```
(*Arnoldi Method*)

(*Create small matrix as proof of concept*)
A = {{1, 0, 0, 0}, {0, 2, 0, 0}, {0, 0, 3, 0}, {0, 0, 0, 4}};
Q = ConstantArray[0, {Length[A], Length[A]}];
b = {{1}, {1}, {1}, {1}};
h = ConstantArray[0, {Length[A] + 1, Length[A] + 1}];
s = ConstantArray[0, Length[A] + 1];

(*initialization*)
q1 = b / Norm[b];

(*arnoldi process of normalization*)
For[n = 1, n ≤ 4, n++,
  t = A.qn;
  For[j = 1, j ≤ n, j++,
   h[[j, n]] = Transpose[qj].t;
   s = h[[j, n]];
   r = s[[1]];
   t = t - r[[1]] * qj;
   h[[j, n]] = r[[1]];
  ];
  h[[n + 1, n]] = N[Norm[t]]; (*note: computing norm non numerically is very costly
    and exceeds precision limitations of mathematica on personal desktop*)
  qn+1 = t / h[[n + 1, n]];
 ];

(*making big H matrix, where H is upper Hessenberg?*)
H = Take[h, {1, Length[A]}, {1, Length[A]}];
displayH = Chop[N[MatrixForm[H]]];

(*making big Q matrix*)
For[p = 1, p ≤ Length[A], p++,
  For[u = 1, u ≤ 4, u++,
    Q[[u, p]] = qp[[u]][[1]];
   ];
 ];

(*the matrices we're working with...*)
Print[MatrixForm[A]];
Print[MatrixForm[b]];
Print[MatrixForm[Q]];
Print[displayH];

(*AQ=QH aka arnoldi factorization, if true, then it works*)
Print[MatrixForm[A.Q]];
Print[MatrixForm[Q.H]];
```