Two-dimensional and Jagged Arrays

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

Two Dimensional Arrays

- Arrays with double indices · · · Imagine a rectangular table of data or a matrix
- To declare, use two pairs of rackets, e.g., `double[][] twoDArray = new double[8][11];`
- We might be tempted to consider this as `(double[])[]`, but in reality the order is reversed
- If only the first index is fixed, it can be treated a one-dimensional array, e.g., for each integer, say j (within the range), `twoDArray[j]` is a one-dimensional `double` array

Read Data from File

- Read data from a file into a two-dimensional array
- The first two tokens in the file are the dimensions of the array
- Then the data elements follow, row-wise
- The file name is given as `args[ 0 ]`

The Code: TwoDData.java

```
import java.util.*;
import java.io.*;

public class TwoDData {
 public static void main( String[] args )
   throws FileNotFoundException {

  Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
  int nRows = fileScanner.nextInt();
  int nCols = fileScanner.nextInt();
  double[][] data = new double[ nRows ][ nCols ];
  for ( int i = 0; i < nRows; i ++ ) {
   for ( int j = 0; j < nCols; j ++ ) {
    data[ i ][ j ] = fileScanner.nextDouble();
   }
  }
```

The main method throws `FileNotFoundException` when the file specified in `args[ 0 ]` does not exist

The Code: TwoDData.java

```java
import java.util.*;
import java.io.*;

public class TwoDData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    int nCols = fileScanner.nextInt();
    double[][] data = new double[ nRows ][ nCols ];
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Create a file scanner from `args[ 0 ]`

The Code: TwoDData.java

```java
import java.util.*;
import java.io.*;

public class TwoDData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    int nCols = fileScanner.nextInt();
    double[][] data = new double[ nRows ][ nCols ];
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Read the dimensions

The Code: TwoDData.java

```java
import java.util.*;
import java.io.*;

public class TwoDData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    int nCols = fileScanner.nextInt();
    double[][] data = new double[ nRows ][ nCols ];
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Create a two-dimensional array to store data using the dimensions

The Code: TwoDData.java

```java
import java.util.*;
import java.io.*;

public class TwoDData {
 public static void main( String[] args )
   throws FileNotFoundException {

  Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
  int nRows = fileScanner.nextInt();
  int nCols = fileScanner.nextInt();
  double[][] data = new double[ nRows ][ nCols ];
  for ( int i = 0; i < nRows; i ++ ) {
   for ( int j = 0; j < nCols; j ++ ) {
    data[ i ][ j ] = fileScanner.nextDouble();
   }
  }
```

Read the data using a double for loop

The Code: TwoDData.java

```
    System.out.println( "Data has been read" );
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < nCols; j ++ ) {
        System.out.printf( "%.2f", data[ i ][ j ] );
        if ( j < nCols - 1 ) {
          System.out.print( " " );
        }
      }
      System.out.println();
    }
  }
}
```

External loop going through the row indices

The Code: TwoDData.java

```
    System.out.println( "Data has been read" );
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < nCols; j ++ ) {
        System.out.printf( "%.2f", data[ i ][ j ] );
        if ( j < nCols - 1 ) {
          System.out.print( " " );
        }
      }
      System.out.println();
    }
  }
}
```

Internal loop going through the column indices; except for the last column index print a white space

The Code: TwoDData.java

```
  System.out.println( "Data has been read" );
  for ( int i = 0; i < nRows; i ++ ) {
   for ( int j = 0; j < nCols; j ++ ) {
    System.out.printf( "%.2f", data[ i ][ j ] );
    if ( j < nCols - 1 ) {
     System.out.print( " " );
    }
   }
   System.out.println();
  }
 }
}
```

At the conclusion of each internal loop print a new line

Table of Contents

1. Two-dimensional Array
2. Jagged Array
3. Simple Application - Courses and Grades

Jagged array

A jagged array is a two-dimensional array in which the number of elements may not be equal among the rows

Jagged array

A jagged array is a two-dimensional array in which the number of elements may not be equal among the rows

For example, consider the problem of recording daily high temperatures over a year, indexed by the month and the day of month

Jagged array

A jagged array is a two-dimensional array in which the number of elements may not be equal among the rows

For example, consider the problem of recording daily high temperatures over a year, indexed by the month and the day of month

Assuming the year is not a leap year, there will be 12 rows and the number of elements in the rows are:
31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31

Jagged Array Creation

A jagged is declared as a regular two-dimensional array:

