حل مسائل ساختمان داده‌ها با جاوا

تألیف:

دکتر رمضان عباس نژاد ورزی

فن‌آوری نوین

سرشناسه	:	**عباس‌نژاد ورزی، رمضان، ۱۳۴۸ –**
عنوان و نام پدیدآور	:	**حل مسائل ساختمان داده‌ها با جاوا/ تالیف رمضان عباس‌نژادورزی.**
مشخصات نشر	:	**بابل: فناوری نوین، ۱۳۹۸.**
مشخصات ظاهری	:	**۲۸۸ص.: مصور، جدول.**
شابک	:	**۵۰۰۰۰۰** ریال :**۹۷۸-۶۰۰-۷۲۷۲-۳۸-۱**
وضعیت فهرست نویسی	:	**فیپا**
موضوع	:	**جاوا (زبان برنامه‌نویسی کامپیوتر)**
موضوع	:	Java (Computer program language)
موضوع	:	**ساختار داده‌ها –– مسائل، تمرین‌ها و غیره**
موضوع	:	Data structures (Computer science) -- Problems, exercises, etc.
موضوع	:	**الگوریتم‌های کامپیوتری –– مسائل، تمرین‌ها و غیره**
موضوع	:	Computer algorithms -- Problems, exercises, etc.
رده بندی کنگره	:	**QA۷۶/۷۳**
رده بندی دیویی	:	**۰۰۵/۱۳۳**
شماره کتابشناسی ملی	:	**۵۹۲۱۶۰۷**

www.fanavarienovin.net

فن‌آوری نوین بابل، کد پستی ۴۷۱۶۷-۷۳۴۴۸ تلفن: ۳۲۲۵۶۶۸۷-۰۱۱

حل مسائل ساختمان داده‌ها با جاوا

تألیف: رمضان عباس نژاد ورزی

نوبت چاپ: چاپ اول

سال چاپ: پاییز ۱۳۹۸

شمارگان: ۲۰۰

قیمت: ۵۰۰۰۰ تومان

نام چاپخانه و صحافی: دفتر فنی سورنا

شابک: ۱-۳۸-۷۲۷۲-۶۰۰-۹۷۸

نشانی ناشر: بابل، چهارراه نواب، کاظم بیگی، جنب مسجد منصور کاظم بیگی، طبقه اول

طراح جلد: کانون آگهی و تبلیغات آبان (احمد فرجی)

تهران، خ اردیبهشت، نبش وحید نظری، پلاک ۱۴۲ تلفکس: ۶۶۴۰۰۱۴۴-۶۶۴۰۰۲۲۰

فهرست مطالب

مقدمه

ساختمان داده، یکی از مباحث بسیار مهم در دروس مهندسی کامپیوتر، فناوری اطلاعات و علوم کامپیوتر است. کتاب‌های زیادی درزمینه ساختمان داده با زبان جاوا ترجمه و تألیف شده است که جای تقدیر و تشکر دارد. اما، جای کتاب حل مسائل ساختمان داده با جاوا که بتواند الگوریتم‌های مختلف ساختمان داده را پیاده-سازی کند، خالی است. هدف این کتاب آموزش جاوا نیست. اما، اگر بخواهید جاوا را آموزش ببینید، می-توانید کتاب‌های آموزش **گام‌به‌گام برنامه‌نویسی جاوا** و **حل مسائل** جاوا از همین انتشارات را مطالعه کنید.

کتاب حاضر الگوریتم‌های مختلف ساختمان داده را به زبان جاوا پیاده‌سازی می‌کند. این کتاب شامل ۷ فصل است که عبارت‌اند از:

فصل اول، تمرین‌های مربوط به مباحثی نظیر **ساختار داده‌ها و الگوریتم‌های بازگشتی** را حل کرده است.

فصل دوم، مسائل مباحث مربوط به **ماتریس‌های یک‌بعدی، دوبعدی و ماتریس‌های اسپارس** را پیاده-سازی کرده است.

