Boğaziçi University, Dept. of Computer Engineering

CMPE 250, DATA STRUCTURES AND ALGORITHMS

Fall 2010, Midterm 1

Name: ________________________________

Student ID: __________________________

Signature: ____________________________

- Please print your name and student ID number and write your signature to indicate that you accept the University honour code.
- During this examination, you may not use any notes or books.
- Read each question carefully and WRITE CLEARLY. Unreadable answers will not get any credit.
- There are 6 questions. Point values are given in parentheses.
- You have 100 minutes to do all the problems.

<table>
<thead>
<tr>
<th>Q</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Score</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>15</td>
<td>80</td>
</tr>
</tbody>
</table>
1. What is .. (Give short answers. Long answers do not get any credit.)

(a) the notation \( O(f(n)) = g(n) \) ? (1pt) **Solution:**

for some constant \( C \) and \( n_0, C f(n) \) is an upper bound of \( g(n) \) for \( n > n_0 \).

(b) the notation \( \Theta(f(n)) = g(n) \) ? (1pt) **Solution:**

for some constants \( C_1, C_2 \) and \( n_0, C_1 f(n) \leq g(n) \leq C_1 f(n) \) for \( n > n_0 \).

(c) the notation \( o(f(n)) = g(n) \) ? (1pt) **Solution:**

Strict lower bound

(d) a Stack ? (1pt) **Solution:**

A list with first in last out property

(e) a Queue ? (1pt) **Solution:**

A list with first in first out property

(f) a Dequeue ? (1pt) **Solution:**

Double ended queue.

(g) the postfix expression for \((a + b) * (-a) + c/g\) ? (1pt) **Solution:**

\( ab + a - *cg/+ \)

(h) the prefix expression for \( abcd + *-\) ? (2pts) **Solution:**

Infix: \( a - b * (c + d), \) prefix \( -a * b + cd \)

(i) a Catalan number ? (1pt) **Solution:**

Number of binary trees with \( n \) nodes

(j) a Bell number ? (1pt) **Solution:**

Number of partitions of a set
(k) the meaning of the expression int* e; in C++? (1pt) Solution:
a pointer to an int

(l) the output of the following code segment C++? Explain (2pts)

```cpp
char a = 'a'; char& c=a; c = 'c'; cout << 'a' << a << 'c' << c;
```

Solution:
Output: accc. c is an alias of a.

(m) a possible way of allocating dynamic memory in C++? (1pt) Solution:
Mem = new char[size];
2. Given the following code segment, give a precise mathematical analysis of the running time in terms of $N$

   1. $\text{sum} = 0$;
   2. $M = \text{random}(1, N); \ // A \text{ Random number between } 1 \text{ and } N$
   3. $\text{for } (i=0; i<M; i++) \{$
   4. $\text{for } (j = 0; j<i; j++) \{ \text{sum++; } \}$
   5. $\}$

Number of operations:

   1. $\text{sum} = 0$; 1
   2. $M = \text{random}(1, N)$; 1
   3. $\text{for } (i=0; i<M; i++) \{ M$
   4. $\text{for } (j = 0; j<i; j++) \{ \text{sum++; } i$
   5. $\}$

Hence we have

   $T(M) = M(M + 1)/2$

But $M < N$ so $M(M + 1)/2 < N(N + 1)/2 = O(N^2)$

(10 points)

3. (a) Write a function $\text{swap(Node* p, Node* q)}$ to swap two elements of a doubly linked list without moving the data and only by adjusting the links. Assume the convention that the list contains only a dummy head node but no dummy tail node.

Solution:
Check if $p$ and $q$ are valid. (Both Not NULL and not equal to )

(10 points)
4. Write a fast exponentiation routine without using recursion. (You can assume the presence of a stack object with appropriate operations. You can also give pseudocode to hide unimportant details.) Solution:

The recursive implementation covered in class was

```c
long pow(long x, long n) {
    if (n==0) return 1;
    if (n==1) return x;
    else {
        if (n % 2 == 0) return pow(x*x, n/2);
        else return pow(x*x, n/2)*x;
    }
}
```

This function is computing the following quantities: $x$, $x^2$, $x^4$, ..., $x^{2i}$ etc. When $n/2^i$ is odd we have an additional factor $x^{2^i-1}$. Rest accumulates the product of additional factors.

```c
long pow(long x, long n) {
    if (n==0) return 1;
    rest = 1;
    while (n>1) {
        if (n % 2) rest *= x;
        x *= x;
        n /= 2;
    }
    return x*rest;
}
```

Actually, a stack is not really needed.
5. (a) Find a recursive expression for the number of distinct ternary trees with $N$ nodes (trees where each node has at most three children)?