```
double[][] temperatures;
```

Jagged Array Creation

A jagged is declared as a regular two-dimensional array:

```
double[][] temperatures;
```

A jagged array should be created using only the first dimension:

```
double[][] temperatures = new double[12][];
```

Note the empty []

Jagged Array Creation

A jagged is declared as a regular two-dimensional array:

```
double[][] temperatures;
```

A jagged array should be created using only the first dimension:

```
double[][] temperatures = new double[12][];
```

Note the empty []

This means that `temperatures` is an array of `double[]` objects and its dimension is 12

Jagged Array Creation

A jagged is declared as a regular two-dimensional array:

```
double[][] temperatures;
```

A jagged array should be created using only the first dimension:

```
double[][] temperatures = new double[12][];
```

Note the empty []

This means that `temperatures` is an array of `double[]` objects and its dimension is 12

Each array should be separately created, e.g.,

```
temperatures[0] = new double[31];
```

Jagged Array Example

Write a generate purpose application for receive data from the file specified by `args[ 0 ]`

- The first entry is the number of rows
- For each row, the file specifies the number of columns in that row and then the said number of elements ensue

The Code: JaggedData.java

```java
import java.util.*;
import java.io.*;

public class JaggedData {
 public static void main( String[] args )
   throws FileNotFoundException {

  Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
  int nRows = fileScanner.nextInt();
  double[][] data = new double[ nRows ][];
  for ( int i = 0; i < nRows; i ++ ) {
   int nCols = fileScanner.nextInt();
   data[ i ] = new double[ nCols ];
   for ( int j = 0; j < nCols; j ++ ) {
    data[ i ][ j ] = fileScanner.nextDouble();
   }
  }
```

The main method throws `FileNotFoundException` when the file specified in `args[ 0 ]` does not exist

The Code: JaggedData.java

```
import java.util.*;
import java.io.*;

public class JaggedData {
 public static void main( String[] args )
   throws FileNotFoundException {

  Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
  int nRows = fileScanner.nextInt();
  double[][] data = new double[ nRows ][];
  for ( int i = 0; i < nRows; i ++ ) {
   int nCols = fileScanner.nextInt();
   data[ i ] = new double[ nCols ];
   for ( int j = 0; j < nCols; j ++ ) {
    data[ i ][ j ] = fileScanner.nextDouble();
   }
  }
```

Create a file scanner from `args[ 0 ]`

The Code: JaggedData.java

```
import java.util.*;
import java.io.*;

public class JaggedData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    double[][] data = new double[ nRows ][];
    for ( int i = 0; i < nRows; i ++ ) {
      int nCols = fileScanner.nextInt();
      data[ i ] = new double[ nCols ];
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Read the number of rows

The Code: JaggedData.java

```java
import java.util.*;
import java.io.*;

public class JaggedData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    double[][] data = new double[ nRows ][];
    for ( int i = 0; i < nRows; i ++ ) {
      int nCols = fileScanner.nextInt();
      data[ i ] = new double[ nCols ];
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Create a jagged array matching the number of rows

The Code: JaggedData.java

```java
import java.util.*;
import java.io.*;

public class JaggedData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    double[][] data = new double[ nRows ][];
    for ( int i = 0; i < nRows; i ++ ) {
      int nCols = fileScanner.nextInt();
      data[ i ] = new double[ nCols ];
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Read the data using a double for loop; the external loop goes through the row indices

The Code: JaggedData.java

```java
import java.util.*;
import java.io.*;

public class JaggedData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    double[][] data = new double[ nRows ][];
    for ( int i = 0; i < nRows; i ++ ) {
      int nCols = fileScanner.nextInt();
      data[ i ] = new double[ nCols ];
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Read the number of columns in the row and then creates the row for that row index

The Code: JaggedData.java

