

ઁરે અને સ્ટ્રિંગનો ઉપયોગ

9

પ્રકરણ 7માં આપણે જાવામાં ઉપલબ્ધ ડેટાના મૂળભૂત પ્રકારોનો અભ્યાસ કર્યો. મૂળભૂત પ્રકારના ચલ (variable)માં એક સમયે ફક્ત એક જ કિંમત રાખી શકાય છે. આ પ્રકારના ચલને અદિશ ચલ કે સ્કેલર વેરિયેબલ (scalar variable) કહેવામાં આવે છે. આ પ્રકરણમાં આપણે કેટલાક સંયોજિત કે કોમ્પોઝિટ (composite) ડેટાપ્રકારો વિશે અભ્યાસ કરીશું જેનો ઉપયોગ એક કરતાં વધારે ડેટાકિંમતોના સમૂહનો સંગ્રહ કરવા માટે થાય છે, ઉદાહરણ તરીકે, ઁરે (array) અને સ્ટ્રિંગ (string).

ઁરેનો પરિચય (Introduction to Array)

ઁરે (array) એ એક જ પ્રકારના ઘટકોના સંગ્રહને રજૂ કરતો ચલ (variable) છે. ઁરે સદિશ (vector), શ્રેણિક (matrix) અને અન્ય બહુ-પરિમાણીય (multi-dimensional) માહિતી રજૂ કરવા માટે ઉપયોગી છે. વેક્ટર (vector) એ એક-પરિમાણીય (1-D) ડેટા-સંરચના (data structure) છે, જે અક્ષરો, સંખ્યાઓ જેવી વસ્તુઓની યાદી સંગ્રહવા માટે વાપરી શકાય છે. રો (આડી હરોળ) અને કોલમ (ઊભી હરોળ) વડે બનેલી કોષ્ટક જેવી દ્વિ-પરિમાણીય (2-D) ડેટા-સંરચના રજૂ કરવા માટે મેટ્રિક્સ (matrix) વપરાય છે.

જ્યારે સમાન પ્રકારના અનેક ઘટકો ઉપર એકસમાન કાર્ય કરવાનું હોય, ત્યારે ઁરે ઉપયોગી બને છે. ઁરેના તમામ ઘટકો મેમરીમાં એકસાથે લગોલગ સંગ્રહાયેલા હોય છે. ઁરેના દરેક ઘટકને ઁરે વેરિયેબલ સાથે સંકળાયેલા સૂચક સ્થાનાંક (index position) વડે ઓળખવામાં આવે છે.

જાવામાં વસ્તુઓની યાદીનું સંચાલન કરવા માટે વપરાતો ઓબ્જેક્ટ એ ઁરે છે. ઁરે બનાવવા માટે બે પગલાંની ક્રિયા છે :

1. ઁરે-ઓબ્જેક્ટ ઘોષિત કરો.
2. ઁરે-ઓબ્જેક્ટની રચના કરો.

ઁરે-ઓબ્જેક્ટ બે રીતથી બનાવી શકાય છે :

1. new પ્રક્રિયક વાપરીને અને તેનું કદ જણાવીને.
2. સીધેસીધા ઁરેના ઘટકોને પ્રારંભિક કિંમત આપીને. (initializing)

એક-પરિમાણીય ઁરે (1-D Array)

એક-પરિમાણવાળા ઁરે 1-D ઁરે તરીકે ઓળખાય છે. ઉદાહરણ તરીકે, વેક્ટર (vector) કે જેને એક અથવા વધારે અદિશ ચલ (scalar variables)ના સમૂહ તરીકે ગણી શકાય. marks1, marks2, marks3, marks4, marks5ની રીતે અલગ-અલગ ચલ ઘોષિત કરવાના બદલે આપણે ઁરે marks[5] ઘોષિત કરી શકીએ અને તેના દરેક ઘટકને marks[0], marks[1], marks[2], marks[3], marks[4] વાપરી આપણે તેને મેળવી શકીએ છીએ.