فصل سوم، تمرین‌های مباحثی مانند **صف‌ها و پشته‌ها** را حل نموده است.

فصل چهارم، مسائل **لیست‌های پیوندی** را پیاده‌سازی می‌کند.

فصل پنجم، تمرین‌های **درخت‌ها و کاربردهای آن‌ها** را حل نمود.

فصل ششم، مسائل **گراف‌ها و کاربردهای آن‌ها** را پیاده‌سازی کرده است.

فصل هفتم، مسائل **جست‌وجو و مرتب‌سازی** را پیاده‌سازی کرده است.

از تمامی اساتید و دانشجویان عزیز تقاضا داریم، هرگونه اشکال، ابهام در متن کتاب، پیشنهاد و انتقادات را به آدرس پست الکترونیک **fanavarienovin@gmail.com** ارسال نمایند.

در پایان امیدوارم این اثر موردتوجه جامعه انفورماتیک کشور، اساتید و دانشجویان عزیز قرار گیرد.

مؤلف

fanavarienovin@gmail.com

فصل ۱

ساختار داده‌ها و الگوریتم‌های بازگشتی

مثال ۱– ۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت می‌کند و از n تا یک را نمایش می‌دهد:

```
package ch1_1;
import java.util.Scanner;
public class Ch1_1 {
    private static void display(int n) {
       if (n == 0) {
          return;
       }
       else  {
          System.out.print(n);
          System.out.print("\t");
          display(n - 1);
       }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       display(n);
     }
}
```

```
Enter n:١٠
١٠    ٩    ٨    ٧    ٦    ٥    ٤    ٣    ٢    ١
```

مثال ۲–۱. تابع بازگشتی که مجموع عناصر یک آرایه را برمی‌گرداند.

```
package ch1_2;
import java.util.Scanner;
public class Ch1_2 {
    private static int listSum(int[] numList, int n)  {
       if (n == 0)  {
          return 0;
       }
       else  {
          return numList[n - 1] + listSum(numList, n - 1);
       }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       int [] a =new int[n];
       System.out.print("Enter ");
       System.out.print(n);
       System.out.print(" numbers:");
       for (int i = 0; i < n; i++) {
         a[i] = in.nextInt();
       }
        System.out.print(listSum(a, n));
    }
}
```

```
Enter n:٥
```

```
Enter ۵ numbers:۱۰ ۲۰ ۳۰ ٤۰ ۵۰
۱۵۰
```

مثال ۳–۱. تابع بازگشتی که حاصل عبارت زیر را محاسبه می‌کند و برمی‌گرداند.

```
package ch1_3;
import java.util.Scanner;
public class Ch1_3 {
    private static float harmonicSum(int n)  {
        if (n < 2) {
           return 1F;
        }
        else  {
           return (1.0F / n + harmonicSum(n - 1));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(harmonicSum(n));
    }
}
```

```
Enter n:۱۰
۲,۹۲۸۹٦۸٤
```

مثال ۴–۱. تابع بازگشتی که بزرگ‌ترین مقسوم‌علیه مشترک بین دو عدد را برمی‌گرداند.

```
package ch1_4;
import java.util.Scanner;
public class Ch1_4 {
    private static int min(int a, int b)  {
      return (a < b  ? a : b);
   }
   private static int max(int a, int b)  {
       return (a > b ? a : b);
   }
   private static int gcd(int a, int b)  {
       int low = min(a, b);
       int high = max(a, b);
       if (low == 0)  {
          return high;
       }
       else if (low == 1) {
          return 1;
       }
       else {
          return gcd(low, high % low);
       }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter a, b:");
       int a = in.nextInt();
       int b = in.nextInt();
       System.out.print(gcd(a, b));
    }
}
```

```
Enter a, b:۹۰ ۲۰
۱۰
```