```java
import java.util.*;
import java.io.*;

public class JaggedData {
  public static void main( String[] args )
      throws FileNotFoundException {

    Scanner fileScanner = new Scanner( new File( args[ 0 ] ) );
    int nRows = fileScanner.nextInt();
    double[][] data = new double[ nRows ][];
    for ( int i = 0; i < nRows; i ++ ) {
      int nCols = fileScanner.nextInt();
      data[ i ] = new double[ nCols ];
      for ( int j = 0; j < nCols; j ++ ) {
        data[ i ][ j ] = fileScanner.nextDouble();
      }
    }
```

Using an internal loop to read the data

The Code: TwoDData.java

```
  System.out.println( "Data has been read" );
  for ( int i = 0; i < nRows; i ++ ) {
   for ( int j = 0; j < data[ i ].length; j ++ ) {
    System.out.printf( "%.2f", data[ i ][ j ] );
    if ( j < data[ i ].length - 1 ) {
     System.out.print( " " );
    }
   }
   System.out.println();
  }
 }
}
```

External loop going through the row indices

The Code: TwoDData.java

```java
    System.out.println( "Data has been read" );
    for ( int i = 0; i < nRows; i ++ ) {
      for ( int j = 0; j < data[ i ].length; j ++ ) {
        System.out.printf( "%.2f", data[ i ][ j ] );
        if ( j < data[ i ].length - 1 ) {
          System.out.print( " " );
        }
      }
      System.out.println();
    }
  }
}
```

Internal loop going through the column indices; the last index is `data[ i ].length - 1`

The Code: TwoDData.java

```java
  System.out.println( "Data has been read" );
  for ( int i = 0; i < nRows; i ++ ) {
   for ( int j = 0; j < data[ i ].length; j ++ ) {
    System.out.printf( "%.2f", data[ i ][ j ] );
    if ( j < data[ i ].length - 1 ) {
     System.out.print( " " );
    }
   }
   System.out.println();
  }
 }
}
```

At the conclusion of each internal loop print a new line

Table of Contents

Maintaining Course List by Semester and Their Grades

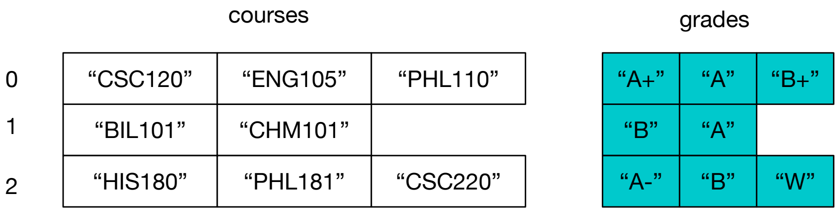
Write a simple application for course records

- Receive data on course numbers and grades by semester
- Compute GPA
- Search for course grades
- Make changes

Idea

- Use two jagged arrays of an idential shape, one for course numbers and the other for letter grades
- The number of rows is the number of semesters
- For each row the number of data elements is the number of courses
- For each user enter number and grade for all the courses

The Idea



courses

0	“CSC120”	“ENG105”	“PHL110”
1	“BIL101”	“CHM101”	
2	“HIS180”	“PHL181”	“CSC220”

grades

“A+”	“A”	“B+”
“B”	“A”	
“A-”	“B”	“W”

User Input

CSC120 A+ ENG105 A PHL110 B+
BIL101 B CHM101 A
HIS180 A- PHL181 B CSC220 W

Grades.java: Grade Conversion

```
  public static double convert( String letter ) {
    switch ( letter ) {
    case "A":
    case "A+": return 4.0;
    case "A-": return 3.7;
    case "B+": return 3.4;
    case "B": return 3.0;
    case "B-": return 2.7;
    case "C+": return 2.4;
    case "C": return 2.0;
    case "C-": return 1.7;
    case "D+": return 1.4;
    case "D": return 1.0;
    case "D-": return 0.7;
    case "F": return 0.0;
    default: return -1.0;
    }
  }
```

Use a switch statement to convert a letter to a grade point value

Grades.java: Grade Conversion

