浙江大学 2007－2008 学年冬季学期

《高级数据结构与算法分析》课程期末考试试卷

开课学院：软件学院、计算机学院、竺可桢学院 ，考试形式：**闭卷**，允许带 无 入场

考试时间：2008 年 1 月 24 日， 所需时间：120 分钟

考生姓名：_____学号：_____专业：_____教师：_____

题序	一	二	三	四	总 分
得分					
评卷人					

Answer Sheet

Part I				
1. b	2. a	3. d	4. bd	5. ab
6. b	7. d	8. bd	9. b, a	10. c

Part II

1.

① `H1->Left = H2`

② `H1->Left->Npl < H1->Right->Npl`

③ `H1->Right->Npl + 1`

2.

① `K2->Left`

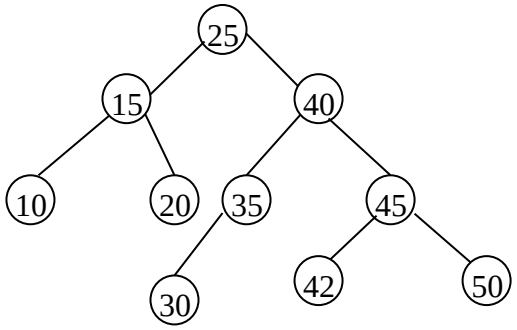
② `K2->Right`

③ `K2`

Part III

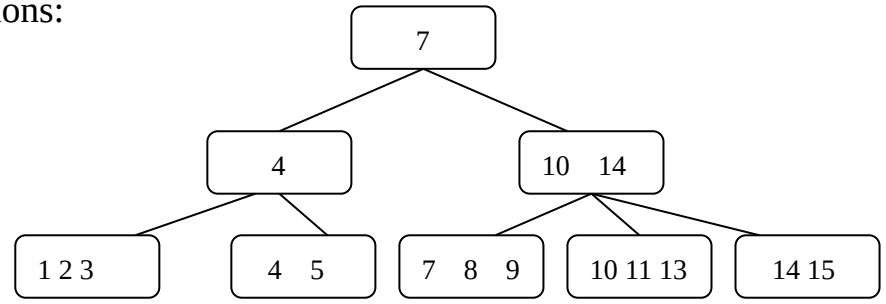
1.

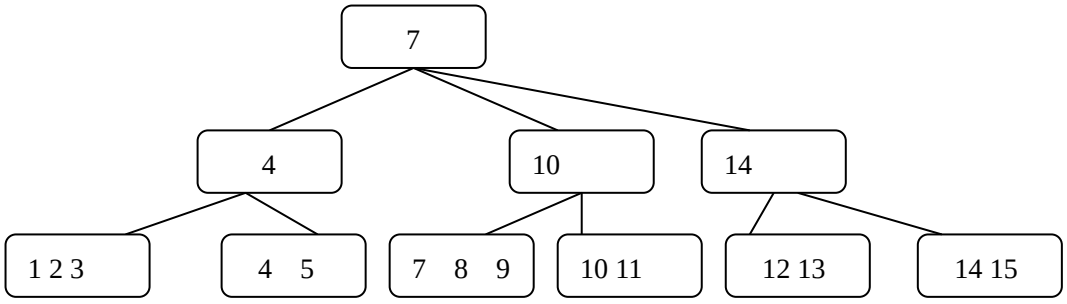
Average search time
= (1+2*2+3*4+4*3)/10
= 2.9

