```
// **********



// GSL_Eigensystems.C

// Demonstriert die Verwendung von GSL am Beispiel eines Eigenwertproblems.



// **********



#include <math.h>
#include <stdio.h>
#include <stdlib.h>

#include <gsl/gsl_blas.h>
#include <gsl/gsl_eigen.h>



// **********



int main(int argc, char **argv)
{
  int i1;

  const int n = 2;  // Matrix ist n x n.


  // *****


  // Die Matrix deren Eigenwerte berechnet werden (hier die erste
  // Pauli-Matrix).

  gsl_matrix *A = gsl_matrix_alloc(n, n);

  gsl_matrix_set(A, 0, 0,  0.0);
  gsl_matrix_set(A, 0, 1, +1.0);
  gsl_matrix_set(A, 1, 0, +1.0);
  gsl_matrix_set(A, 1, 1,  0.0);

  // Hier werden am Ende die Eigenwerte gespeichert.
  gsl_vector *lambda = gsl_vector_alloc(n);


  // *****


  // Temporärer Speicherplatz für Eigenwertberechnung.
  gsl_eigen_symm_workspace *workspace = gsl_eigen_symm_alloc(n);

  // Der Aufruf der GSL-Eigenwert-Funktion (symmetric bidiagonalization,
  // QR reduction).
  gsl_eigen_symm(A, lambda, workspace);

  // Ausgabe der berechneten Eigenwerte.
  for(i1 = 0; i1 < n; i1 ++)
    fprintf(stderr, "\\lambda_%d = %+.6lf\n", i1, gsl_vector_get(lambda, i1));

  // Speicher freigeben.
  gsl_eigen_symm_free(workspace);


  // *****


  // Speicher freigeben.
  gsl_matrix_free(A);
  gsl_vector_free(lambda);


  // *****


  return EXIT_SUCCESS;
}



// **********



// Ausgabe des Programms:
//
// \lambda_0 = +1.000000
// \lambda_1 = -1.000000



// **********



// Beim Kompilieren muss GSL eingebunden werden ... wie das geht hängt vom
// System und vom Compiler ab ... könnte z.B. so aussehen:
//
// g++ -o GSL_Eigensystems GSL_Eigensystems.C -lm -lgsl -lgslcblas



// **********
```