```
124  void cg_test() {
125
126      int i,j, counter0=0;
127      // remove warning:
128      counter0=counter0;
129      double **A;
130      double *u;
131      double *v;
132
133      find_N();
134
135      A = malloc(sizeof(double*)*N);
136      for(i=0;i<N;i++)
137          A[i] = malloc(sizeof(double)*N);
138
139      // initialise A to all zeroes. (sparse)
140      for(i = 0; i<N; i++)
141          for(j = 0; j<N; j++)
142              A[i][j] = 0.0;
143
144      read_A(A);
145
146      v = malloc(sizeof(double)*N);
147      read_v(v);
148
149      u = malloc(sizeof(double)*N);
150      //initialise guess of u:
151      for(i=0;i<N;i++)
152          u[i] = 0.0;
153
154      // start 'heavy lifting'
155
156      int k = 0; // iteration number
157
158      double *r;
159      r = malloc(N * sizeof(double));
160      mv(A,u,r);
161      neg(r);
162      add(v,r,r);
163
164      double rho, rho_old, beta, alpha, gamma;
165
166      rho_old = 0; //kill warning
167
168      rho = ip(r,r);
169
170      double *p; p = malloc(N*sizeof(double));
171      double *w; w = malloc(N*sizeof(double));
172
173      while(sqrt(rho) > EPS * sqrt(ip(v,v)) &&
174              k < K_MAX)
175      {
176          if(k==0) {
177              copy(r,p);
178          } else {
179              beta = rho/rho_old;
180              scale(beta, p, p);
181              add(r,p,p);
182          }
183          mv(A,p,w);
184          gamma = ip(p,w);
185          alpha = rho/gamma;
186
187          scale(alpha, p, p);
188          add(u,p,u);
189
190          neg(w);
191          scale(alpha,w,w);
192          add(r,w,r);
193          rho_old = rho;
194          rho = ip(r,r);
195          k++;
196
```

```
197       }
198
199       printf("after %d iterations, my answer is:\n", k);
200       for(i=0;i<N;i++)
201           printf("u[%d] = %lf\n", i, u[i]);
202
203       printf("-> Filling in gives:\n");
204       mv(A,u,p);
205       for(i=0;i<N;i++)
206           printf("A.u[%d] = %lf \t orig_v[%d]=%lf\n", i, p[i], i, v[i]);
207
208       printf("\nFinal error=%lf\n", rho);
209       free(p);
210       free(w);
211       free(r);
212       free(u);
213       free(v);
214       for(i=0;i<N;i++)
215           free(A[i]);
216       free(A);
217
218   }
```