مثال ۵–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، مجموع ارقام آن را برمی‌گرداند.

```
package ch1_5;
import java.util.Scanner;
public class Ch1_5 {
    private static int  sumDigit(int n) {
    if (n == 0)
        return 0;
    else
        return (n % 10 + sumDigit(n / 10));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(sumDigit(n));
    }
}
```

```
Enter n:۳٤۱۲٥
۱٥
```

مثال ۶–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، حاصل‌ضرب ارقام فرد آن را برمی‌گرداند.

```
package ch1_6;
import java.util.Scanner;
public class Ch1_6 {
    private static int mulOddDigit(int n)   {
        if (n == 0)  {
            return 1;
        }
        else if (n % 10 % 2 == 1)  {
            return (n % 10 * mulOddDigit((int)(n / 10)));
        }
        else  {
            return (mulOddDigit(n / 10));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
        System.out.print(mulOddDigit(n));
   }
}
```

```
Enter n:٥٦٨٨۰۹
٤٥
```

مثال ۷–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، حاصل جمع ارقام بالای ۶ آن را برمی‌گرداند.

```
package ch1_7;
import java.util.Scanner;
public class Ch1_7 {
    private static int sumDigitG6(int n)   {
        if (n == 0)   {
            return 0;
        }
        else if (n % 10 > 6)   {
            return (n % 10 + sumDigitG6(n / 10));
        }
        else    {
            return (sumDigitG6((int)(n / 10)));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(sumDigitG6(n));
    }
}
```

```
Enter n:٦٧٣٤٨
١٥
```

مثال ۸–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، nامین جمله فیبوناچی را برمی‌گرداند.

```
package ch1_8;
import java.util.Scanner;
public class Ch1_8 {
    private static int Fibo(int n)   {
        if (n == 1 || n == 2)   {
            return 1;
        }
        else   {
            return (Fibo(n - 1) + Fibo(n - 2));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(Fibo(n));
    }
}
```

```
Enter n:١٠
٥٥
```

مثال ۹–۱. تابع بازگشتی که m و n را به‌عنوان پارامتر دریافت کرده، $\binom{n}{m} = C(n,\ m)$ را برمی‌گرداند.

```
package ch1_9;
import java.util.Scanner;
public class Ch1_9 {
```

```
    private static int C(int n, int m)  {
        if (n == m || m == 0)  {
            return 1;
        }
        else  {
            return (C(n - 1, m - 1) + C(n - 1, m));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n, m:");
       int n = in.nextInt();
       int m = in.nextInt();
       System.out.print(C(n, m));
    }
}
```

```
Enter n, m:۱۰ ۷
۱۲۰
```

مثال ۱۰–۱. تابع بازگشتی که a و n را به‌عنوان پارامتر دریافت کرده، حاصل عبارت $\sqrt{a+\sqrt{a+\sqrt{a+\cdots}}}$ را برای n مرتبه برمی‌گرداند.

```
package ch1_10;
import java.util.Scanner;
public class Ch1_10 {
    private static double S(double a, double n) {
        if (n == 1)  {
            return Math.sqrt(a);
        }
        else  {
            return (Math.sqrt(a + S(a, n - 1)));
        }
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter a, n:");
        double a = in.nextDouble();
        double n = in.nextDouble();
        System.out.print(S(a, n));
    }
}
```

```
Enter a, n:۱۰۰ ۵۰
۱۰,۵۱۲۴۹۲۱۹۷۲۵۰۳۹٤
```

مثال ۱۱–۱. تابع بازگشتی که a و b را به‌عنوان پارامتر دریافت کرده، a تقسیم بر b را برمی‌گرداند.

```
package ch1_11;
import java.util.Scanner;
public class Ch1_11 {
    private static int Div(int a, int b)  {
        if (a < b)   {
            return 0;
```

```
        }
        else   {
            return (1 + Div(a - b, b));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter a, b:");
       int a = in.nextInt();
       int b = in.nextInt();
       System.out.print(Div(a, b));
    }
}
```

```
Enter a, b:۹ ۲
٤
```

