ሽሬይ (Array)

6.1 ምዕላደ-ቃላት

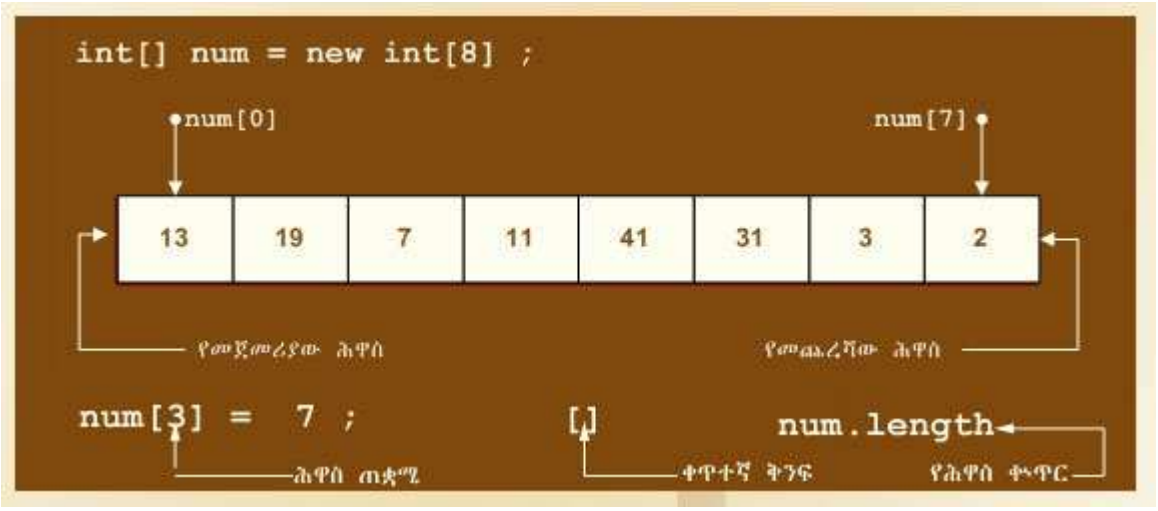
አማርኛ ቃል	እንግሊዘኛ ቃል
ሽሬይ	array
ሜሞሪ	memory
ሕዋስ	cell
ሕዋስ ጠቋሚ	index
የመደብ ርቢ	object
ታህታይ ገደብ	lower bound
ላዕላይ ገደብ	upper bound
ቀጥተኛ ቅንፍ	[]

6.2 የሽሬይ ዓይነት ሲሉ

የዲታ ቁጥር እየበዛ ሲመጣ ፥ ማጠናቀሩ ፥ ማደራጀቱ ፥ እንዲሁም ልዩ ልዩ ተግባራትን መፈጸሙ ፥ ስልታዊ የሜሞሪ አያያዝና አጠቃቀም ይጠይቃል። ሜሞሪን ከምናደራጅበት መንገዶች መካከል አንዱ ሽሬይ ነው።

ሽሬይ ስንል ፥ ተመሳሳይ የዲታ ዓይነት ያላቸው ፥ በአንድ ስም የተወከሉና ፥ በተርታ የተደራጁ የሜሞሪ ክፍሎችን ነው። ይህ አባባል ፥ የጃቫን ሽሬይ መሠረታዊ ባሕሪያት በሙሉ አይጠቀልልም ፤ ግን አጠቃላይ መንፈሱን ያንጸባርቃል። የቀረውን ወደፊት እንደርስበታለን። ለእንደ ሲ (C) ላሉት ቋንቋዎች ግን አባበሉ ከእውነት አይርቅም።

የሽሬይ ሥዕላዊ መልኩ ይኸን ይመስላል።



የአንድ ሽሬይ መሠረታዊ ባሕሪያት እነዚህ ናቸው።

- በአንድ ተውላጠ-ቃል ይወከላል ፤
- አንድ የዴታ ዓይነት ይጠብቃል ፤
- የመደብ ርባታ ነው ፤
- በተርታ የተደራጀ የሜሞሪ ክፍል አለው ፤
- እያንዳንዱ የሽሬይ ሕዋስ በተናጠል መጠቀስ አለበት ፤

