```
(*Conjugate Gradient Method*)

PosDefSymMat = Import["Desktop/PosDefSymMatrix.dat"]; (*data*)
A = PosDefSymMat; (*renaming PosDefSymMat to A*)
b = ConstantArray[0, {Length[A], 1}]; (*creating b vector = alternating 1 and 0*)
For[k = 1, k ≤ Length[A], k = k + 2,
  b[[k, 1]] = 1;
 ];

x0 = ConstantArray[1, Length[A]]; (*initial x0 = all 1's*)
r0 = b - A.x0; (*setting initial residual*)
P0 = r0; (*setting whatever p is to r*)
a0 = ((Transpose[r0]).r0) / (Transpose[P0].A.P0); (*creating initial constant*)
α0 = a0[[1]][[1]];

test = Transpose[r0].r0; (*i do this to see when to stop iteration*)
resvalue = test[[1]][[1]]; (*if resvalue = 0, then we stop iteration*)

i = 1;

While[resvalue ≠ 0, (*starting while loop by checking if r = 0*)
  xi = N[xi-1 + αi-1 * Pi-1]; (*loops and creates x(i)*)
  ri = ri-1 - αi-1 * A.Pi-1; (*loops and creates r(i)*)

  test = Transpose[ri].ri; (*loops to create the r constant*)
  resvalue = test[[1]][[1]]; (*accessing the r constant*)

  If[resvalue ≠ 0, (*if r isnt = 0,
   then we process, otherwise we divide by 0 and that is no good*)

   bi-1 = (Transpose[ri].ri) / (Transpose[ri-1].ri-1);
   (*creating stuff to plug into our new P(i)'s*)
   βi-1 = bi-1[[1]][[1]]; (*accessing the new variables we just made*)
   Pi = N[ri + βi-1 * Pi-1]; (*creating new P(i)'s*)
   ai = ((Transpose[ri]).ri) / (Transpose[Pi].A.Pi); (*creating new constants*)
   αi = ai[[1]][[1]], (*accessing the new constants*)

   (*the ELSE statement starts here, so if r DOES equal 0, then we terminate the loop*)

   Print[StringForm["x is obtained in "], i, StringForm[" iterations"]];
   (*show x and the number of iterations it took*)

   Break[];

  ];

  i++;

 ];
```

```
(***** REMOVE COMMENT RESTRAINTS BELOW TO DISPLAY
 ANSWER [WARNING IT'S A 1000 BY 1 VECTOR, SO ITS HUGE]*****)

(*Print[StringForm["x = "],MatrixForm[Chop[x_i]]];*)
```

x is obtained in 82 iterations