1-D ઁરે ઘોષિત કરવા માટે ચોરસ કૉસની જોડી [] ઁરેના નામ પછી અથવા ડેટાના પ્રકાર પછી આપણે વાપરીએ છીએ. ઁરે ઘોષિત કરવાની વાક્યરચના નીચે પ્રમાણે છે :

<data type> <array name> [] ; અથવા <data type> [] <array name>;

ઉદાહરણ તરીકે, એક વિદ્યાર્થીના ગણિત વિષયની પાંચ કસોટીમાં મેળવેલા ગુણનો સંગ્રહ કરવા માટે આપણે પાંચ પૂર્ણાંક સંખ્યા ધરાવતા ઘટકોનો ઁરે વાપરી શકીએ. 'marks' નામનો પાંચ ઘટકો (elements) ધરાવતો ઁરે-ઓબ્જેક્ટ નીચે મુજબ બનાવી શકાય છે :

```
int marks[];           // declare array object
marks = new int [5];    // create array object
```

ઉપરનાં બંને વિધાનોને ભેગાં કરીને નીચે મુજબ એક જ વિધાનથી પણ આપણે એરે બનાવી શકીએ :

`int marks[] = new int [5];` અથવા `int [] marks = new int [5];`

અહીં એરેનું નામ 'marks' છે. પાંચ પૂર્ણાંક સંખ્યાઓને સંગ્રહ જ્યાં થશે, તે મેમરી સ્થાનાંકનો તે નિર્દેશ કરે છે. `int` ડેટા પ્રકારની પૂર્ણાંક સંખ્યાના સંગ્રહ માટે 4 બાઈટ વપરાય છે. આથી, 'marks' એરે માટે મેમરીમાં સળંગ $5 \times 4 = 20$ બાઈટની જરૂર પડે છે.

એરેના કોઈ ઘટકનો નિર્દેશ કરવા માટે આપણે આકૃતિ 9.1માં દર્શાવ્યા પ્રમાણે એરેના નામ પછી મોટા કૉસ []ની અંદર ઇન્ડેક્સ (index) અથવા સબસ્ક્રિપ્ટ (subscript)નો ઉપયોગ કરીએ છીએ. ઇન્ડેક્સ કે સબસ્ક્રિપ્ટ એરેમાં ઘટકનું સ્થાન દર્શાવે છે. ઉદાહરણ તરીકે, `marks[2]`. ઇન્ડેક્સની કિંમત શૂન્ય (0)થી શરૂ થાય છે.

`int marks[] = {90, 60, 70, 65, 80};`

	Element Reference	Array Content	Array Index
marks Reference Variable	marks[0]	90	0
	marks[1]	60	1
	marks[2]	70	2
	marks[3]	65	3
	marks[4]	80	4

આકૃતિ 9.1 : એક-પરિમાણીય એરે

આકૃતિ 9.1માં એરે ચલ `marks`ની ઇન્ડેક્સકિંમત 0થી 4 છે. અહીં, `marks[0]` એરેના પ્રથમ ઘટકનો નિર્દેશ કરે છે અને `marks[4]` એ છેલ્લા ઘટકનો નિર્દેશ કરે છે.

પ્રકરણ 8માં સમજૂતી આપ્યા પ્રમાણે ઓબ્જેક્ટ એક રેફરન્સ ચલ છે. એરે જાવામાં એક ઓબ્જેક્ટ હોવાથી એરેનું નામ પણ એક રેફરન્સ ચલ છે. તે મેમરીના એ સ્થાનાંકનો રેફરન્સ ધરાવે છે, જ્યાં એરેના ઘટકોનો સંગ્રહ થયેલો છે. આકૃતિ 9.1 જુઓ.