የሚከተለው ምሳሌ የኢንትጀር ሽሬይ ፈጥሮ ፥ በ«ባቦ ሰጥ» የወጡ ቁጥሮችን ሞላቶ ፥ በመጨረሻ የሽሬዩን ይዘት ያትማል። የሽሬይ አወጣጥን ፥ በእያንዳንዱ ሕዋስ ውስጥ ዴታ ለማስገባት ወይም ለማስወጣት የሽሬይ አጠቃቀስን መንገድ ያሳያል።

ሽሬይ አወጣጥና አጠቃቀም

```
import java.util.* ;
public class SimpleArray {

   /** An entry point for program execution */
   public static void main(String[] args) {
      // instantiate an object of the Random class
      Random rand = new Random() ;

      // declare an integer array
      int [] num = new int[8] ;

      // fill the array with randomely generated numbers
      for (int i=0; i < num.length; i++) {
         num[i] = rand.nextInt() ;
      }

      // print the content of the array
      for (int i=0; i < num.length; i++) {
         System.out.println("num[" + i + "] = " + num[i]) ;
      }
   }
}
```

Download: SimpleArray.java

6.3 ሽሬይ አወጣጥና አሰነዳድ

በአጠቃላይ ደረጃ ፥ የሽሬይ ተውላጠ-ቃል አወጣጥ ይኸን ይመስላል። ከይሆናል (=) በስተግራ ያለው የሽሬዩን ዴታ ዓይነት እንዲሁም ጥልቀትና ስም ይገልጻል ፤ ከይሆናል (=) በስተቀኝ ፥ የሽሬዩን ቦታ ይከፍታል።

ቃሉን በዝርዝር እንመልከት።

- **int[] num**: የኧሬዩ የዴታ ዓይነት ኢንትጀር ነው ይላል። የ [] ምልክት «ኧሬይ» መሆኑን ያመለክታል። የኧሬዩ ስም «num» ነው። በማንኛውም የጃቫ ዴታ ዓይነት ላይ የተመሠረተ ኧሬይ መፍጠር ስንፈልግ መልኩ ይኸን መምሰል አለበት። ለምሳሌ፦

```
char[] alpha ;
float[] salary ;
double[] number ;
byte[] sequence ;
short[] alpha ;
```

- **=**: የይሆናል ምልክት ሲሆን በስተቀኝ በኩል የሚከፈተውን ቦታ ይሰይማል።
- **new**: የተጠየቀውን የሜሞሪ ቦታ ይከፍታል/ይደለድላል።
- **int[32]**: ተከፋቹ የዴታ ዓይነትና መጠን።

አንድ ኧሬይ ከተፈጠረ በኋላ ለሥር ዝግጁ ነው። ሕዋሶቹ ውስጥ ዴታ መጻፍ ወይም ማንባብ ይቻላል። የዴታ አጻጻፉ ይኸን ይመስላል።

```
double[] constants = new double[19] ;
constants[0] = 3.14159 ;
constants[1] = 2.71 ;
constants[2] = 1.6 ;
```

በአንድ ሕዋስ ውስጥ ያለውን ይዘት ለማወቅ የሕዋሱን ጠቋሚ መጠቀም ይጠይቃል።

```
double value = constants[0] ;
```

ወይም

```
System.out.println(constants[0]) ;
```

ማንኛውም የጃቫ ኧሬይ ተፈጥሯዊ መጠኑን በ**length** አባሉ በኩል ይጠብቃል። ሕዋሳቱን ለማየት ከመሞከራችን በፊት የኧሬዩን መጠን አለማለፋችንን ለማረጋገጥ የ**length**ን ዕሴት መመለክት አሥፈላጊ ነው።

አስቀድመን ያየናቸው ሼሬዮች በመሠረታዊ የዴታ ዓይነት ላይ የተገነቡ ናቸው። ቀጥለን የመደብ የሼሬይ ዓይነቶችን እንመለከታለን። ሼሬይ ሁልጊዜ ሼሬይ ነው። በ`String` መደብ ላይ ይሁን ወይም በ`int` ላይ ፥ በቃል አገባብና አሠራር አንድ ነው።

የሜይን መላ **(main method)** ከጠሪው የሚላከውን ዕሴት የሚቀበልበት ተውላጠ-ቃል የእስትሪንግ ሼሬይ ነው። ከላይ የተሰጠውን የSimpleArray መደብ ብንመለከት የሜይን መላ የእስትሪንግ ሼሬይ በአናቱ ላይ አውጆ ዕሴት ለመቀበል ዝግጁ ነው።