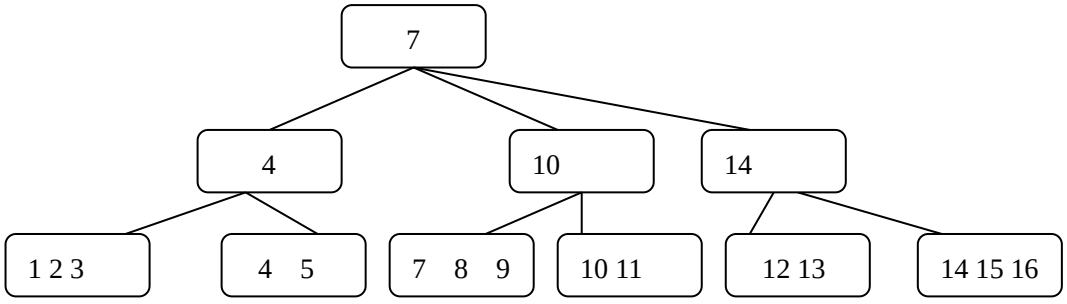


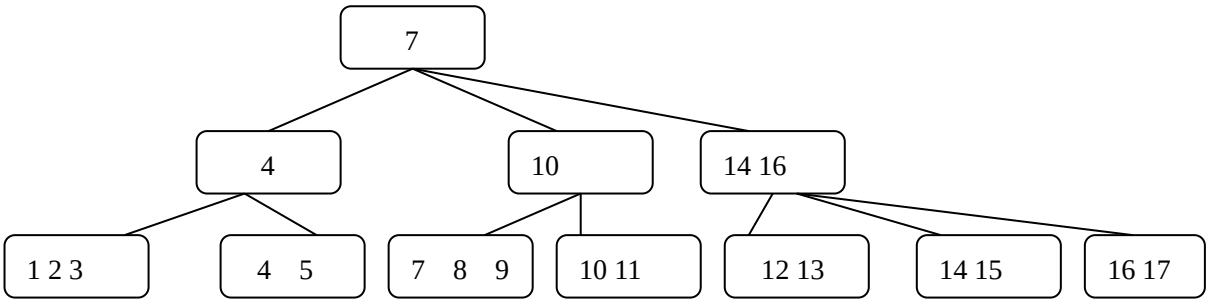
2.

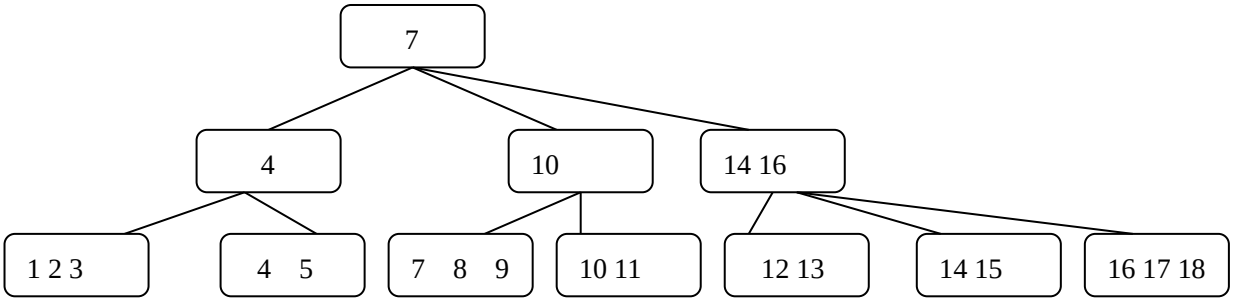
a. Insertions:



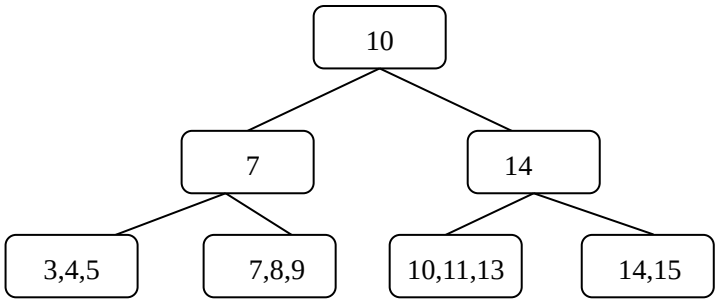


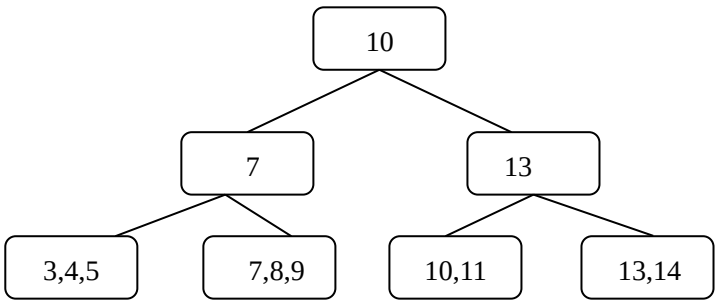






b. Deletions:





3.

a: 184

b: 69

c: 60

-	D	G	H
110	11110	11111	1011
I	O	S	T
1110	0	100	1010

4.