એરે એક ઓબ્જેક્ટ છે, આથી તેના ઘટકોને પૂર્વનિર્ધારિત (default) કિંમતો આપવામાં આવે છે (initialized). જાવા-પ્રોગ્રામમાં એરેનો ઉપયોગ કેવી રીતે કરવો તે આકૃતિ 9.2માં દર્શાવ્યું છે.

```

Array1.java - SciTE
File Edit Search View Tools Options Language Buffers Help

1 Array1.java
class Array1
{
    public static void main (String [] s)
    {
        int marks[];
        marks = new int [7];

        for (int i=0; i<7; i++)
        {
            System.out.println ("marks [ "
                                + i + " ] is " + marks[i]);
        }
    }
}

>javac Array1.java
>Exit code: 0
>java -cp . Array1
marks [ 0 ] is 0
marks [ 1 ] is 0
marks [ 2 ] is 0
marks [ 3 ] is 0
marks [ 4 ] is 0
marks [ 5 ] is 0
marks [ 6 ] is 0
>Exit code: 0

```

આકૃતિ 9.2 : એરે-ઓબ્જેક્ટના ઘટકોને પૂર્વનિર્ધારિત કિંમતો પ્રારંભિક કિંમત તરીકે આપવી

જ્યારે એરે ઘોષિત કરવામાં આવે છે, તે સમયે પણ તેના ડેટા ઘટકોની પ્રારંભિક કિંમત (initial values) જણાવી શકાય છે. 1-D એરેના ઘટકોની પ્રારંભિક કિંમતો છગડિયા કૉસ { }માં અલ્પવિરામ વડે અલગ પાડેલી કિંમતો વડે આપી શકાય છે. ઉદાહરણ તરીકે,

`int marks[ ] = {90, 70, 77};` અથવા `int [] marks = {90, 70, 77};`

અહીં નોંધ કરો કે આપણે એરે બનાવતા સમયે જો તેના ઘટકોની પ્રારંભિક કિંમતો આપીએ, તો `new` પ્રક્રિયકના ઉપયોગની જરૂર રહેતી નથી. એરેનું કદ કૌંસમાં આપેલી કિંમતોની સંખ્યા બરાબર છે.

આકૃતિ 9.3માં આપેલો જાવાકોડ એરે બનાવવા માટે તેમજ તેના ઘટકોની પ્રારંભિક કિંમતો આપવા માટેની અલગ-અલગ રીતો દર્શાવે છે. એરેના ઘટકોની કિંમતો પ્રદર્શિત કરવા માટે વાપરેલી `display()` ક્લાસ મેથડની નોંધ લો. ક્લાસમેથડને `static` ઘોષિત કરાય છે અને તે ક્લાસના કોઈ પણ ઓબ્જેક્ટ વિના વાપરી શકાય છે.

```
File Edit Search View Tools Options Language Buffers Help
1 Array2.java
/* creating and initializing 1-D array in different ways */
class Array2
{
    public static void main (String [] s)
    {
        int marks1[];
        marks1 = new int [3];

        int marks2[] = new int[3];
        int [] marks3 = new int [3];
        int marks4[] = {50, 60, 70};
        int [] marks5 = {70, 80, 90};

        System.out.print ("Array marks1:\t");
        display(marks1,3);
        System.out.print ("Array marks2:\t");
        display(marks2,3);
        System.out.print ("Array marks3:\t");
        display(marks3,3);
        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array2.java
>Exit code: 0
>java -cp . Array2
Array marks1:  0  0  0
Array marks2:  0  0  0
Array marks3:  0  0  0
Array marks4:  50  60  70
Array marks5:  70  80  90
>Exit code: 0
```

આકૃતિ 9.3 : એરે-ઓબ્જેક્ટ બનાવવા તથા તેના ઘટકોને પ્રારંભિક કિંમતો આપવાની વિવિધ રીત