የማንኛውም የመደብ ዓይነት ሼሬይ መፍጠር እንችላለን። ለምሳሌ ፥

```
String[] names = new String[32] ;
```

ይህ ቃል 32 ሕዋሳት ያሉት የእስትሪንግ ቦታ ይከፍታል። በዚህ ሼሬይ ልንጠብቅ የምንችለው የእስትሪንግ ብዛት 32 ነው። ልንስተው የሚይገባ አንድ ነገር ቢኖር ፥ 32 ሕዋስ ያለው ሼሬይ በመፍጠራቸን 32 የእስትሪንግ ርቢ አረባን ማለት አይደለም። ቦታና ርቢ ለይቅል ናቸው።

ሼሬዩን በእስትሪንግ ርቢ ለመሙላት፦

```
names[0] = "Addis Ababa" ;
```

ወይም

```
String capital = new String("Addis Ababa") ;
names[0] = capital ;
```

የሼሬዩ የመጀመሪያ ሕዋስ ውስጥ የተሰጠውን የእስትሪንግ ርቢ ይከታል። የሁለቱም ቃሎች የመጨረሻ ውጤት አንድ ዓይነት ነው። ሁሉንም የሼሬይ ሕዋሳት በተመሳሳይ የእስትሪንግ ርቢ እንሙላ ከተባለ ይኸን ይመስላል።

```
for (int i=0; i < names.length; i++)
    names[i] = capital ;
```

እያንዳንዱን ሕዋስ ብንመረምር ተመሳሳይ ዕሴት ይኖረዋል። እናም፦

```
for (int i=0; i < names.length; i++) {
    System.out.print("name[" + i + "] = ") ;
    System.out.print(names[i]) ;
}
```

የሚከተለው ምሳሌ የአንባቢውን ትግስትና አትኩሮት ይጠይቅ ይሆናል።

መደባዊ ሼሬይ አወጣጥና አጠቃቀም

```
public class ObjectCounter {
   /** declares a static variable */
   static int count = 0 ;

   /** Constructor keeps track of object instantiation */
   public ObjectCounter () {
```

```
        count++ ;
        System.out.println("Object created so far " + count) ;
    }

    /** An entry point for program execution */
    public static void main(String[] args) {
        // instantiate an object of ObjectCounter
        ObjectCounter[] oc = new ObjectCounter[11] ;

        // fill the array with instnace of ObjectCounter */
        for (int i=0; i < oc.length; i++) {
            oc[i] = new ObjectCounter() ;
        }
    }
}
```

Download: ObjectCounter.java

6.4 ሽሬይና መጠኑ

ሽሬይ ስንፈጥር ስንት ሕዋስት እንደሚኖረው መወሰን እንደምንችል አስካሁን ድረስ ያየነው ነው። ነገር ግን ይህ ግዴታ አይደለም። የሽሬዩን መጠን እንደተፈላጊነቱ የመቀየር ችሎታው አለ። አንዳንድ ጊዜ የሽሬይን ተውላጠ-ቃል አውጥቶ ከጊዜ በኃላ ቦታውን መፍጠር አንደዚሁ። ምሳሌዎችን እንይ።

```
ObjectCounter[] oc ;
```

ይህ ሽሬይ መደባዊ ሆኖ ቦታ ግን አልተሰጠውም። ከጊዜ በኃላ ፥ ቦታ መሰየም እንደሚከተለው ይፈቀዳል። በዚህ መሠረት ሽሬዩ 128 ሕዋሳት ይኖሩታል።

```
oc = new ObjectCounter[128] ;
```

የሽሬዩ የሕዋሳት ቁጥር ብዙ ሆኖ ከተገኘ ፥ በቀላሉ መቀየር ይቻላል።

```
oc = new ObjectCounter[16] ;
```

አንድ ሽሬይ አዲስ ቦታዎች ሲሰይም ፥ ከዛ በፊት ይዞት የነበረውን ቦታ የመልቀቅ ግዴታ አለበት። መልሶ የማየት ዕድል በፍጹም የለውም። የተለቀቁ ቦታዎች ከፕሮግራሙ ውጭ ይሆናሉ። ሽሬዮዎችን አዳዲስ ቦታዎቹን መሰየም ቢቻልም ቅሉ የተሰየመን ቦታ ማስፋት ወይም ማጥበብ ክልክል ነው። በሌላ በኩል ሽሬዩ የያዘውን ቦታ ማራገፍ ከተፈለገ መንገዱ እንዲህ ነው።

```
oc = null ;
```

