Sometimes it’s useful to store related data in two or more arrays. For example, assume a program uses the following arrays:

```java
String[] names = new String[5];
String[] addresses = new String[5];
```

The `names` array is used to store the names of five persons, and the `addresses` array is used to store the addresses of the same five persons. The data for one person is stored in the same relative location in each array. For example, the first person’s name is stored at `names[0]` and that same person’s address is stored at `addresses[0]`. This relationship between the arrays is illustrated in Figure CS4-1.

**Figure CS4-1** The relationship between the names and addresses arrays

![Diagram showing the relationship between names and addresses arrays](image-url)
To access the data, use the same subscript with both arrays. For example, the following loop displays each person’s name and address.

```java
for (int i = 0; i < names.length; i++)
{
    System.out.println("Name: " + names[i]);
    System.out.println("Address: " + addresses[i]);
}
```

The names and addresses arrays are examples of parallel arrays. Parallel arrays are two or more arrays that hold related data, and the related elements of each array are accessed with a common subscript.

Parallel arrays are especially useful when the related data is of unlike types. For example, the program in Code Listing CS4-1 is a payroll program. It uses two arrays: one to store the hours worked by each employee (as int values), and another to store each employee’s hourly pay rate (as double values).

Case Study 4
Parallel Arrays

**Code Listing CS4-1** *(ParallelArrays.java)*

```java
1 import java.util.Scanner;
2 import java.text.DecimalFormat;
3
4 /**
5    This program demonstrates parallel arrays.
6 */
7
8 public class ParallelArrays
9 {
10    public static void main(String [] args)
11    {
12        final int NUM_EMPLOYEES = 3;
13        int[] hours = new int[NUM_EMPLOYEES];
14        double[] payRates = new double[NUM_EMPLOYEES];
15
16        // Get the hours worked by each employee.
17        getPayData(hours, payRates);
18
19        // Display each employee's gross pay.
20        displayGrossPay(hours, payRates);
21    }
22
23    /**
24        The getPayData method accepts as arguments arrays for employees' hours and pay rates. The user enters values for these arrays.
25     */
26
27    }
28```
private static void getPayData(int[] hours, double[] payRates)
{
    // Create a Scanner object for keyboard input.
    Scanner keyboard = new Scanner(System.in);

    // Get each employee's hours worked and pay rate.
    for (int i = 0; i < hours.length; i++)
    {
        // Get the hours worked for this employee.
        System.out.print("Enter the hours worked by " +
                         "employee #" + (i + 1) +
                         ": ");
        hours[i] = keyboard.nextInt();

        // Get the hourly pay rate for this employee.
        System.out.print("Enter the hourly pay rate for " +
                         "employee #" + (i + 1) +
                         ": ");
        payRates[i] = keyboard.nextDouble();
    }
}

/**
   * The displayGrossPay method accepts as arguments
   * arrays for employees' hours and pay rates. The
   * method uses these arrays to calculate and display
   * each employee's gross pay.
   */

private static void displayGrossPay(int[] hours,
                                    double[] payRates)
{
    double grossPay; // To hold gross pay

    // Create a DecimalFormat object.
    DecimalFormat dollar = new DecimalFormat("#,##0.00");

    // Display each employee's gross pay.
    for (int i = 0; i < hours.length; i++)
    {
        // Calculate the gross pay.
        grossPay = hours[i] * payRates[i];

        // Display the gross pay.
        System.out.println("The gross pay for " +
                           "employee #" + (i + 1) +
                           " is $" +
                           dollar.format(grossPay));
    }
}
Program Output with Example Input Shown in Bold

Enter the hours worked by employee #1: 15 [Enter]
Enter the hourly pay rate for employee #1: 12.00 [Enter]
Enter the hours worked by employee #2: 20 [Enter]
Enter the hourly pay rate for employee #2: 18.00 [Enter]
Enter the hours worked by employee #3: 40 [Enter]
Enter the hourly pay rate for employee #3: 25.50 [Enter]
The gross pay for employee #1 is $180.00
The gross pay for employee #2 is $360.00
The gross pay for employee #3 is $1,020.00