```
  public static double convert( String letter ) {
    switch ( letter ) {
    case "A":
    case "A+": return 4.0;
    case "A-": return 3.7;
    case "B+": return 3.4;
    case "B": return 3.0;
    case "B-": return 2.7;
    case "C+": return 2.4;
    case "C": return 2.0;
    case "C-": return 1.7;
    case "D+": return 1.4;
    case "D": return 1.0;
    case "D-": return 0.7;
    case "F": return 0.0;
    default: return -1.0;
    }
  }
```

If there is a match return the corresponding value

Grades.java: Grade Conversion

```
  public static double convert( String letter ) {
    switch ( letter ) {
    case "A":
    case "A+": return 4.0;
    case "A-": return 3.7;
    case "B+": return 3.4;
    case "B": return 3.0;
    case "B-": return 2.7;
    case "C+": return 2.4;
    case "C": return 2.0;
    case "C-": return 1.7;
    case "D+": return 1.4;
    case "D": return 1.0;
    case "D-": return 0.7;
    case "F": return 0.0;
    default: return -1.0;
    }
  }
```

Otherwise, return a negative value

Grades.java: Find All Grades Matching a Query

```
public static void find( String query,
    String[][] courses, String[][] letters ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].indexOf( query ) >= 0 ) {
        System.out.printf( "semester=%d, course=%s, grade=%s%n",
          s, courses[ s ][ pos ], letters[ s ][ pos ] );
      }
    }
  }
}
```

`query` is the pattern to look for
`courses` is the jagged array of course numbers
`leters` is the jagged array of grades

Grades.java: Find All Grades Matching a Query

```
public static void find( String query,
    String[][] courses, String[][] letters ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].indexOf( query ) >= 0 ) {
        System.out.printf( "semester=%d, course=%s, grade=%s%n",
          s, courses[ s ][ pos ], letters[ s ][ pos ] );
      }
    }
  }
}
```

Generate all possible index value pairs `(s, pos)`
Note the maxium value for each row is given by `courses[ s ].length`

Grades.java: Find All Grades Matching a Query

```java
public static void find( String query,
    String[][] courses, String[][] letters ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].indexOf( query ) >= 0 ) {
        System.out.printf( "semester=%d, course=%s, grade=%s%n",
          s, courses[ s ][ pos ], letters[ s ][ pos ] );
      }
    }
  }
}
```

If there is a match (by way of indexOf), print the semester, course number, and the grade

Grades.java: GPA calculation

```
  public static void gpa( String[][] courses, String[][] letters ) {
    int count = 0;
    double sum = 0;
    for ( int s = 0; s < courses.length; s ++ ) {
      for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
        double value = convert( letters[ s ][ pos ] );
        if ( value >= 0 ) {
          count ++;
          sum += value;
        }
      }
    }
    if ( count > 0 ) {
      sum /= count;
    }
    System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
  }
```

`courses` is the jagged array of course numbers

`leters` is the jagged array of grades

Grades.java: GPA calculation

```java
public static void gpa( String[][] courses, String[][] letters ) {
  int count = 0;
  double sum = 0;
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      double value = convert( letters[ s ][ pos ] );
      if ( value >= 0 ) {
        count ++;
        sum += value;
      }
    }
  }
  if ( count > 0 ) {
    sum /= count;
  }
  System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
}
```

Use `count` to compute the number of courses that count towards gpa; use `sum` to compute the total grade point value

Grades.java: GPA calculation

```
  public static void gpa( String[][] courses, String[][] letters ) {
    int count = 0;
    double sum = 0;
    for ( int s = 0; s < courses.length; s ++ ) {
      for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
        double value = convert( letters[ s ][ pos ] );
        if ( value >= 0 ) {
          count ++;
          sum += value;
        }
      }
    }
    if ( count > 0 ) {
      sum /= count;
    }
    System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
  }
```

Try all index position pairs

Grades.java: GPA calculation

