative-weight cycle ?

HW 24. (Problem 27.2-9.) The *edge connectivity* of an undirected graph is the minimum number of edges that must be removed to disconnect the graph. Write an algorithm which determines the edge connectivity of any given $G = (V, E)$ and analyze its running time.

HW-25. (Problem 30.2-3.) Give an n-processor CRCW algorithm that can compute the OR of n boolean values in $O(1)$ time.

HW 26. (Problem 33.5-2.) Find all integers x that leave remainders $1, 2, 3, 4, 5$ when divided by $2, 3, 4, 5, 6$, respectively.

HW 27. (Problem 34.1-2.) Show that the worst-case time for the naive string matcher to find the *first* occurence of a pattern in a text is $\Theta((n - m + 1)m)$.

HW 28. (Problem 36.1-6.) Show that an algorithm that makes at most a constant number of calls to polynomial-time subroutines runs in polynomial time, but that a polynomial number of calls to polynomial-time subroutines may result in an exponential-time algorithm.

HW 29. $P(m, n)$ is the probability that x divides y where x is chosen at random among $\{1, 2, \ldots, m\}$ and y is chosen at random among $\{1, 2, \ldots, n\}$. Calculate $P(m, m!)$ to two significant figures for m equal to a million. (*Hint:* $\ln 10 = 2.302585\ldots$, Euler's constant $\gamma = 0.57721\ldots$.)

MT 1-1. (Problem 1.1-3.) Consider the *searching problem:*
Input: A sequence of n numbers $A = < a_1, a_2, \ldots, a_n >$ and a value v.
Output: An index i such that $v = A[i]$ or the special value NIL if v does not appear in A.
Write pseudocode for LINEAR-SEARCH, which scans through the sequence, looking for v.

MT 1-2. (Problem 1.3-3.) Use mathematical induction to show that the solution of the following recurrence is $T(n) = n \lg n$:

$$T(n) = \begin{cases} 2 & \text{if } n = 2, \\ 2T(n/2) + n & \text{if } n = 2^k, k > 1. \end{cases}$$

MT 1-3. (Problem 2.2-5.) Is the function $\lceil \lg n \rceil!$ polynomially bounded ?

MT 1-4. (Problem 3.2-2.) Find an asymptotic upper bound on the summation $\sum_{k=0}^{\lfloor \lg n \rfloor} \lceil n/2^k \rceil$.

MT 1-5. (Problem 4.1-1.) Show that the solution of $T(n) = T(\lceil n/2 \rceil) + 1$ is $O(\lg n)$.

MT 1-6. (Problem 5.2.) Reword the following statement as a theorem about undirected graphs, and then prove it. Assume that friendship is symmetric but not reflective.
Every group of six people contains either three mutual friends or three mutual strangers.

MT 1-7. (Problem 6.1-5.) Prove the identity $\binom{n}{k} = \frac{n}{k}\binom{n-1}{k-1}$ for $0 < k \leq n$.

MT 2-1. A library wants to make an 'alphabetically' ordered list of titles of the books it has. Formulate the rule according to which one title should precede the other. Note that the titles may contain punctuation marks and decimal ciphers (like '1984' by George Orwell).

MT 2-2. How many elements there may be in a heap of height h ? Give the exact formula for the height of a heap which has n elements.

MT 2-3. The following algorithm is given:

Random-Permutation($A[1..n]$)
for $i \leftarrow 2$ to n do
    swap $A[i] \leftrightarrow A[\text{Random}(1, i)]$

It takes as input an array $A[1..n]$ and performs permutations on the array elements with some probabilities. Which are the smallest and the largest of these probabilities ? (Assume that all the elements of the input array are different.)

MT 2-4. Professor Brag claimes to have a comparison sort whose running time is $O(n)$ for a fraction $1/2^n$ of the $n!$ possible inputs of length n. Prove that he is wrong.

MT 2-5. You are given a subroutine Median which finds the median(s) of any given array in a linear time. Using it, give a linear-time algorithm that solves the selection problem for an arbitrary order statistic.

MT 3-1. (Problem 11.4-4.) Write an $O(n)$-time procedure that prints all the keys of an arbitrary rooted tree with n nodes, where the tree is stored using the left-child, right-sibling representation.

MT 3-2. (Compare Problem 11.2-7.) Write a $\Theta(n)$-time procedure that reverses a singly linked list of n elements. The procedure should use no more than constant storage beyond that needed for the list itself.

MT 3-3. n elements are simply uniformly hashed into $2n$ slots. Give an asymptotically tight bound (when $n \to \infty$) for the logarithm of the probability that there are no collisions.

MT 3-4. Suppose that a node is deleted from a binary search tree with Tree-Delete and then immediately inserted back with Tree-Insert. May the resulting binary search tree differ ¿from the original one ? If yes, give an example; if not, prove.