જાવામાં આપણે જ્યારે એરે ઘોષિત કરીએ, તે સમયે પરિમાણનું માપ (dimensions) અને તેના ઘટકોની પ્રારંભિક કિંમતો (initial values) બંને એકસાથે આપી શકતા નથી. આકૃતિ 9.4માં આ વિધાનને સમજાવતો કોડ દર્શાવ્યો છે. જો આપણે એરે ચલ પ્રારંભિક કિંમતો વગર (without initialization) ઘોષિત કરીએ, તો આકૃતિ 9.1માં દર્શાવ્યા પ્રમાણે સંદર્ભ રાખવા માટેનો માત્ર ચલ બને છે. એ સમયે એરે-ઓબ્જેક્ટ બનતો નથી, એટલે કે એરોના ઘટકો માટે મેમરી ફાળવવામાં આવતી નથી. જો આપણે એરેના ઘટકો વાપરવા માટે પ્રયત્ન કરીએ તો તે ભૂલમાં પરિણમે છે.

```
File Edit Search View Tools Options Language Buffers Help
1 Array3.java
/* creating and initializing 1-D array
specifying size with array variable*/
class Array3
{
    public static void main (String [] s)
    {
        int marks4[3] = {50, 60, 70};
        int [5] marks5 = {70, 80, 90};

        System.out.print ("Array marks4:\t");
        display(marks4,3);
        System.out.print ("Array marks5:\t");
        display(marks5,3);
    }

    static void display(int arr[], int size)
    {
        for (int i=0; i<size; i++)
        {
            System.out.print (arr[i] + "\t");
        }
        System.out.println();
    }
}
```

```
>javac Array3.java
Array3.java:8: ']' expected
int marks4[3] = {50, 60, 70};
^
Array3.java:8: illegal start of expression
int marks4[3] = {50, 60, 70};
^
Array3.java:8: illegal start of expression
int marks4[3] = {50, 60, 70};
^
Array3.java:8: not a statement
int marks4[3] = {50, 60, 70};
^
Array3.java:8: ';' expected
int marks4[3] = {50, 60, 70};
^
Array3.java:9: ']' expected
int [5] marks5 = {70, 80, 90}.
^
Array3.java:9: ';' expected
int [5] marks5 = {70, 80, 90};
^
Array3.java:9: <identifier> expected
int [5] marks5 = {70, 80, 90}.
^
Array3.java:11: <identifier> expected
System.out.print ("Array marks4\t");
^
Array3.java:11: illegal start of type
System.out.print ("Array marks4\t");
^
```

આકૃતિ 9.4 : એરે-ઓબ્જેક્ટના ઘટકોને પ્રારંભિક કિંમતો આપતા સમયે તેનું કદ જણાવવું અમાન્ય છે.

એરેનો દરેક ઘટક એક અલગ ચલ છે, જેનો ઇન્ડેક્સ (index) વાપરીને નિર્દેશ કરી શકાય છે. એરેના ઘટકની કિંમત બદલવા માટે અન્ય ચલની જેમ આપણે ઘટક ચલને એસાઈનમેન્ટ વિધાનમાં ડાબી બાજુએ વાપરી શકીએ છીએ. ઉદાહરણ તરીકે, `marks[3] = 56;` ચાલો, આપણે 10 અપૂર્ણાંક સંખ્યાઓની સરેરાશ (મધ્યક) શોધવા માટેનો પ્રોગ્રામ લખીએ અને તેનો અમલ કરીએ. આકૃતિ 9.5માં આપેલું કોડલિસ્ટિંગ અને તેના આઉટપુટનો અભ્યાસ કરો.

```

ArrayAvg.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 ArrayAvg.java
/* compute average of 10 numbers */
class ArrayAvg
{
    public static void main (String [] s)
    {
        double numbers [] = { 10.5, 20.6, 30.8, 15.5,
                               17.3, 25.5, 27.2,
                               20, 30, 18.5};

        byte ctr;
        double sum=0, avg;

        System.out.println ("list of numbers is");
        for (ctr=0; ctr<10; ctr++)
        {
            System.out.println (numbers[ctr]);
            sum = sum + numbers[ctr];
        }
        avg = sum/10;
        System.out.println ("\nAverage of above numbers is "
                             + avg);
    } // main
} // class

```

```

>javac ArrayAvg.java
>Exit code: 0
>java -cp . ArrayAvg
list of numbers is
10.5
20.6
30.8
15.5
17.3
25.5
27.2
20.0
30.0
18.5
Average of above numbers is 21.59
>Exit code: 0