```
  public static void gpa( String[][] courses, String[][] letters ) {
    int count = 0;
    double sum = 0;
    for ( int s = 0; s < courses.length; s ++ ) {
      for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
        double value = convert( letters[ s ][ pos ] );
        if ( value >= 0 ) {
          count ++;
          sum += value;
        }
      }
    }
    if ( count > 0 ) {
      sum /= count;
    }
    System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
  }
```

Obtain the grade point value of the course

Grades.java: GPA calculation

```java
public static void gpa( String[][] courses, String[][] letters ) {
  int count = 0;
  double sum = 0;
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      double value = convert( letters[ s ][ pos ] );
      if ( value >= 0 ) {
        count ++;
        sum += value;
      }
    }
  }
  if ( count > 0 ) {
    sum /= count;
  }
  System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
}
```

If the value is nonnegative, it counts; update the count and the total

Grades.java: GPA calculation

```java
  public static void gpa( String[][] courses, String[][] letters ) {
    int count = 0;
    double sum = 0;
    for ( int s = 0; s < courses.length; s ++ ) {
      for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
        double value = convert( letters[ s ][ pos ] );
        if ( value >= 0 ) {
          count ++;
          sum += value;
        }
      }
    }
    if ( count > 0 ) {
      sum /= count;
    }
    System.out.printf( "#courses=%d, gpa=%.3f%n", count, sum );
  }
```

Compute the average (division occurs only if the count is positive) and report the average

Grades.java: Make a Change

```java
public static void change( String[][] courses, String[][] letters,
    String c, String g ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].equals( c ) ) {
        letters[ s ][ pos ] = g;
        return;
      }
    }
  }
}
```

`courses` and `letters` are as before

`c` and `g` are the course to make a change and the new grade, respectively

Grades.java: Make a Change

```
public static void change( String[][] courses, String[][] letters,
    String c, String g ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].equals( c ) ) {
        letters[ s ][ pos ] = g;
        return;
      }
    }
  }
}
```

Try all index value pairs

Grades.java: Make a Change

```
public static void change( String[][] courses, String[][] letters,
    String c, String g ) {
  for ( int s = 0; s < courses.length; s ++ ) {
    for ( int pos = 0; pos < courses[ s ].length; pos ++ ) {
      if ( courses[ s ][ pos ].equals( c ) ) {
        letters[ s ][ pos ] = g;
        return;
      }
    }
  }
}
```

If the course number matches, make the change and return

Grades.java: Interaction

```
public static void interact( String[][] courses, String[][] letters ) {
 Scanner console = new Scanner( System.in );
 String response;
 do {
  System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
  response = console.nextLine();
  switch ( response.charAt( 0 ) ) {
  case ’f’:
   System.out.print( "Enter key: " );
   String query = console.nextLine();
   find( query, courses, letters );
   break;
  case ’g’: gpa( courses, letters );
   break;
  case ’c’:
   System.out.print( "Enter course: " );
   String c = console.nextLine();
   System.out.print( "Enter new grade: " );
   String g = console.nextLine();
   change( courses, letters, c, g );
  }
 } while ( !response.startsWith( "q" ) );
}
```

`courses` and `letters` are as before

Grades.java: Interaction

```
public static void interact( String[][] courses, String[][] letters ) {
  Scanner console = new Scanner( System.in );
  String response;
  do {
    System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
    response = console.nextLine();
    switch ( response.charAt( 0 ) ) {
    case 'f':
      System.out.print( "Enter key: " );
      String query = console.nextLine();
      find( query, courses, letters );
      break;
    case 'g': gpa( courses, letters );
      break;
    case 'c':
      System.out.print( "Enter course: " );
      String c = console.nextLine();
      System.out.print( "Enter new grade: " );
      String g = console.nextLine();
      change( courses, letters, c, g );
    }
  } while ( !response.startsWith( "q" ) );
}
```

`console` as before; `response` is the user response

Grades.java: Interaction

