```
(*Gauss-Seidel Method*)

(*We can extend this to whatever size matrix we
 want. Will use a small 10x10 matrix as proof of concept*)
size = 10;

(*create matrix A*)
A = ConstantArray[0, {size, size}];

min = 1;
max = 9;

(*creating a strictly positive matrix*)
For[ i = 1, i ≤ size, i++,
  For[j = 1, j ≤ size, j++,
    If[i ≤ j, A[[i, j]] = RandomInteger[{min, max}]];
   ];
 ];

For[ i = 1, i ≤ size, i++,
  For[j = 1, j ≤ size, j++,
    If[i ≥  j, A[[i, j]] = A[[j, i]];
     ];
   ];
 ];

(*adding up the columsn of each row to force diuagonal dominance*)
sums = Total[A];

(*enforcing diagonal dominance on matrix A*)
For[i = 1, i ≤ size, i++,
 A[[i, i]] = RandomInteger[{sums[[i]], sums[[i]] + 1}];
]

b = ConstantArray[0, {Length[A], 1}];

(*making the vector b to be <1,2,3,4,5,6,7,8,9,10>*)
For[i = 1, i ≤ size, i++,
  b[[i]] = i;
 ];

(*creating L and U to be zero matrices*)
U = ConstantArray[0, {size, size}];
L = ConstantArray[0, {size, size}];

(* making L matrix the lower half of A,
including the diagonal and U matrix the upper half of A*)
For[i = 1, i ≤ size, i++,
  For[j = 1, j ≤ size, j++,
   If[i ≥ j,
    L[[i, j]] = A[[i, j]],
```

```
      U[[i, j]] = A[[i, j]];
     ]
    ]
   ];

  (*choose x1 = vector of all 1's beccause, why not... *)
  x1 = ConstantArray[1, {size, 1}];

  (*looking at the structure of the matrices and vectors*)
  MatrixForm[A]
  MatrixForm[U]
  MatrixForm[L]
  MatrixForm[b]

  (*iterative step of GS method*)
  For[i = 1, i ≤ 500, i++,
    xi+1 = Inverse[L].(b - U.xi);
    last = xi+1
   ];

  xreal = LinearSolve[A, b]; (*since we chose a small system, this is the actual *)
  xbar = last; (*the 500's iteration of the iterated x*)
  b1 = A.xbar; (*creating vector b1, which should approximate vector b*)
  precision = 30; (*going to numerically round to the nearest 30th decimal*)
  Print[MatrixForm[N[b1, precision]]]; (*if b1 ~ b, then we're correct*)
```