مثال ۱۲–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، حاصل جمع اعداد فرد ۱ تـا n را برمـی‌گرداند.

```
package ch1_12;
import java.util.Scanner;
public class Ch1_12 {
    private static int sumOdd (int n) {
        if (n < 1)
            return 0;
        else if (n % 2 == 1)
            return (n + sumOdd(n - 2));
        else
            return (sumOdd(n - 1));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(sumOdd(n));
    }
}
```

```
Enter n:٤٥
٥٢٩
```

مثال ۱۳–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، حاصل‌ضرب اعداد زوج ۲ تا n را برمـی‌گرداند.

```
package ch1_13;
import java.util.Scanner;
public class Ch1_13 {
    private static long  MulEven(int n) {
        if (n < 2)
            return 1;
        else if (n % 2 == 0)
            return (n * MulEven(n - 2));
```

```
        else
            return (MulEven(n - 1));
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter n:");
        int n = in.nextInt();
        System.out.print(MulEven(n));
    }
}
```

```
Enter n:۱۷
۱۰۳۲۱۹۲۰
```

مثال ۱۴–۱. تابع بازگشتی که اعداد صحیح a و b بزرگ‌تر از صفر را دریافت کرده، a ضرب در b را برمی‌گرداند.

```
package ch1_14;
import java.util.Scanner;
public class Ch1_14 {
    private static int  Mul(int a, int b) {
        if (b == 1)
            return a;
        else
            return (a + Mul(a, b - 1));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter a, b:");
       int a = in.nextInt();
       int b = in.nextInt();
       System.out.print(Mul(a, b));
    }
}
```

```
Enter a, b:۱۰ ۸
۸۰
```

مثال ۱۵–۱. تابع بازگشتی که a و b (اعداد صحیح) را به‌عنوان پارامتر دریافت کرده، باقی‌مانده تقسیم صحیح a بر b را برمی‌گرداند.

```
package ch1_15;
import java.util.Scanner;
public class Ch1_15 {
    private static int  Mod(int a, int b) {
        if (a < b)
            return a;
        else
            return(Mod(a - b, b));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter a, b:");
       int a = in.nextInt();
       int b = in.nextInt();
       System.out.print(Mod(a, b));
```

```
    }
}
```

```
Enter a, b:٥٤ ١٠
٤
```

مثال ۱۶–۱. تابع بازگشتی که k و n را به‌عنوان پارامتر دریافت کرده، مجموع تمام اعداد مضرب k کوچک تر یا مساوی n را برمی‌گرداند.

```
package ch1_15;
import java.util.Scanner;
public class Ch1_15 {
    private static int  Mod(int a, int b) {
        if (a < b)
            return a;
        else
            return(Mod(a - b, b));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter a, b:");
       int a = in.nextInt();
       int b = in.nextInt();
       System.out.print(Mod(a, b));
    }
}
```

```
Enter a, b:٥٤ ١٠
٤
```

مثال ۱۷–۱. تابع بازگشتی که n و k را به‌عنوان پارامتر دریافت کرده، تعداد ارقام k آن را برمی‌گرداند.

```
package ch1_17;
import java.util.Scanner;
public class Ch1_17 {
    private static int countK(int n, int k) {
        if (n == 0)
            return 0;
        else if ( n % 10 == k)
            return (1 + countK(n /  10, k));
        else
            return (countK(n / 10, k));
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n, k:");
       int n = in.nextInt();
       int k = in.nextInt();
       System.out.print(countK(n, k));
    }
}
```

```
Enter n, k:٢٢٢٣٢٤٢٥ ٢
٥
```