```
public static void interact( String[][] courses, String[][] letters ) {
 Scanner console = new Scanner( System.in );
 String response;
 do {
  System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
  response = console.nextLine();
  switch ( response.charAt( 0 ) ) {
  case ’f’:
   System.out.print( "Enter key: " );
   String query = console.nextLine();
   find( query, courses, letters );
   break;
  case ’g’: gpa( courses, letters );
   break;
  case ’c’:
   System.out.print( "Enter course: " );
   String c = console.nextLine();
   System.out.print( "Enter new grade: " );
   String g = console.nextLine();
   change( courses, letters, c, g );
  }
 } while ( !response.startsWith( "q" ) );
}
```

The do-while loop executes until the user enters ’q’

Grades.java: Interaction

```
  public static void interact( String[][] courses, String[][] letters ) {
    Scanner console = new Scanner( System.in );
    String response;
    do {
      System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
      response = console.nextLine();
      switch ( response.charAt( 0 ) ) {
      case 'f':
        System.out.print( "Enter key: " );
        String query = console.nextLine();
        find( query, courses, letters );
        break;
      case 'g': gpa( courses, letters );
        break;
      case 'c':
        System.out.print( "Enter course: " );
        String c = console.nextLine();
        System.out.print( "Enter new grade: " );
        String g = console.nextLine();
        change( courses, letters, c, g );
      }
    } while ( !response.startsWith( "q" ) );
  }
```

Prompt the user to receive response

Grades.java: Interaction

```
public static void interact( String[][] courses, String[][] letters ) {
  Scanner console = new Scanner( System.in );
  String response;
  do {
    System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
    response = console.nextLine();
    switch ( response.charAt( 0 ) ) {
    case 'f':
      System.out.print( "Enter key: " );
      String query = console.nextLine();
      find( query, courses, letters );
      break;
    case 'g': gpa( courses, letters );
      break;
    case 'c':
      System.out.print( "Enter course: " );
      String c = console.nextLine();
      System.out.print( "Enter new grade: " );
      String g = console.nextLine();
      change( courses, letters, c, g );
    }
  } while ( !response.startsWith( "q" ) );
}
```

Use a switch to select action to perform

Grades.java: Interaction

```java
  public static void interact( String[][] courses, String[][] letters ) {
    Scanner console = new Scanner( System.in );
    String response;
    do {
      System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
      response = console.nextLine();
      switch ( response.charAt( 0 ) ) {
      case 'f':
        System.out.print( "Enter key: " );
        String query = console.nextLine();
        find( query, courses, letters );
        break;
      case 'g': gpa( courses, letters );
        break;
      case 'c':
        System.out.print( "Enter course: " );
        String c = console.nextLine();
        System.out.print( "Enter new grade: " );
        String g = console.nextLine();
        change( courses, letters, c, g );
      }
    } while ( !response.startsWith( "q" ) );
  }
```

For “find”, receive user from the query and then call the find methd

Grades.java: Interaction

```java
  public static void interact( String[][] courses, String[][] letters ) {
    Scanner console = new Scanner( System.in );
    String response;
    do {
      System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
      response = console.nextLine();
      switch ( response.charAt( 0 ) ) {
      case 'f':
        System.out.print( "Enter key: " );
        String query = console.nextLine();
        find( query, courses, letters );
        break;
      case 'g': gpa( courses, letters );
        break;
      case 'c':
        System.out.print( "Enter course: " );
        String c = console.nextLine();
        System.out.print( "Enter new grade: " );
        String g = console.nextLine();
        change( courses, letters, c, g );
      }
    } while ( !response.startsWith( "q" ) );
  }
```

For “gpa” call the gpa methd

Grades.java: Interaction

```java
public static void interact( String[][] courses, String[][] letters ) {
  Scanner console = new Scanner( System.in );
  String response;
  do {
    System.out.print( "(f)ind, (c)hange, (g)pa, (q)uit " );
    response = console.nextLine();
    switch ( response.charAt( 0 ) ) {
    case 'f':
      System.out.print( "Enter key: " );
      String query = console.nextLine();
      find( query, courses, letters );
      break;
    case 'g': gpa( courses, letters );
      break;
    case 'c':
      System.out.print( "Enter course: " );
      String c = console.nextLine();
      System.out.print( "Enter new grade: " );
      String g = console.nextLine();
      change( courses, letters, c, g );
    }
  } while ( !response.startsWith( "q" ) );
}
```

For “change” call the change method after receiving the course number and the new grade

Grades.java: Main