MT 3-5. Let y and x be nodes of a binary search tree. Let y be the successor of x in the sorted order determined by the inorder tree walk. Is it possible that x has a right child and y has a left child ? If yes, give an example; if not, prove.

MT 3-6. (Problem 15.1-3.) Write a non-recursive version of OS-SELECT.

MT 4-1. Write a pseudocode of a procedure which returns the length of the longest common subsequence of any three given sequences

$$X =< x_1, \ldots, x_m >, \quad Y =< y_1, \ldots, y_n >, \quad Z =< z_1, \ldots, z_p > .$$

(Their elements are letters of some finite alphabet.) Analyze the running time of your procedure.

MT 4-2. (Compare Problem 17.2-5.) Several houses are placed at some points of a straight road. The smallest possible number of fire stations must be built at the same road so that every house would be within 1-mile range of some fire station. Write a pseudocode of a procedure which solves this problem and analyze its running time.

MT 4-3. An alphabet consists of four letters all of which have certain positive frequencies the sum of which equals 1. It has a Huffman code in which all the four codewords have one and the same length 2. Let L stand for one of the letters. What can you say about its frequency $f(L)$? (Give the set of its possible values.)

MT 4-4. (Problem 18.1-1.) If a MULTIPUSH operation were included in the set of stack operations, would the $O(1)$ bound on the amortized cost of stack operations continue to hold ?

MT 4-5. A k-bit binary counter counts upward from 0 and adds a constant number d (modulo 2^k) at each operation. What is the amortized cost of each operation ?

MT 5-1. (Compare Problem 23.1-3.) The *transpose* of a directed graph $G = (V, E)$ is the graph $G^T = (V, E^T)$, where

$$E^T = \{(u, v) \in V \times V : (v, u) \in E\}.$$

Thus, G^T is G with all its edges reversed. Describe an algorithm for computing G^T from G, for adjacency-list representations and analyze the running time.

MT 5-2. (Compare Problem 23.2-6.) A *bipartite* graph is an undirected graph $G = (V, E)$ in which V can be partitioned into two sets L and R such that $(u, v) \in E$ implies either $u \in L$ and $v \in R$ or $u \in R$ and $v \in L$. Give an algorithm to determine if an undirected graph is bipartite and analyze its running time.

MT 5-3. (Problem 24.2-6.) Give an algorithm that, given an undirected weighted graph $G = (V, E)$, determines a spanning tree of G whose largest edge weight is minimum over all spanning trees of G. Analyze the running time of your algorithm.

MT 5-4. (Problem 25.2-4.) We are given a directed graph $G = (V, E)$. Each edge $(u, v) \in E$ is interpreted as a communication channel and has an associated value $r(u, v) \in [0, 1]$, which represents the probability that the channel from u to v will not fail. Assume that failures of different channels are independent. Give an algorithm to find the most reliable path between two given vertices.

MT 5-5. (Compare Problem 26.4-6). A firm wishes to send a truck from city A to city B laden as heavily as possible. Capacity of the truck is unlimited, but each road has a maximum weight limit on trucks that use the road. Describe the graph problem which models this situations. Give an algorithm to solve it and analyze its running time.

A-1. For every natural n decide, what is greater: $2^{(3^{2n})}$ or $3^{(2^{3n})}$.

A-1'. For every natural n decide, what is greater: $2^{(3^{(2^{3n})})}$ or $3^{(2^{(3^{2n})})}$.

A-2. An infinite sequence $x_0, x_1, \ldots, x_n, \ldots$ is defined by the rule $x_n = x_{n-1} + 2x_{n-2}$ for all $n \geq 2$. The initial conditions: $x_0 = 0$ and $x_1 = 1$. Calculate an exact formula for x_n and $\lim_{n\to\infty}((\lg x_n)/n)$.

A-3. A fair die (in the form of a regular cube whose faces are marked $1, 2, 3, 4, 5, 6$) is cast $6n$ times. Derive an exact formula for the probability that every face will come out exactly n times. Use the Stirling's approximation to transform it into an asymptotic formula which contains no factorials.

A-4. Write an algorithm which, given a natural number n, prints all the permutations of $1, 2, \ldots, n$ and nothing else. Everyone of these permutations must be printed only once, in brackets. The running time must be O(printing time).

A-5. Write an algorithm which, given a natural number n, prints those and only those strings of the length n which consist of letters A and B and in which A never immediately follows another A. Everyone of these strings must be printed only once, in brackets. The run time must be O(printing time).

A-6. Write an algorithm which, given a natural number n, prints those and only those strings of the length $2n$ which consist of letters A and B and which contain A and B equal (that is n) number of times. Everyone of these strings must be printed only once, in brackets. The run time must be O(printing time).