مثال ۱۸–۱. تابع بازگشتی که تعداد تکرار یک مقدار در یک آرایه را برمی‌گرداند.

```
package ch1_18;
import java.util.Scanner;
public class Ch1_18 {
    private static int listCount(int[] numList, int n, int k) {
        if (n == 0)   {
           return 0;
        }
        else if (numList[n - 1] == k)       {
           return (1 + listCount(numList, n - 1, k));
        }
        else   {
           return (listCount(numList, n - 1, k));
        }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       int [] a =new int[n];
       System.out.print("Enter ");
       System.out.print(n);
       System.out.print(" numbers:");
       for (int i = 0; i < n; i++) {
         a[i] = in.nextInt();
       }
       System.out.print("Enter k:");
       int k = in.nextInt();
       System.out.print(listCount(a, n, k));
    }
}
```

Enter n:٦

Enter ٦ numbers:١ ٢ ١ ٣ ١ ٤

Enter k:١

٣

مثال ۱۹–۱. تابع بازگشتی که کوچک‌ترین عنصر یک آرایه را برمی‌گرداند.

```
package ch1_19;
import java.util.Scanner;
public class Ch1_19 {
    private static int findMin_min = 10000000;
    private static int findMin(int[] numList, int n)  {
       if (n == 0) {
          return findMin_min;
       }
       else if (numList[n - 1] < findMin_min)  {
          findMin_min = numList[n - 1];
          return (findMin(numList, n - 1));
       }
       else   {
          return (findMin(numList, n - 1));
       }
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter n:");
```

```
        int n = in.nextInt();
        int [] a =new int[n];
        System.out.print("Enter ");
        System.out.print(n);
        System.out.print(" numbers:");
        for (int i = 0; i < n; i++)  {
         a[i] = in.nextInt();
        }
        System.out.print(findMin(a, n));
    }
}
```

Enter n:٦

Enter ٦ numbers:٠ ٧ ٨ ٤ ٩ ١

٠

مثال ۲۰–۱. تابع بازگشتی که دو آرایه را با هم مقایسه کرده، نتیجه را برمی‌گرداند.

```
package ch1_20;
import java.util.Scanner;
public class Ch1_20 {
   private static boolean equals(int[] list1,int[] list2,int n) {
        if (n == 0) {
          return true;
        }
       else if (list1[n - 1] != list2[n - 1]) {
          return false;
       }
       else   {
          return (equals(list1, list2, n - 1));
       }
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       int [] a =new int[n];
       System.out.print("Enter ");
       System.out.print(n);
       System.out.print(" numbers:");
       for (int i = 0; i < n; i++)   {
         a[i] = in.nextInt();
       }
       int [] b =new int[n];
       System.out.print("Enter ");
       System.out.print(n);
       System.out.print(" numbers:");
       for (int i = 0; i < n; i++)  {
         b[i] = in.nextInt();
       }
       if (equals(a, b, n))  {
            System.out.print("Yes");
        }
        else  {
            System.out.print("No");
        }
    }
}
```

En ter n:٤

En ter ٤ numb ers:١ ٢ ٣ ٤

En ter ٤ numb ers:١

۲۳٤

Yes

مثال ۲۱–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، معکوس آن را برمی‌گرداند.

```
package ch1_21;
import java.util.Scanner;
public class Ch1_21 {
    private static int reverse_r = 0;
    private static int reverse(int n)  {
        if (n == 0)  {
            return 0;
        }
        reverse_r = reverse_r * 10;
        reverse_r = reverse_r + n % 10;
        reverse(n / 10);
        return reverse_r;
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
        System.out.print(reverse(n));
    }
}
```

Enter n:٦٧٨٩

٩٨٧٦

مثال ۲۲–۱. تابع بازگشتی که n را به‌عنوان پارامتر دریافت کرده، بزرگ‌ترین رقم آن را برمی‌گرداند.

```
package ch1_22;
import java.util.Scanner;
public class Ch1_22 {
    private static int maxDigit_max = -1;
    private static int maxDigit(int n)    {
        if (n == 0)  {
            return maxDigit_max;
        }
        else if (n % 10 > maxDigit_max)   {
            maxDigit_max = n % 10;
            maxDigit(n / 10);
        }
        else     {
            maxDigit(n / 10);
        }
        return maxDigit_max;
    }
    public static void main(String[] args) {
       Scanner in = new Scanner(System.in);
       System.out.print("Enter n:");
       int n = in.nextInt();
       System.out.print(maxDigit(n));
   }
}
```