የnull ትርጉም «ምንም» ነው። ሽሬዩ ይዞት የነበረው ቦታ ይጠፋል። አሁን ባለበት ሁኔታ ለሥራ ዝግጁ አይደለም።

6.5 ሽሬይ አስነዳድ

የጃቫ ፕሮግራሞች ለሥራ በሚሰማሩበት አካባቢዎች ውስጥ የሚሞሪ ቦታዎች ጠባቂና አስተዳዳሪ ኦ/ኤስ (Operating System) ነው። ለፕሮግራሞች ቦታ ያድላል። የለቀቁትን ይሰበስባል። የቦታዎችን አጥረት የመከታተል ኃላፊነቱ የእሱ ነው።

የሚሞሪ ቦታዎችን ሲያድል ውስጣቸውን አያጸዳም። ተጠቃሚዎቹ ፕሮግራሞች የለቀቁትን ቦታ ሲመልሱ እንዲሁ። ይህ ሁኔታ ለግድፈቶች መነሻ ጥርጊያ መንገዶች ይከፍታል።

ፕሮግራሞች ተውላጠ-ቃል ሲያወጡ ፣ መደብ ሲያራቡ ፣ እንዲሁም ኧሬይ ሲፈጥሩ ኦ/ኤሱ የተፈለገውን ቦታ ያድላል። የተገኙ ቦታዎች ስለማይጸዱ ፣ ይዞታቸው ምን እንደሆነ ማወቅ ወይም መገመት አይቻልም። ጥንቃቄ ካልወሰድን ፣ ያልታሰበ ግድፈት ፕሮግራማችን ውስጥ እንከታለን።

ስለዚህ ፣ በአጠቃላይ ደረጃ የተለመደው ፣ ተውላጠ-ቃላትን ስናወጣ ፣ እንዲሁም ኧሬዮችን ስንፈጥር ፣ ለሥራ በሥነ-ሥርዓት በቅድሚያ ማዘጋጀት ነው። ተውላጠ-ቃላቱንም ሆነ ኧሬዮችን ባግባብ መነሻ ዋጋ መሰየም። ይኽን ተግባር በእንግሊዘኛ «initialization» ይሉታል።

ጃቫ በተፈጥሮው ተውላጠ-ቃላትን በራሱ መነሻ ዋጋ ይሰይማል። በመሆኑም ፣ ላይ የጠቀስነውን የግድፈት መነሻ ሁኔታ ይቀንሳል። ይህ በዚህ እንዳለ ፣ ኧሬይ ስናወጣ ፣ መነሻ ዋጋ መሰየም ተፈላጊ ከሆነ ፣ ኃላፊነቱ የራሳችን ነው። ጥቂት ምሳሌዎችን እንመለከት።

```
double[] num = {3.14159, 2.71, 1.6} ;
```

ይህ ኧሬይ ቃል ፣ የሕዋሳቱን መጠን በገሀድ ሳይጠቅስ ኧሬይ ይፈጥራል። ሰንት ሕዋሳት መኖር እንዳለበቸው ወሳኙ የጃቫ ኮምፓይለር ሲሆን ቁጥሩ የሚመጣው ከመነሻ ዋጋዎቹ ነው። ስለዚህ የnum ሕዋሳት ቁጥር 3 ነው። በተጨማሪ የኧሬዩ የሕዋሳት ቁጥር 3 ከመሆኑም በላይ በእያንዳንዱ ሕዋሳት ውስጥ ከላይ የቀረቡት ዕሴቶች በተርታ ይጠበቃሉ። ዝርዝሩ፦

- የnum[0] ይዘት 3.14159
- የnum[1] ይዘት 2.71
- የnum[2] ይዘት 1.6

መነሻ ዋጋዎች አሰያይም ልዩ ልዩ መልክ ወይም ቃል አገባብ አላቸው። የchar ኧሬይ እንዲህ ነው።

```
char[] lc = {'a','b','c','d','e','f','g','h','i'} ;
```