```java
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Receive the number of semesters

Grades.java: Main

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Create the jagged arrays

Grades.java: Main

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Read data for each semester

Grades.java: Main

```
public static void main( String[] args ) {
 Scanner console = new Scanner( System.in );
 System.out.print( "Enter the number of semesters: " );
 int noSemester = Integer.parseInt( console.nextLine() );
 String[][] courses = new String[ noSemester ][];
 String[][] letters = new String[ noSemester ][];
 for ( int i = 0; i < noSemester; i ++ ) {
  System.out.print( "Course and grade list sem. " + i + " > ");
  String w = console.nextLine();
  String[] parts = w.split( " " );
  int count = parts.length / 2;
  courses[ i ] = new String[ count ];
  letters[ i ] = new String[ count ];
  for ( int pos = 0; pos < count; pos ++ ) {
   courses[ i ][ pos ] = parts[ 2 * pos ];
   letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
  }
 }
 interact( courses, letters );
}
```

Receive from the user a line with data for the semester

Grades.java: Main

```java
public static void main( String[] args ) {
 Scanner console = new Scanner( System.in );
 System.out.print( "Enter the number of semesters: " );
 int noSemester = Integer.parseInt( console.nextLine() );
 String[][] courses = new String[ noSemester ][];
 String[][] letters = new String[ noSemester ][];
 for ( int i = 0; i < noSemester; i ++ ) {
  System.out.print( "Course and grade list sem. " + i + " > ");
  String w = console.nextLine();
  String[] parts = w.split( " " );
  int count = parts.length / 2;
  courses[ i ] = new String[ count ];
  letters[ i ] = new String[ count ];
  for ( int pos = 0; pos < count; pos ++ ) {
   courses[ i ][ pos ] = parts[ 2 * pos ];
   letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
  }
 }
 interact( courses, letters );
}
```

The split method splits the string into string array using " " as the delimiter

Grades.java: Main

```java
public static void main( String[] args ) {
  Scanner console = new Scanner( System.in );
  System.out.print( "Enter the number of semesters: " );
  int noSemester = Integer.parseInt( console.nextLine() );
  String[][] courses = new String[ noSemester ][];
  String[][] letters = new String[ noSemester ][];
  for ( int i = 0; i < noSemester; i ++ ) {
    System.out.print( "Course and grade list sem. " + i + " > ");
    String w = console.nextLine();
    String[] parts = w.split( " " );
    int count = parts.length / 2;
    courses[ i ] = new String[ count ];
    letters[ i ] = new String[ count ];
    for ( int pos = 0; pos < count; pos ++ ) {
      courses[ i ][ pos ] = parts[ 2 * pos ];
      letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
    }
  }
  interact( courses, letters );
}
```

One half of the array length is the number of courses

Grades.java: Main

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Create the array for the semester for both jagged arrays

Grades.java: Main

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Copy all even indexed values to course numbers and the rewst to course grades

Grades.java: Main

```java
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter the number of semesters: " );
    int noSemester = Integer.parseInt( console.nextLine() );
    String[][] courses = new String[ noSemester ][];
    String[][] letters = new String[ noSemester ][];
    for ( int i = 0; i < noSemester; i ++ ) {
      System.out.print( "Course and grade list sem. " + i + " > ");
      String w = console.nextLine();
      String[] parts = w.split( " " );
      int count = parts.length / 2;
      courses[ i ] = new String[ count ];
      letters[ i ] = new String[ count ];
      for ( int pos = 0; pos < count; pos ++ ) {
        courses[ i ][ pos ] = parts[ 2 * pos ];
        letters[ i ][ pos ] = parts[ 2 * pos + 1 ];
      }
    }
    interact( courses, letters );
  }
```

Call the interact method

The End