```
double A[1:n,1:n], LU[1:n,1:n];  # assume A initialized
int ps[1:n];                     # pivot row indices
double pivot; int pivotRow;      # pivot value and row
double mult; int t;              # temporaries

# initialize ps and LU
for [i = 1 to n] {
  ps[i] = i;
  for [j = 1 to n]
    LU[i,j] = A[i,j];
}

# perform Gaussian elimination with partial pivoting
for [k = 1 to n-1] {     # iterate down main diagonal
  pivot = abs(LU[ps[k],k]); pivotRow = k;
  for [i = k+1 to n] {   # select pivot in column k
    if (abs(LU[ps[i],k]) > pivot) {
      pivot = abs(LU[ps[i],k]); pivotRow = i;
    }
  }
  if (pivotRow != k) {   # swap rows by swapping indices
    t = ps[k]; ps[k] = ps[pivotRow]; ps[pivotRow] = t;
  }
  pivot = LU[ps[k],k];      # get actual value of pivot
  for [i = k+1 to n] {      # for all rows in submatrix
    mult = LU[ps[i],k]/pivot;  # calculate multiplier
    LU[ps[i],k] = mult;        #   and save it
    for [j = k+1 to n]      # eliminate across columns
      LU[ps[i],j] = LU[ps[i],j] - mult*LU[ps[k],j];
  }
}
```

Figure 11.16 Sequential program for LU decomposition of a matrix.