Enter n:٦٥٤٧

۷

مثال ۲۳–۱. تابع بازگشتی که جست‌وجوی دودویی را در آرایه مرتب‌شده انجام می‌دهد.

```
package ch1_23;
import java.util.Scanner;
public class Ch1_23 {
    private static int BinarySearch(int[] nums, int low, int
      high, int searchedNumber)   {
       if (low == high)  {
              if (nums[low] == searchedNumber)  {
                     return low;
              }
              else  {
                     return -1;
              }
       }
       else  {
              int leftAns = BinarySearch(nums, low, (high +
                low) / 2, searchedNumber);
              int rightAns = BinarySearch(nums, (high + low) /
                2 + 1, high, searchedNumber);
              if (leftAns != -1)    {
                     return leftAns;
              }
              else if (rightAns != -1)  {
                     return rightAns;
              }
              else          {
                     return -1;
              }
       }
    }
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter n:");
        int n = in.nextInt();
        int [] a =new int[n];
        System.out.print("Enter ");
        System.out.print(n);
        System.out.print(" sorted numbers:");
        for (int i = 0; i < n; i++)   {
         a[i] = in.nextInt();
        }
       System.out.print("Enter x:");
       int x = in.nextInt();
        System.out.print(BinarySearch(a, 0, n - 1, x));
    }
}
```

Enter n:۷

Enter ۷ sorted numbers:۱ ۷ ۹ ۱۰ ۱۲ ۱٤ ۲٦

Enter x:۹

۲

فصل ۲

آرایه‌ها و ماتریس‌ها

مثال ۱ – ۲. برنامه‌ای که یک ماتریس را دریافت کرده، در جهت عقربه ساعت چرخش می‌دهد. اگر ماتریس به‌صورت زیر باشد:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

خروجی به‌صورت ماتریس زیر خواهد شد:

5	1	2	3
9	10	6	4
13	11	7	8
14	15	16	12

```
package ch2_1;
import java.util.Scanner;
public class Ch2_1 {
    static int R = 4;
    static int C = 4;
    static void rotatematrix( int mat[][], int m, int n)
    {
        int row = 0, col = 0;
        int prev, curr;
         /*
               row - Staring row index
               m - ending row index
               col - starting column index
               n - ending column index
               i - iterator
         */
         while (row < m && col < n)
         {
            if (row + 1 == m || col + 1 == n)
                break;
            // Store the first element of next
            // row, this element will replace
            // first element of current row
            prev = mat[row + 1][col];
            // Move elements of first row
            // from the remaining rows
            for (int i = col; i < n; i++)
            {
                curr = mat[row][i];
                mat[row][i] = prev;
                prev = curr;
            }
            row++;
            // Move elements of last column
            // from the remaining columns
```

```
        for (int i = row; i < m; i++)
        {
           curr = mat[i][n-1];
           mat[i][n-1] = prev;
                  prev = curr;
        }
        n--;
        // Move elements of last row
        // from the remaining rows
        if (row < m)
        {
           for (int i = n-1; i >= col; i--)
           {
                curr = mat[m-1][i];
               mat[m-1][i] = prev;
               prev = curr;
           }
        }
        m--;
        // Move elements of first column
        // from the remaining rows
        if (col < n)
        {
           for (int i = m-1; i >= row; i--)
           {
                  curr = mat[i][col];
                  mat[i][col] = prev;
                  prev = curr;
           }
        }
        col++;
      }
   }
   static void readA2D( int mat[][], int m, int n, Scanner in)
   {
        for(int i = 0; i < R; i++)
        {
            System.out.print("Enter ");
            System.out.print(C);
            System.out.print(" numbers:");
            for (int j = 0; j < C; j++)
            {
               mat[i][j] = in.nextInt();
            }
         }
    }
   static void printA2D( int mat[][], int m, int n)
   {
       for(int i = 0; i < R; i++)
       {
          for (int j = 0; j < C; j++)
          {
              System.out.print(mat[i][j]+"\t");
      }
      System.out.print("\n");
   }
}
public static void main(String[] args)
{
    Scanner in = new Scanner(System.in);
    System.out.print("Enter R, C:");
```

```
            R = in.nextInt();
            C = in.nextInt();
            int a[][] = new int[R][C];
            readA2D(a, R, C, in);
            System.out.print("Original Matrix\n");
            printA2D(a, R, C);
            rotatematrix(a, R, C);
            System.out.print("Rotated Matrix\n");
            printA2D(a, R, C);
        }
}
```