A-7. Write an algorithm which, given two natural numbers, calculates their least common multiple. Analyze the running time.

A-8. Write an algorithm which, given a natural number n, decides whether it is prime or not. The running time must be $O(\sqrt{n})$.

A-9. Write an algorithm to convert a given n-bit binary integer to a decimal representation assuming that one operation can handle only $O(1)$ binary or decimal digits.. Analyze the running time.

A-10. Write a parallel $O(\lg n)$-time algorithm which compares two n-digit natural decimal numbers assuming that one operation can handle only $O(1)$ decimal digits. The algorithm must decide whether the first or the second number is greater or they are equal.

A-11. Write a parallel $O(\lg n)$-time algorithm which, given a real number x and a natural n, computes the values $x^2, x^3, \ldots, x^n$.

A-12. Call a directed graph $G(V, E)$ *strongly connected* if every two vertices are reachable from each other. Call a subgraph $G' = (V', E')$ of $G = (V, E)$ *spanning for* G if G' is strongly connected and $V' = V$. Write an algorithm which, given a directed weighted graph G, finds its spanning subgraph whose weight is minimal or returns NIL if G has no spanning subgraph. Analyze the running time.

A-13. A plane with a Cartesian coordinate system is given. The *Manhattan distance* $M(P_1, P_2)$ between two points $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ in this plane stands for

$$M(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|.$$

Write an $O(n)$-time algorithm which, given n points in the plane, finds such two of these points for which the Manhattan distance between them is maximal.

Q-1. Give an example of a function of a natural argument which grows faster than all polinoms but slower than all exponents.

Q-2. Calculate $\lim_{n\to\infty}(\binom{\lg(3n}{n)}/n)$.

Q-3. Several kids were driven to school in several buses. Statistician Alpha asked every driver how many kids were in his bus and calculated the average. Statistician Beta asked every kid how many kids (including himself) were in his bus and calculated the average. Are these averages equal ? If not, which is greater ? (Assume that nobody miscalculates or cheat.)

Q-4. Alice and Bob decided to have a date. They agreed to come to a certain place between 5pm and 6pm and wait for each other for 15 minutes. What is the probability that they meet ?

Q-5. The computer language you use has a generator which generates random bits, i.e. independent Boolean variables which equal 0 or 1 with frequencies 1/2 and 1/2. Describe a procedure which outputs a Boolean value which equals 0 with probability 2/3 and 1 with probability 1/3.

P-1. Nineteen students' scores are independent random naturals in the range $[1, 250]$. Professor Jeopardy bets that all the scores are different. What is more probable: that he wins or loses ?

P-2. $P(m, n)$ is the probability that x divides y where x is chosen at random among $\{1, 2, \ldots, m\}$ and y is chosen at random among $\{1, 2, \ldots, n\}$. These choises are uniform and independent. Calculate $P(m, m!)$ to two significant figures for m equal to a million. (*Hint:* $\ln 10 = 2.302585\ldots$, Euler's constant $\gamma = 0.57721\ldots$.)

In the following problems write an algorithm which prints all the elements of a given set S, every element of S being printed only once in brackets, the run time being time of printing plus $O(|S|)$.

P-3. S is the set of those strings of length $2n$ which consist of n zeroes and n ones.

E-1. (Problem 16.3-6) Give an $O(n \lg n)$-time algorithm to find the longest monotonically increasing subsequence of a sequence of n numbers.

E-2. (BB) Governor elections are held in the state Majorland. A candidate is considered elected if more than half of the voters chose this candidate's name. Write an $O(n)$-time algorithm which, given an array of n names, chosen by n voters, decides whether a governor is elected or not and if yes, outputs the name.

SOLUTIONS

A-5. Call PRINTSTRING$(n, \emptyset)$ where $+$ means concatenation:

```
PRINTSTRING(n, prefix)
case n
            0 : print prefix;
            1 : PRINTSTRING(n − 1, prefix + A);
                PRINTSTRING(n − 1, prefix + B);
    otherwise : PRINTSTRING(n − 2, prefix + AB);
                PRINTSTRING(n − 1, prefix + B);
```

A-6. To calculate the least common multiple of a and b, call LCM(a, b):

```
LCM(a, b)
ma ← a
mb ← b
while ma ≠ mb do
    if ma < mb
```

$ma \leftarrow ma + a$

else

$mb \leftarrow mb + b$

return(ma)