```

આકૃતિ 9.5 : 1-D એરે અને લૂપનો ઉપયોગ કરીને 10 સંખ્યાની સરેરાશ શોધવાનો પ્રોગ્રામ

અહીં, આપણને સમાન પ્રકારના અલગ-અલગ ઘટકો ઉપર એક જ જાતની ક્રિયા ‘સંખ્યાને સરવાળામાં ઉમેરો’, વારંવાર કરવાની જરૂર પડે છે. એકસાથે 10 ચલને ઘોષિત કરવા અને લૂપનો ઉપયોગ કરી એરેના જુદા-જુદા ઘટકો ઉપર એકસમાન પ્રક્રિયા કરવામાં એરેનો ઉપયોગ મદદરૂપ થાય છે.

એરે ઉપર આપણે અન્ય અનેક વિવિધ પ્રક્રિયાઓ કરી શકીએ છીએ. ઉદાહરણ તરીકે, બે એરેની સરખામણી કરવી, એક એરેના બધા ઘટકોની અન્ય એરેમાં નકલ કરવી, કોઈ ચોક્કસ ઘટક એરેમાં શોધવો, એરેના ઘટકોને ક્રમાનુસાર ગોઠવવા વગેરે. આ પ્રકારનાં બધાં કાર્યો માટે જેમ આપણે ઘટકોનું સરેરાશ શોધવા માટે લખી હતી તે પ્રમાણે પ્રોસીજર (procedures) લખી શકીએ. આ માટેનો કોડ આપણે જાતે લખવાને બદલે જાવા દ્વારા પૂરી પાડવામાં આવેલી `java.util.Arrays` ક્લાસની વિવિધ static methodsનો આપણે ઉપયોગ કરી શકીએ છીએ. જાવામાં એરેને `Arrays` classની મેથડ (methods)નો ઉપયોગ કરવાની સગવડ આપે છે. આકૃતિ 9.6માં દર્શાવેલું કોડલિસ્ટિંગ અને તેનો આઉટપુટ `java.util.Arrays` ક્લાસની `sort()` અને `fill()` મેથડનો ઉપયોગ કઈ રીતે કરી શકાય તે દર્શાવે છે.

આપણે સમગ્ર એરે કે તેના અમુક ભાગને ક્રમાનુસાર ગોઠવવા માટે (શોર્ટ કરવા) `sort()` મેથડ વાપરી શકીએ છીએ. આપણે જ્યારે મેથડના આર્ગ્યુમેન્ટ (argument) તરીકે ફક્ત એરે આપીએ, ત્યારે તે સમગ્ર એરેને (એરેના બધા ઘટકોને) સોર્ટ કરે છે. જો આપણે એરે, આરંભસ્થાન, અંતિમસ્થાન એમ પણ આર્ગ્યુમેન્ટ આપીને મેથડ વાપરીએ, તો તે આરંભસ્થાનના ઘટકથી અંતિમસ્થાનથી આગળના ઘટક સુધીના આંશિક એરેને શોર્ટ (sort) કરશે. ઉદાહરણ તરીકે, `java.util.Arrays.sort(list, 1, 5)`નો ઉપયોગ કરવાથી તે `list[1]` થી `list[5-1]` સુધીના એરેના ઘટકોને શોર્ટ કરશે. જુઓ આકૃતિ 9.6.

