```
// Fig. 6.22: fig06_22.c
// Two-dimensional array manipulations.
#include <stdio.h>
#define STUDENTS 3
#define EXAMS 4

// function prototypes
int minimum(const int grades[][EXAMS], size_t pupils, size_t tests);
int maximum(const int grades[][EXAMS], size_t pupils, size_t tests);
double average(const int setOfGrades[], size_t tests);
void printArray(const int grades[][EXAMS], size_t pupils, size_t tests);

// function main begins program execution
int main(void)
{
   // initialize student grades for three students (rows)
   int studentGrades[STUDENTS][EXAMS] =
      { { 77, 68, 86, 73 },
        { 96, 87, 89, 78 },
        { 70, 90, 86, 81 } };

   // output array studentGrades
   puts("The array is:");
   printArray(studentGrades, STUDENTS, EXAMS);

   // determine smallest and largest grade values
   printf("\n\nLowest grade: %d\nHighest grade: %d\n",
      minimum(studentGrades, STUDENTS, EXAMS),
      maximum(studentGrades, STUDENTS, EXAMS));

   // calculate average grade for each student
   for (size_t student = 0; student < STUDENTS; ++student) {
      printf("The average grade for student %u is %.2f\n",
         student, average(studentGrades[student], EXAMS));
   }
}

// Find the minimum grade
int minimum(const int grades[][EXAMS], size_t pupils, size_t tests)
{
   int lowGrade = 100; // initialize to highest possible grade

   // loop through rows of grades
   for (size_t i = 0; i < pupils; ++i) {

      // loop through columns of grades
      for (size_t j = 0; j < tests; ++j) {
```

```
         if (grades[i][j] < lowGrade) {
            lowGrade = grades[i][j];
         }
      }
   }

   return lowGrade; // return minimum grade
}

// Find the maximum grade
int maximum(const int grades[][EXAMS], size_t pupils, size_t tests)
{
   int highGrade = 0; // initialize to lowest possible grade

   // loop through rows of grades
   for (size_t i = 0; i < pupils; ++i) {

      // loop through columns of grades
      for (size_t j = 0; j < tests; ++j) {

         if (grades[i][j] > highGrade) {
            highGrade = grades[i][j];
         }
      }
   }

   return highGrade; // return maximum grade
}

// Determine the average grade for a particular student
double average(const int setOfGrades[], size_t tests)
{
   int total = 0; // sum of test grades

   // total all grades for one student
   for (size_t i = 0; i < tests; ++i) {
      total += setOfGrades[i];
   }

   return (double) total / tests; // average
}

// Print the array
void printArray(const int grades[][EXAMS], size_t pupils, size_t tests)
{
   // output column heads
   printf("%s", "                [0]  [1]  [2]  [3]");
```

```
   // output grades in tabular format
   for (size_t i = 0; i < pupils; ++i) {

      // output label for row
      printf("\nstudentGrades[%u] ", i);

      // output grades for one student
      for (size_t j = 0; j < tests; ++j) {
         printf("%-5d", grades[i][j]);
      }
   }
}

/**************************************************************************
 * (C) Copyright 1992-2015 by Deitel & Associates, Inc. and               *
 * Pearson Education, Inc. All Rights Reserved.                           *
 *                                                                        *
 * DISCLAIMER: The authors and publisher of this book have used their     *
 * best efforts in preparing the book. These efforts include the          *
 * development, research, and testing of the theories and programs        *
 * to determine their effectiveness. The authors and publisher make       *
 * no warranty of any kind, expressed or implied, with regard to these    *
 * programs or to the documentation contained in these books. The authors *
 * and publisher shall not be liable in any event for incidental or       *
 * consequential damages in connection with, or arising out of, the       *
 * furnishing, performance, or use of these programs.                     *
 *************************************************************************/
```