የመነሻ ዋጋዎችን ልብ ካልን እያንዳንዱ ሆሄ በተናጠል የጥቅስ ምልክት ታጥሯል። የኧሬዩ የሕዋሳት ቁጥር 9 ነው። በመደብ ላይ የተመሠረቱ ኧሬዮች መነሻ ርቢ አሰያየም ደግሞ ለጥቆ።

```
String[] names = {new String("Earth"), new String("Mars")} ;
```

የnames ኧሬይ 2 ሕዋሳት አሉት። የመጀመሪያው ሕዋስ ይዘት የእስትሪንግ ርቢ ሲሆን ዕሴቱ «Earth» ነው። የሁለተኛው ሕዋስ ይዘት «Mars» ነው።

የሚከተለው ምሳሌ ስለተርታ (sorting) ነው። የእስትሪንግ ኧሬይ ከፈጠረ በኃላ ተርታ ለማስያዝ ይሞክራል። መላዎች ሲጠሩ የኧሬይን አለካክና አቀባበል እንዲሁም ለውጥ የሚያስከትለው ሁኔታ ጥብቅ ትኩረት ቢሰጠው መልካም ነው። ምሳሌው ለጥናት መልካም ይሁን እንጂ ለተግባራዊ ሥራ ሌሎች መፍትሔዎች ይመረጣሉ።

የፍልቅልቅ ተርታ ስልት

```
public class BubbleSort {
   /** Sorts names using bubble sorting algorithm */
   void bubble(String[] names) {
      for (int i=0; i < names.length -1; i++) {
```

```
            for (int j=i+1; j < names.length; j++) {
               if (names[i].compareTo(names[j]) > 0) {
                  String temp = names[i] ;
                  names[i] = names[j] ;
                  names[j] = temp ;
               }
            }
         }
      }

      /** Prints array */
      void print(String[] names) {
         for (int i=0; i < names.length; i++)
            System.out.println(names[i]) ;
      }

      /** An entry point for program execution */
      public static void main(String[] args) {
         // delcare array of String
         String[] list = { "Mercury", "Venus", "Earth", "Mars" } ;

         BubbleSort sort = new BubbleSort() ; // instantiate obj.
         sort.bubble(list) ; // Sort the array
         sort.print(list) ;  // print sorted array
      }
   }
```

Download: BubbleSort.java

6.6 ማንኛውም ኧሬይ የመደብ ርባታ (object) ነው

በጃቫ ቋንቋ ፥ ማንኛውም ኧሬይ የመደብ ርቢ (object) ነው። ይህ ከሌላ የፕሮግራም ቋንቋ ለምንመጣ አደናጋሪ ሊሆን ይችላል። ነገር ግን አውነት ነው። የመደብ ዝምድናው ከ**Object** መደብ ጋር ነው። ወደፊት እንደምናየው ማንኛውም መደብ የObject ዘር ነው። ከሱ ጋር የማይዛመድ መደብ መኖር አይችልም።

ኧሬዮች የመደብ ርቢ በመሆናቸው ሁሉም የሚጋሯቸው የመላና የተውላጠ-ቃል አባላት አሏቸው። የሕዋሳት ቁጥራቸውን በlength አባል ተውላጠ-ቃል ይጠብቃሉ።

የአንድ ኧሬይ መጠን (length) የሚወሰነው የኧሬዩ ቦታ ሲከፈት ወይም ሲታወቅ ነው። በዛን ጊዜ የlength ይዘት የኧሬዩ መጠን ይሆናል። የlengthን ይዘት መቀየር በፍጹም አይቻልም።

```
double[] charge = new double[9] ;
```

ልክ ይህ ኧሬይ ሲፈጠር ፥ ከሚኖሩት አባላቱ መካከል አንዱ length ከመሆኑም በላይ የኧሬዩን መጠን ስንት እንደሆነ በሆድቃው ይጠብቃል። መጠኑ 9 ነው።

ምንም እንኳን በ**Object** መደብ ውስጥ ያሉትን መላዎች ማንኛውም ኧሬይ ቢወርስም ፥ በተግባር ደረጃ ይኸን ያህል አያስፈልጉም። ለማንኛውም ግን መኖራቸው መረሳት የለበትም።

6.7 ባለብዙ ኧሬይ (Multidimensional array)

