<table>
<thead>
<tr>
<th>Author</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>G.R. Finnie</td>
<td>Modeller: A DSS for Experimental Study of Human-Computer Interaction</td>
<td>1</td>
</tr>
<tr>
<td>S.W. Postma</td>
<td>New CS Syllabus at Natal (Pmb)</td>
<td>7</td>
</tr>
<tr>
<td>C.C. Handley</td>
<td>Finding All Primes Less than Some Maximum</td>
<td>15</td>
</tr>
<tr>
<td>A-K. Heng</td>
<td>Some Remarks on Deleting a Node from a Binary Tree</td>
<td>21</td>
</tr>
<tr>
<td>A-K. Heng</td>
<td>A Summation Problem</td>
<td>23</td>
</tr>
<tr>
<td>S. Berman</td>
<td>A Persistent Programming Language - Preliminary Report</td>
<td>25</td>
</tr>
</tbody>
</table>

**BOOK REVIEWS**

14
33

An official publication of the Computer Society of South Africa and of the South African Institute of Computer Scientists

'\nAmptelike tydskrif van die Rekenaarvereniging van Suid-Afrika en van die Suid-Afrikaanse Instituut van Rekenaarwetenskaplikes
QUAESTIONES INFORMATICAE

An official publication of the Computer Society of South Africa and of the South African Institute of Computer Scientists

'n Amptelike tydskrif van die Rekenaarvereniging van Suid-Africa en van die Suid-Afrikaanse Instituut van Rekenaarwetenskaplikes

Editor

Professor G. Wiechers
INFOPLAN
Private Bag 3002
Monument Park 0105

Editorial Advisory Board

Professor D.W. Barron
Department of Mathematics
The University
Southampton S09 5NH, England

Professor J.M. Bishop
Department of Computer Science
University of the Witwatersrand
1 Jans Smuts Avenue
Johannesburg, 2001

Professor K. MacGregor
Department of Computer Science
University of Cape Town
Private Bag
Rondebosch, 7700

Dr H. Messerschmidt
IBM South Africa
P.O. Box 1419
Johannesburg, 2000

Dr P.C. Pirow
Graduate School of Business Admin.
University of the Witwatersrand
P.O. Box 31170, Braamfontein, 2017

Mr P.P. Roets
NRIMS
CSIR
P.O. Box 395
Pertoria, 0001

Professor S.H. von Solms
Department of Computer Science
Rand Afrikaans University
Auckland Park
Johannesburg, 2001

Professor M.H. Williams
Department of Computer Science
Herriot-Watt University, Edinburgh
Scotland

Subscriptions

Annual subscription are as follows:

<table>
<thead>
<tr>
<th></th>
<th>SA</th>
<th>US</th>
<th>UK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Individuals</td>
<td>R10</td>
<td>$7</td>
<td>£5</td>
</tr>
<tr>
<td>Institutions</td>
<td>R15</td>
<td>$14</td>
<td>£10</td>
</tr>
</tbody>
</table>

Quaestiones Informaticae is prepared by the Computer Science Department of the University of the Witwatersrand and printed by Printed Matter, for the Computer Society of South Africa and the South African Institute of Computer Scientists.
SOME REMARKS ON DELETING A NODE FROM A BINARY TREE

HENG Aik-Koan
Dept of Information Systems and Computer Science
National University of Singapore
Lower Kent Ridge Road, Singapore 0511

To delete an arbitrary node from a lexically ordered binary tree, the algorithm is given as follows:

1. Determine the parent node of the node marked for deletion, if it exists, note that it will not exist if we are deleting the root node.
2. If the node being deleted has either a left or a right empty subtree, then append this nonempty subtree to its grandparent node (that is, the node found in STEP 1) and EXIT.
3. Obtain the inorder successor of the node to be deleted. Append the right subtree of this successor node to its grandparent. Replace the node to be deleted by its inorder successor. This is accomplished by appending the left and right subtrees of the node marked for deletion to the successor node. Also, the successor node is appended to the parent of the node just deleted (that is, the node obtained in STEP 1).

For example, given the following binary tree:

```
      50
     /  \
   40  60
      / \
     55 70
    /\   /\   /\  
   54 56 65 80 85 90
```

If we wish to delete node 60, according to the algorithm, we should replace 60 by the inorder successor which is 65.

But if our purpose is to obtain just a lexically ordered tree, then the replacement of this deleting node could be any node on its right subtree i.e. 65, 70, 80, 85 and 90 in our example.

If we consider the number of changes in the deletion, using the algorithm, the deleting node is replaced by the inorder successor node, the left and right subtrees of the deleting node will be appended to this successor node, and the parent of this inorder successor node has now an empty left subtree. The resulting tree becomes, in our example, as follows:
But if we replace the deleting node by the root node of its right subtree. Then the left subtree of the deleting node will be appended to the left subtree of the replacing node. In our example, the resulting tree becomes:

![Tree Diagram]

1 replacement + 1 change

In the second case, the height of the resulting tree could be modified and become more unbalanced, all these will depend on the structure of the left subtree of the deleting node, which is a disadvantage.

REFERENCE