Step 1: FindMin in *Bk*
Step 2: Remove *Bk* from *H* and get *H'*
Step 3: Remove root from *Bk* and get *H"*
Step 4: Merge (*H'*, *H"*)

5. test on:

else+{if..ok} => 0.25+0.12+0.08+0.02=0.47
else+if+ok => 0.02+(0.25+0.08)*2=0.68
{else..if}+ok => 0.08+0.29+0.25+0.02=0.64
Hence the optimal tree has the root else, with minimum cost of 0.47

6a: 5400	6b: 2500, $((M_1\ M_2)(M_3\ M_4))$	6c: 2500, $((M_1\ M_2)(M_3\ M_4))$

Part IV

```
void Connectivity( TwoDimArray A, TwoDimArray C, int N)
{
   int i,j,k;

   for (i=0; i<N; i++)
      for (j=0; j<N; j++)
         if ( i == j )     C[i][i] = 1;
         else     C[i][j] = A[i][j];

   for (k=0; k<N; k++)
      for (i=0; i<N; i++)
         for (j=0; j<N; j++)
            if (C[i][k] && C[k][j])
               C[i][j] = 1;
}
```

NOTE: Please write your answers on the answer sheet.

注意：请将答案填写在答题纸上。

I. Please fill in the blanks (There could be multiple answers for one blank). (22 points) *Note: Zero point for a blank selection since there is at least one answer for each problem.*

(1) Is an AVL tree a binary search tree? Is an AVL tree a complete binary tree? _____. (2 points)

a. Yes. Yes. b. Yes. No. c. No. Yes. d. No. No.

(2) To delete X from a splay tree, the operations are: (1) Find X; (2)Remove X; (3) FindMax(T_L); and the last operation is _____.(2 points)

a. Make T_R the right child of the root of T_L

b. Make T_R the left child of the root of T_L

c. Make T_L the right child of the root of T_R

d. Make T_L the left child of the root of T_R

(3) Which of the following sentence(s) is(are) true? _____ (2 points)

a. The Hamiltonian cycle problem is not NP-complete

b. All decidable problems are in NP

c. All decidable problems are not in NP

d. Not all decidable problems are in NP

(4) Leftist heaps _____.(2 points)

a. are not binary trees b. process Merge in O(N) time

c. have the same structural property as the ordinary binary heaps

d. have the same ordering property as the ordinary binary heaps

(5) Skew heaps _____. (2 points)

a. have O(logN) amortized cost per operation

b. do not need to maintain the null path length of any node

c. no longer have the same ordering property as the ordinary binary heaps

d. have O(logN) worst-case cost for merging

(6) A binomial queue of size 42 can be represented by the following binomial queues _____. (2 points)

a. B_0 B_1 B_2 B_3 B_4 B_5 b. B_1 B_3 B_5 c. B_1 B_5 d. B_2 B_4

(7) The turnpike reconstruction problem is to reconstruct a point set from distances between every pair of points. Given a set of distances {1, 2, 3, 4, 4, 5, 6, 7, 8, 10}, there are 5 corresponding points. What is the **next point** after obtaining x1=0, x5=10, and x4=8, if we solve this problem by backtracking method? _____(2 points)

a. x3=7 b. x2=1 c. x2=2 d. x2=3

(8) Which of the following data structure(s) can achieve the amortized time bound of O(logN) while the worst-case complexity is still O(N) for each single operation? _____ . (2 points)

a. AVL trees b. Splay trees c. binary search trees d. Skew heaps

(9) To solve a problem with input size N by divide and conquer algorithms, if the conquer step takes O(N logN) extra work to form the solution from the sub-solutions, then among the following four dividing methods, _____ (2 points) is the best one while _____ (2 points) is the worst one.

a. divide into 4 sub-problems of equal complexity N/3
b. divide into 3 sub-problems of equal complexity N/4
c. divide into 4 sub-problems of equal complexity N/4
d. divide into 3 sub-problems of equal complexity N/3

(10) Given 3 items and a knapsack with capacity 20. The items have weights 18, 15 and 10, and profits 25, 24 and 15, respectively. The percentages of the items that should be packed to obtain maximum profit is _____. (2 points)

a. 100%, 13.33%, 0 b. 0, 66.67%, 100%
c. 0, 100%, 50% d. 16.67%, 100%, 20%

II. Given the function descriptions of the following two (pseudo-code) programs, please fill in the blank lines. (18 points)

(1) The function is to merge two leftist heaps H1 and H2. (9 points)

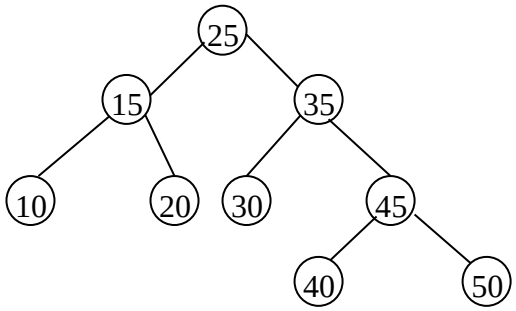
```
PriorityQueue  Merge ( PriorityQueue H1, PriorityQueue H2 )
{   if ( H1 == NULL )   return H2;
    if ( H2 == NULL )   return H1;
    if ( H1->Element < H2->Element )  return Merge1( H1, H2 );
    else return Merge1( H2, H1 );
}
static PriorityQueue
Merge1( PriorityQueue H1, PriorityQueue H2 )
{   if ( H1->Left == NULL )
        ① _____;
    else {
        H1->Right = Merge( H1->Right, H2 );
        if ( ② _____)
            SwapChildren( H1 );
        H1->Npl = ③_____;
    }
    return H1;
}
```