સમગ્ર કે આંશિક એરેને કોઈ ચોક્કસ કિંમતથી ભરવા માટે ‘fill’ મેથડનો ઉપયોગ થાય છે. જ્યારે મેથડ એરે અને કિંમત એમ બે આર્ગ્યુમેન્ટ સાથે ઇન્વોક કરવામાં આવે, ત્યારે એરેના બધા ઘટકોમાં તે જણાવેલી કિંમત ભરવામાં આવે છે. જ્યારે આ મેથડ એરે, આરંભસ્થાન, અંતિમ સ્થાન અને કિંમત એમ ચાર આર્ગ્યુમેન્ટ સાથે ઇન્વોક કરવામાં આવે છે, ત્યારે તે આરંભસ્થાનથી શરૂ કરી અંતિમ સ્થાનથી આગળના ઘટક સુધી આપેલી કિંમત વડે ભરી દે છે. ઉદાહરણ તરીકે, `fill(list, 7)` એરેના બધા ઘટકો કિંમત 7 વડે ભરવામાં આવે છે, જ્યારે `fill(list, 2, 6, 5)` એરેના ઘટકો `list[2]` થી `list[6-1]`માં કિંમત 5 ભરવામાં આવે છે. જુઓ આકૃતિ 9.6.

```

ArraysClassSortFill.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 ArraysClassSortFill.java
class ArraysClassSortFill
{ // sort and fill methods on whole or partial array
public static void main (String [] s)
{
    double list[] = { 6.4, 8, 7.8, 9.8, 9.5,
                     6, 7, 8, 8.5, 5.9 };

    int indx;

    System.out.println ("Initial Elements:");
    display(list);
    java.util.Arrays.sort (list, 3, 9); //sort partial array
    System.out.println ("nsort partial array: list[3] to list[8]:");
    display(list);
    java.util.Arrays.sort (list); //sort whole array
    System.out.println ("nsort whole array:");
    display(list);

    java.util.Arrays.fill (list,7); //fill whole array
    System.out.println ("nFill whole array:");
    display(list);
    java.util.Arrays.fill (list, 2, 6, 5); //fill partial array
    System.out.println ("nFill partial array: list[2] to list[5]");
    display(list);
} // end main

static void display(double ary[])
{
    for (int i=0; i<ary.length; i++)
    {
        System.out.print (ary[i] + " ");
    }
    System.out.println();
} // end display
} // end class

>javac ArraysClassSortFill.java
>Exit code: 0
>java -cp . ArraysClassSortFill
Initial Elements:
6.4 8.0 7.8 9.8 9.5 6.0 7.0 8.0 8.5 5.9
sort partial array: list[3] to list[8]:
6.4 8.0 7.8 6.0 7.0 8.0 8.5 9.5 9.8 5.9
sort whole array:
5.9 6.0 6.4 7.0 7.8 8.0 8.0 8.5 9.5 9.8
Fill whole array:
7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0 7.0
Fill partial array: list[2] to list[5]
7.0 7.0 5.0 5.0 5.0 5.0 7.0 7.0 7.0 7.0
>Exit code: 0

```

આકૃતિ 9.6 : એરેક્લાસની sort() અને fill() મેથડનો ઉપયોગ

આપેલી કિંમતવાળો ઘટક એરેમાં શોધવા માટે એરે ક્લાસમાં binarySearch() મેથડ ઉપલબ્ધ છે. સુરેખશોધ (linear search) પદ્ધતિથી આપેલી કિંમતવાળો ઘટક એરેમાં શોધવા માટે આપણે આપણી પોતાની મેથડ લખીશું. સુરેખશોધ પદ્ધતિ એરેના એક પછી એક ઘટકોને આપેલી કિંમત સાથે ક્રમાનુસાર સરખાવે છે. આકૃતિ 9.7માં આપેલા કોડલિસ્ટિંગ અને તેના આઉટપુટનો અભ્યાસ કરો. આ પદ્ધતિ મુજબ જો એરેમાં આપેલી કિંમતવાળો ઘટક મળે, તો તેનું સૂચક સ્થાન (index position) પરત કરે છે, નહિ તો -1 કિંમત પરત કરે છે.

```

LinearSrch.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 LinearSrch.java
// Search element in array using linear search
class LinearSrch
{
public static void main (String [] s)
{
    double list[] = {6, 5, 7, 9, 9.5, 6.5, 7.5, 8 };
    int indx;

    System.out.println ("Given Array elements are:");
    display(list);

    indx=search(list, 8); // search 8
    if (indx < 0)
        System.out.println("nelement 8 is not found in array");
    else
        System.out.println("nelement 8 is found at position " + indx);
    indx=search(list,5.5); // search 5.5
    if (indx < 0)
        System.out.println("nelement 5.5 is not found in array");
    else
        System.out.println("nelement 5.5 is found at position " + indx);
} // end main
static void display(double ary[])
{
    for (int i=0; i<ary.length; i++)
    {
        System.out.print ( ary[i] );
    }
    System.out.println();
} // end display
static int search(double ary[], double x)
{
    for (int i=0; i<ary.length; i++)
    {
        if (ary[i] == x ) return i;
    }
    return -1;
} // end search
} // end class

>javac LinearSrch.java
>Exit code: 0
>java -cp . LinearSrch
Given Array elements are:
6.0
5.0
7.0
9.0
9.5
6.5
7.5
8.0
element 8 is found at position 7
element 5.5 is not found in array
>Exit code: 0