Enter R, C:۳ ۳
Enter ۳ numbers:۱ ۲ ۳
Enter ۳ numbers:٤ ٥ ٦
Enter ۳ numbers:۷ ۸ ۹
Original Matrix
۱ ۲ ۳
٤ ٥ ٦
۷ ۸ ۹
Rotated Matrix
٤ ۱ ۲
۷ ٥ ۳
۸ ۹ ٦

مثال ۲– ۲. برنامه‌ای که یک ماتریس را بدون استفاده از حافظه اضافی ۹۰ درجه چرخش می‌دهد. به‌عنوان‌مثال اگر آرایه ورودی به‌صورت زیر باشد:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

خروجی به‌صورت ماتریس زیر خواهد شد:

4	8	12	16
3	7	11	15
2	6	10	14
1	5	9	13

مراحل طراحی و اجرا:

```
package ch2_2;
import java.util.Scanner;
public class Ch2_2 {
    static int R = 4;
    static int C = 4;
    static void reverseColumns(int arr[][], int R, int C)
    {
        for (int i = 0; i < R; i++)
            for (int j = 0, k = C - 1;  j < k; j++, k--) {
                int temp = arr[j][i];
                arr[j][i] = arr[k][i];
                arr[k][i] = temp;
            }
```

```
        }
        // Function for do transpose of matrix
        static void transpose(int arr[][], int R, int C)
        {
                 for (int i = 0; i < R; i++)
                        for (int j = i; j < C; j++) {
                                int temp = arr[j][i];
                                arr[j][i] = arr[i][j];
                                arr[i][j] = temp;
                        }
        }
        // Function to anticlockwise rotate
        // matrix by 90 degree
        static void rotate90(int arr[][], int R, int C)
        {
                 transpose(arr, R, C);
          reverseColumns(arr, R, C);
        }
        static void readA2D( int mat[][], int m, int n, Scanner in)
        {
            for(int i = 0; i < R; i++)
            {
                System.out.print("Enter ");
                System.out.print(C);
                System.out.print(" numbers:");
                for (int j = 0; j < C; j++)
                {
                   mat[i][j] = in.nextInt();
                }
            }
        }
        static void printA2D( int mat[][], int m, int n)
        {
            for(int i = 0; i < R; i++)
            {
                for (int j = 0; j < C; j++)
                {
                    System.out.print(mat[i][j]+"\t");
                }
                System.out.print("\n");
            }
        }
        public static void main(String[] args) {
            Scanner in = new Scanner(System.in);
            System.out.print("Enter R, C:");
            R = in.nextInt();
            C = in.nextInt();
            int a[][] = new int[R][C];
            readA2D(a, R, C, in);
            System.out.print("Orginal Matrix:\n");
             printA2D(a, R, C);
            rotate90(a, R, C);
            System.out.print("Result  Matrix:\n");
          printA2D(a, R, C);

        }
    }
```

Enter R, C:٤ ٤
Enter ٤ numbers:١ ٢ ٣ ٤
Enter ٤ numbers:٥ ٦ ٧ ٨
Enter ٤ numbers:٩ ١٠ ١١ ١٢