(b) Call the number of unique partitions of a set $A$ with $N$ elements $P(N)$. (For example $A = \{a, b, c\}$ implies we have the partitions $(a)(b)(c)$, $(a, b)(c)$, $(a, c)(b)$, $(b, c)(a)$, $(a, b, c)$ so $P(3) = 5$). Find a recursive expression for $P(N)$.

(c) Design an efficient algorithm to calculate $P(N)$.

Solution:

(a) We already know the number for binary trees with $n$ nodes

$$T_2(n + 1) = \sum_{k=0}^{n} T_2(k)T_2(N-k)$$

This was derived by excluding the root and identifying the number of distinct trees. Now, consider the root of a ternary tree. We can choose the first subtree with $k$ nodes. The second subtree will have $l$ nodes and the third one will contain the rest $N-(k+l)$. The total will be

$$T_3(N+1) = \sum_{k=0}^{N} \sum_{l=0}^{N-k} T_3(k)T_3(l)T_3(N-k-l)$$

(b) (This problem was already announced during the lectures – one solution is sketched below)

![Tree T for generating all partitions of sets with 1 to 4 elements.](image)

From the figure we see that

$$P(1) = 1$$
$$P(2) = 2$$
$$P(3) = 5$$
$$P(4) = 15$$

The pattern is as follows: Consider a node of the tree $T$ corresponding to a partition with $M$ disjoint sets. The children will be generated as follows: A new element will be inserted to one of the $M$ sets or a new one, giving a total of $M+1$ children. $M$ of these nodes will have $M$ sets and one will have $M+1$ sets. In the table below, $m$ correspond to the number of partitions.
and \( n \) denotes the number of objects. The entry \( S_{m,n} \) denotes the number of partitions of \( n \) elements that have \( m \) sets.

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & n-1 & n \\
1 & 1 & 1 & 1 & 1 & 1 \\
2 & 1 & 3 = 1 + 2 \times 1 & 7 = 1 + 2 \times 3 & & \\
3 & 1 & 6 = 3 + 3 \times 1 & & & \\
4 & & 1 & & & \\
m-1 & & S_{m-1,n-1} & & & \\
m & S_{m,n} & S_{m,n} = S_{m-1,n-1} + mS_{m,n-1} & & & \\
\hline \\
\text{Sum} & 1 & 2 & 5 & 15 & P(n) = \sum_{m=1}^{n} S_{m,n} \\
\end{array}
\]

(c)

```c
long int S[N+1][N+1];
long int P[N+1];

S[1][1] = 1;
P[1] = 1;
for (n=2; n<=N; n++)
    P[n] = S[1][n] = 1;
for (m=2; m<=N; m++) {
    S[m][n] = S[m-1][n-1] + m*S[m][n-1];
    P[n] += S[m][n];
}
```

(15 points)
6. What is the output of the following C++ program? For each line numbered from 1-6, write the output and explain. (Hint: Be careful with implicit calls to constructors and destructors).

```cpp
struct obj {
    int i;
    obj() {cout << '+';}
    obj(obj& o2) {this->i=o2.i; cout << '<';}
    ~obj() {cout << '-';}
    obj& operator=(obj& o2) {this->i=o2.i; cout << '='; return o2;}
};
void fun1(obj& o) {o.i=1; cout << '1'; return;} 
void fun2(obj o) {o.i=2; cout << '2'; return;}

int main() {
    obj o; Output: +
    fun1(o); cout << o.i; Output: 11
    fun2(o); cout << o.i; Output: <2-1
    obj o2 = o; Output: <
    o2 = o; Output: =
    return 0; Output: ——
}
```

(15 points)