Program 4.23: Read in a sparse matrix

```
matrixPointer mread(void)
{/* read in a matrix and set up its linked representation.
    An auxiliary global array hdnode is used */
  int numRows, numCols,  numTerms, numHeads, i;
  int row, col, value, currentRow;
  matrixPointer temp,last,node;

  printf("Enter the number of rows, columns
                        and number of nonzero terms: ");
  scanf("%d%d%d",&numRows, &numCols, &numTerms);
  numHeads = (numCols > numRows) ? numCols : numRows;
  /* set up header node for the list of header nodes */
  node = newNode(); node→tag = entry;
  node→u.entry.row = numRows;
  node→u.entry.col = numCols;

  if (!numHeads) node→right = node;
  else { /* initialize the header nodes */
    for (i = 0; i < numHeads; i++) {
      temp = newNode;
      hdnode[i] = temp; hdnode[i]→tag = head;
      hdnode[i]→right = temp;  hdnode[i]→u.next = temp;
    }
    currentRow = 0;
    last = hdnode[0]; /* last node in current row */
    for (i = 0; i < numTerms; i++) {
      printf("Enter row, column and value: ");
      scanf("%d%d%d",&row,&col,&value);
      if (row > currentRow) {/* close current row */
        last→right = hdnode[currentRow];
        currentRow = row; last = hdnode[row];
      }
      MALLOC(temp, sizeof(*temp));
      temp→tag = entry;  temp→u.entry.row = row;
      temp→u.entry.col = col;
      temp→u.entry.value = value;
      last→right = temp; /* link into row list */
      last = temp;
      /* link into column list */
      hdnode[col]→u.next→down = temp;
      hdnode[col]→u.next = temp;
    }
    /*close last row */
    last→right = hdnode[currentRow];
    /* close all column lists */
    for (i = 0; i < numCols; i++)
      hdnode[i]→u.next→down = hdnode[i];
    /* link all header nodes together */
    for (i = 0; i < numHeads-1; i++)
      hdnode[i]→u.next = hdnode[i+1];
    hdnode[numHeads-1]→u.next =  node;
    node→right = hdnode[0];
  }
  return node;
}
```

Program 4.24: Write out a sparse matrix

```
void mwrite(matrixPointer node)
{/* print out the matrix in row major form */
   int i;
   matrixPointer temp, head = node→right;
   /* matrix dimensions */
   printf(" \n numRows = %d, numCols = %d \n",
                  node→u.entry.row, node→u.entry.col);
   printf(" The matrix by row, column, and value: \n\n");
   for (i = 0; i < node→u.entry.row; i++) {
   /* print out the entries in each row */
      for (temp = head→right; temp != head;
                                    temp = temp→right)
         printf("%5d%5d%5d \n",temp→u.entry.row,
               temp→u.entry.col, temp→u.entry.value);
      head = head→u.next; /* next row */
   }
}
```

Program 4.25: Erase a sparse matrix

```
void merase(matrixPointer *node)
/* erase the matrix, return the nodes to the heap */
   matrixPointer x,y, head = (*node)→right;
   int i;
   /* free the entry and header nodes by row */
   for (i = 0; i < (*node)→u.entry.row; i++) {
      y = head→right;
      while (y != head) {
         x = y; y = y→right; free(x);
      }
      x = head; head = head→u.next; free(x);
   }
   /* free remaining header nodes*/
   y = head;
   while (y != *node) {
      x = y; y = y→u.next; free(x);
   }
   free(*node); *node = NULL;
}
```