እስካሁን ድረስ የተመለከትናቸው ኧሬዮዎች በሙሉ ባለተናጠል መስመር ናቸው። በተርታ የተደራጁ ፥ ቅደም ተከተል ያላቸው ፥ አንዱ ከአንዱ እንደ ሰንሰለት የተያያዙ። ተናጠል ኧሬዮዎች በተናጠል ለሚደራጁ ዴታዎች አመቺ ሆነው በሠንጠረጅ መዋቅር ለሚደራጁ ዴታዎች ግን አመቺነታቸው ይቀንሳል።

በሠንጠረጅ መልክ መደራጀት ያለባቸው ዴታዎች አንድ አይነት መሆናቸው ከታወቀ ፥ ባለብዙ ኧሬይ መጠቀሙ አንዱ መፍትሔ ነው።

ባለብዙ ኧሬይ ፥ በእንግሊዘኛ **(multidimensional array)** ፥ በአያሌ አቅጣጫዎች ተዘርጊ ቦታ ለመፍጠር ያስችላል። ባለሠንጠረጅ ኧሬይ ሁለት አቅጣጫ አለው ፤ ዓምዳዊና ረድፋዊ።

ሁለት አቅጣጫ ያለው የባለብዙ ኧሬይ ቃል አወጣጥ ይኸን ይመስላል።

```
int[][] table = new int[10][10] ;
```

ኧሬዩ ስንት አቅጣጫዎች እንዳሉት ወሳኙ የ[] ቁጥር ነው። በዚህ ምሳሌ ሁለት ቁጥተኛ ቅንፎች አሉን ፤ በመሆኑም **table** ባለሁለት አቅጣጫ ኧሬይ ይሆናል። በስተቀኝ በኩል ያለው 10 በ 10 የተባዛ ቦታ ይፈጥራል። በአጠቃላይ የሕዋሳቱ ቁጥር 100 ነው። እነዚህን ሕዋሳት ማየት የሚቻለው አንድ በአንድ ሲሆን ቃል አገባቡ ይኸን ይመስላል።

```
table[0][9] = 12 ;
```

በ0 ረድፍና በ9 ዓምድ ያለው ሕዋስ ውስጥ 12ን ይከታል። በዚህ መንገድ እያንዳንዱን ሕዋስ መጠቀም ይቻላል። ሁሉንም በአንድ ቃል ማየት ወይም መንካት አይፈቀድም።

የሕዋሳቱ መጠኖች የግድ ተመሳሳይ መሆን የለባቸውም። የተፈለገውን ያህል ልዩ ልዩ መሆን ይችላሉ። ለምሳሌ፦

```
Double[][][] values = new Double[3][7][2] ;
```

ባለሦስት አቅጣጫ ኧሬይ ይፈጠራል። ነገር ግን እያንዳንዳቸው ያላቸው የሕዋሳት መጠን የተለያየ ነው። ይህ ባለሦስት ኧሬይ በአጠቃላይ 42 ሕዋሳት አሉት። የመጨረሻው ሕዋስ ውስጥ 12.29 ማስገባት ብንሻ ቃሉ ይኸን ይመስላል።

```
values[2][6][1] = 12.29 ;
```

ይኸን ክፍል በተግባራዊ ምሳሌ እንጨርሳለን። ፕሮግራሙ በተሰጠው የላዕላይ ቁጥር ላይ በመንተራስ የማባዣ ሠንጠረጅ ያበጃል።

የማባዣ ሠንጠረጅ

```
public class MultTable {
   /** Generates a multiplication table */
   void create(int max) {
      //    if max bad, terminate
      if (max <= 0) return ;

      // declare two dimensional array
      int[][] table = new int[max][max] ;

      // generate the multiplication table
```

```
      for (int i=1; i <= max; i++) {
         for (int j=1; j <= max; j++) {
            table[i-1][j-1] = i * j ;
         }
      }
      // print the table
      for (int i=0; i < max; i++) {
         for (int j=0; j < max; j++) {
            System.out.print("" + table[i][j] + '\t') ;
         }
         System.out.println() ;
      }
   }

   /** An entry point for program execution */
   public static void main(String[] args) {
      int max = 10;
      MultTable m = new MultTable() ; // instantiate obj.
      m.create(max) ;
   }
}
```

Download: MultTable.java

To contact: info@senamirmir.com
Copyright © 2002-2005 Senamirmir Project