

6 [10 marks for the whole question]

The following program computes the minimum and maximum from array $a[1 \dots n]$ of integer. Procedure `minmax` computes those from $a[i \dots j]$. The method is based on divide-and-conquer. It selects the minimum and maximum each from the left half and right half, and select the minimum of the two minima and maximum of the two maxima. The parameter with “var” is to show call-by-variable, meaning that the value of the actual parameter can be changed. Assume n is a power of 2. All variables take integer values. We call the contents of “a” keys.

```

procedure minmax(i, j, var min, var max);
var m, min1, max1, min2, max2;
begin
  if i < j-1 then begin /* size of a[i .. j] is greater than 2 */
    m := (i + j) div 2; /* division with truncation */
    minmax(i, m, min1, max1); minmax(m+1, j, min2, max2);
    if min1 < min2 then min:=min1 else min:=min2;
    if max1 > max2 then max:=max1 else max:=max2
  end
  else /* size is 2 */ if a[i]<a[j] then begin min:=a[i]; max:=a[j] end
    else begin min:=a[j]; max:=a[i] end
end;
begin /* main program */
  /* initialization of n and a, n ≥ 2 */
  minmax(1, n, min, max);
  output(min, max)
end.

```

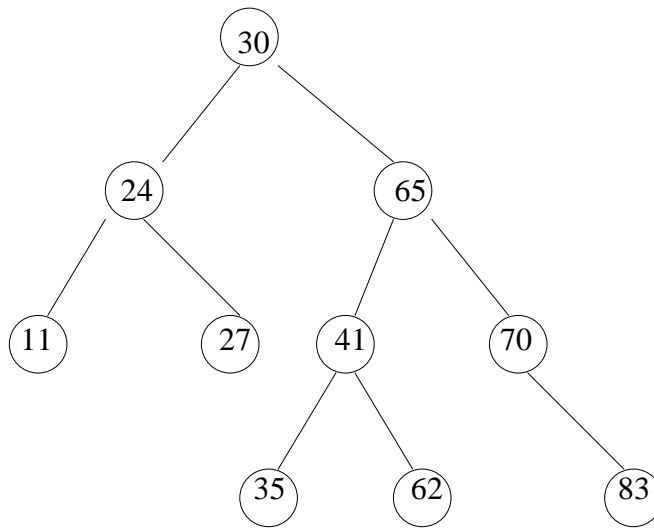
(a) [2 marks] Let $T(n)$ be the number of key comparisons. Write a recurrence equation for $T(n)$. Assume $n \geq 2$. Hint. $T(2)=1$.

(b) [4 marks] Prove that $T(n) = (3/2)n - 2$ by induction.

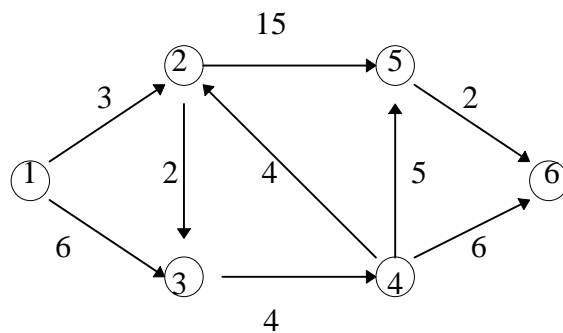
(c) [4 marks] If we use a primitive method of choosing minimum with $n-1$ comparisons by scanning the array, deleting it, and choosing maximum with $n-2$ comparisons, the total number of comparisons becomes $2n - 3$. What is the merit of the above program over this primitive method?

TURN OVER

7. [10 marks for the whole question] Insert 32 into the following AVL tree and rotate it if necessary. Show the subtrees A, B, etc., and the labels L, R, and E at the nodes, indicating left, right, and equal, before and after insertion.



8 [18 marks for the whole question] Consider the following graph.

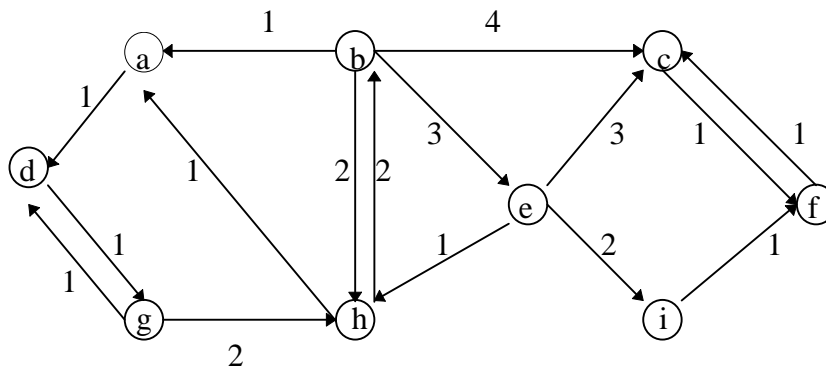


(a) [8 marks] Trace Dijkstra’s algorithm with this graph. Show the contents of array “d”, the solution set, and the frontier set at each stage.

(b) [10 marks] Trace Floyd’s algorithm with this graph. Confirm that the first row of the resulting matrix is consistent with the result in (a).

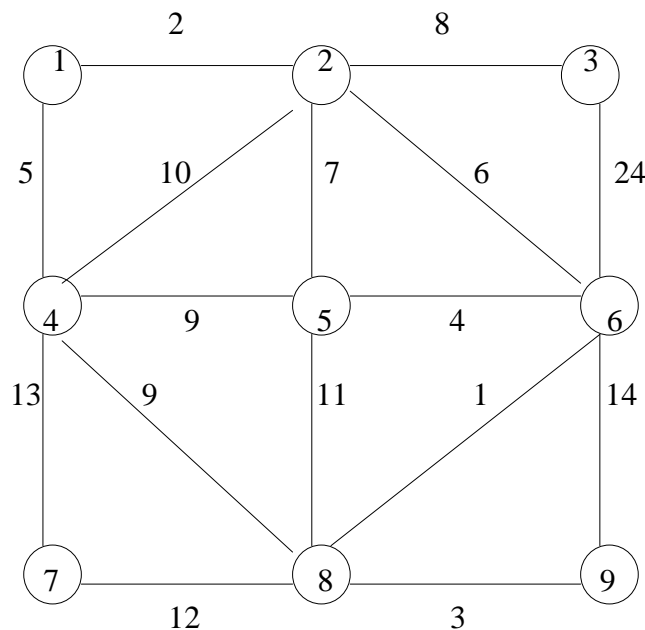
TURN OVER

9. [10 marks for the whole question] Trace Tarjan’s algorithm for strongly connected components in page 36 of the notes with the following graph and starting vertex of b.



For the trace, draw the depth-first spanning tree (forest), attach the DFS visit numbers and LOWLINK numbers to the nodes, and list up strongly connected components. The numbers attached to edges tell you the order in which you explore edges from a vertex. For example, $OUT(b) = \{a, h, e, c\}$, and you explore edges (b, a) , (b, h) , (b, e) , and (b, c) in this order. Note that we need “push v into STACK” at the beginning of $CONNECT(v)$ in Tarjan’s algorithm.

10 [10 marks for the whole question] Trace Kruskal’s algorithm for the minimum cost spanning tree with the following graph. Show the contents of array “s” at each stage.



END OF PAPER