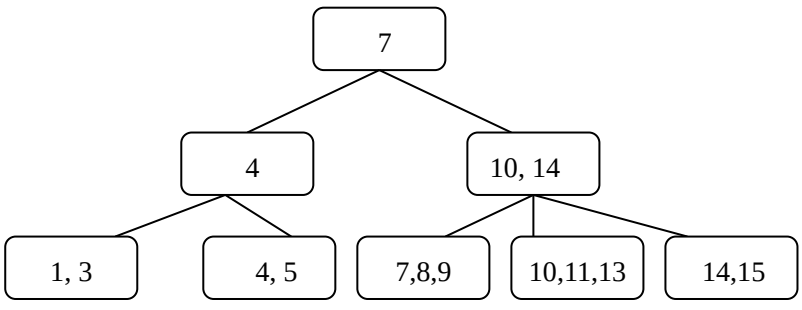
(2) The function is to do the left—right double rotation for an AVL tree, where K is the trouble finder. (9 points)

```
static Position  DoubleRotateWithLeft( Position  K )
{   Position  K1, K2;
    K1 = K->Left;
    K2 = K1->Right;
    K1->Right = ①_____;
    K->Left = ②_____;
    K2->Left = K1;
    K2->Right = K;
    K1->Height = Max( Height(K1->Left), Height(K1->Right) ) + 1;
    K->Height = Max( Height(K->Left), Height(K->Right) ) + 1;
    K2->Height = Max( K1->Height, K->Height ) + 1;
    return  ③_____;
}
```

III. Please write or draw your answers for the following problems on the answer sheet. (48 points)

(1) show the result of inserting 42 into the given AVL tree, using the algorithm discussed in class. What is the average successful search length with equal probability for the resulting tree? (6 points)





(2) Given a B-tree of order 3. Please show the results of a. inserting keys 2, 12, 16, 17, 18, (10 points) and b. deleting keys 1 and 15 from the **original** B-tree (***before insertion***). (5 points)

(3) Given a text string of length 23: OH-THIS-IS-SOOOO-GOOOOD

a. How many bits will it take to store the string as one-byte characters? (1 point)

b. How many bits will it take to store the string with standard encoding (that is, with equal-length codes), despite the space taken to store the code table?(1 point)

c. Please give the Huffman codes for the 8 distinct characters. What is the minimum length of the encoded text? (6 points)

(4) Please **briefly** describe the algorithm of deleting the minimum key from a binomial queue in O(logN) time. (4 points)

(5) Given the partial computation table of the optimal binary search tree for key words: **and**, **but**, **can**, **do**, **else**, **if**, **ok**. What is the root and the minimum-cost of the optimal subtree for key words "else-if-ok"? Please show the way you obtain your answers. (7 points)

and..and		but..but		can..can		do..do		else.. else		if..if		ok..ok	
0.22	and	0.18	but	0.20	can	0.05	do	0.25	else	0.02	if	0.08	ok
and.. but		but..can		can..do		do..else		else..if		if.. ok			
0.58	and	0.56	can	0.30	can	0.35	else	0.29	else	0.12	ok		
								else..ok					
								?	?				

(6) To compute the product of four matrices M_1 M_2 M_3 M_4, where the dimensions of the matrices are: M_1:10×20, M_2:20×5, M_3:5×40, M_4:40×6, please find the number of multiplications when following

a. the sequential computing from left to right, that is, $(((M_1\ M_2)\ M_3)\ M_4)$;

b. the greedy strategy of "compute the cheapest product first";

c. the optimal ordering such that this number is minimized.

Please specify the **orders** of computing the product by greedy and optimal methods, respectively. (8 points)

IV. Given the adjacency matrix A of an unweighted graph with N vertices. Please write a C function to compute the connectivity matrix C of all pairs of vertices. That is, after calling the function, C[i][j] = 1 if there exists a *path* between vertices i and j, and C[i][j] = 0 otherwise. Notice that A[i][i] is presumed to be zero but C[i][i] is 1, and your algorithm must have a time complexity no more than O(N^3).

(12 points)

```
void Connectivity( TwoDimArray A, TwoDimArray C, int N)
```