From: dgd David G. Durand   ID# 038-48-1537

Subject: Homework assignment --- Merge function

```
Merge(A, p, q, r)     |> |p      q|q+1     r|
declare C[r-p+1]
i := p
j := q+1
for k := 1 to length[C]
   if i <= q and j <= r then
     if A[i] < A[j] then
        C[k] := A[i]
        i := i + 1
     else
        C[k] := A[j]
        j := j + 1
     endif
   else
     if i <= q
        C[k] := a[i]
        i := i + 1
     else
        C[k] := A[j]
        j := j + 1
     endif
   endif

for k = 1 to length[C]
    A[p+k-1] := C[k]
===========================================
```

From: nagy@bucsf.bu.edu (Sue Nagy)

```
CS530                               Susan Nagy
Homework #1                         032547408

procedure merge(A,p,q,r)

{ procedure to combine two sorted lists - one from A[p..q] }
{ and the other from A[q+1..r] into A[p..r]                }

lowfirst = p           { lower bound of A[p...q] }
lowsecond = q+1        { lower bound of A[q+1..r] }
index = 1              { index for new array B }
while (lowfirst <= q) and (lowsecond <= r) do   { while neither empty }
  if A[lowfirst] < A[lowsecond] then            { copy from A[p..q] }
    B[index] = A[lowfirst]
```

```
    lowfirst = lowfirst + 1
  else                                         { copy from A[q+1..r] }
    B[index] = A[lowsecond]
    lowsecond = lowsecond + 1
  index = index + 1
while (lowfirst <= q) do                     { copy remaining elements }
  B[index] = A[lowfirst]                     { from A[p..q] }
  lowfirst = lowfirst + 1
  index = index + 1
while (lowsecond <= r) do                    { copy remaining elements}
  B[index] = A[lowsecond]                    { from A[q+1..r] }
  lowsecond = lowsecond + 1
  index = index + 1
j = p                                        { save p }
for i = 1 to (index - 1) do                  { copy elements of B }
  A[j] = B[i]                                { back to A }
  j = j + 1
=========================================================================
From: rinnie (Harina Kapoor  BU#  017 72 5671)

The C code for HW-1:

#include <stdio.h>

copy(b,c,j,n,k)
float b[20],c[40];
int j,n,*k;
{
  int i,kk;
  kk=*k;
  for(i=j;i<=n;i++){
    c[kk] = b[i];
    kk++;}
  *k=kk;
}

shortmerge(a,b,c,m,j,k)
float a[20],b[20],c[40];
int m,*j,*k;
{
  int i,jj,kk;
  i=1;
  jj=*j;
  kk=*k;
  while (i<=m){
    if(a[i] <= b[jj]){
      c[kk] = a[i];
      i++;}
    else {
      c[kk] = b[jj];
      jj++;}
```

```
    kk++;
  }
  *k=kk;
  *j=jj;
}

mergecopy(a,b,c,m,n)
float a[20],b[20],c[40];
int n,m;
{
  int i,j,k,jj,kk;
  i=1;j=1;k=1;
  if (a[m] <= b[i]){
    copy(a,c,i,m,&k);
    copy(b,c,j,n,&k);
  }
  else {
    shortmerge(a,b,c,m,&j,&k);
    copy(b,c,j,n,&k);
  }
}

merge(a,b,c,n,m)
float a[20],b[20],c[40];
int n,m;
{
    if (a[m] <= b[n])
    mergecopy(a,b,c,m,n);
    else
    mergecopy(b,a,c,n,m);
}

input(a,b,n,m)
float a[20],b[20];
int *n,*m;
{
  int i,j,tn,tm;
  float data;
  printf("enter the maximum number of data elements in array a\n");
  scanf("%d",&tm);
  printf("enter max number into array b\n");
  scanf("%d",&tn);

  for(i=1;i<=tm;i++){
    printf("enter data into array a\n");
    scanf("%f",&data);
    a[i] = data;
    data = 0.0;
  }
  *m=tm;
  data  = 0.0;
```

```
  for(j=1;j<=tn;j++){
    printf("enter hte data in array b\n");
    scanf("%f",&data);
    b[j] = data;
    data = 0.0;
  }
  *n=tn;
}

main()
{
  int i, j, k, l, n, m, p,nn,mm;
  float data, key, temp, a[20], b[20], c[40];
  int nm;
  input(a,b,&n,&m);
  nn=n;
  mm=m;
  merge(a,b,c,nn,mm);
  nm = nn+mm;
  for (p = 1; p <= nm; p++){
    printf("c[%d] = %f\n",p,c[p]);
  }
}
======================================================
From: ansh (Anshu Chandra)
I was wondering if you could help us solve 26.2-7 ?

Anshu
======================================================
Tomorrow during the review, would you discuss problems
25.2-5, 25.2-6, and 26.2-7?  Thank you.

Do you think you can discuss problem 30.1-4 in class?
A few of us were talking about it and couldn't
quite figure out how to solve it.

Sue
=======================================================
```