```

આકૃતિ 9.7 : એરેમાં ચોક્કસ કિંમતવાળો ઘટક શોધવો

2-D એરે (2-D Array)

દ્વિ-પરિમાણીય (2-D) એરે હાર અને સ્તંભ સ્વરૂપે કોષ્ટકીય માહિતીનો સંગ્રહ કરવા માટે વપરાય છે. ઉદાહરણ તરીકે, પાંચ વિદ્યાર્થીઓના ત્રણ કસોટીના ગુણ દર્શાવવા માટે આપણે આકૃતિ 9.8માં દર્શાવ્યા મુજબ પાંચ હાર અને ત્રણ સ્તંભવાળી કોષ્ટકીય ગોઠવણીનો ઉપયોગ કરી શકીએ. ગણિતમાં આપણે તેને પાંચ હાર અને ત્રણ સ્તંભવાળું મેટ્રિક્સ (matrix) કહીએ છીએ, જેમાં દરેક ઘટકમાં ગુણ હોય છે.

Students	Test1	Test2	Test3
1	50	60	70
2	35	30	50
3	70	75	80
4	80	85	90
5	40	50	55

આકૃતિ 9.8 : 2-D એરેની કોષ્ટકીય રજૂઆત

જાવામાં 2-D એરે ઘોષિત કરવા માટે એરેનું નામ અને મોટા કૌંસની બે જોડી [] [] કે જે અનુક્રમે પરિમાણો હાર (row) અને સ્તંભ (column)ના કદનો ઉલ્લેખ કરે છે, તેનો ઉપયોગ થાય છે. ઉદાહરણ તરીકે, સળંગ મેમરીમાં $5 \times 3 = 15$ પૂર્ણાંક કિંમતોનો સંગ્રહ કરવા માટે marks નામનો એરે ઘોષિત કરવા માટે તેમજ બનાવવા માટે નીચે આપેલું વિધાન વપરાય છે :

```
int marks [][] = new int [5][3];
```

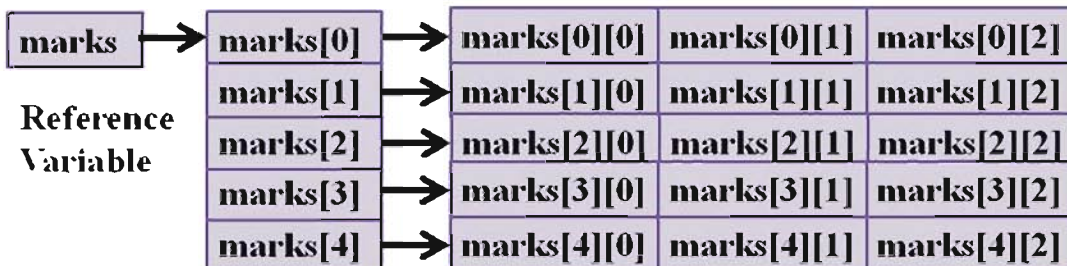
અહીં, તાર્કિક દૃષ્ટિએ એરે એ 5 હાર અને 3 સ્તંભ સાથેનું એક કોષ્ટક છે. ભૌતિક દૃષ્ટિએ તે સળંગ મેમરીમાં સંગ્રહાયેલી 15 પૂર્ણાંક કિંમતો છે, જે 60 બાઈટ જગ્યા મેમરીમાં રોકે છે.

જાવા સીધી રીતે બહુ-પરિમાણીય એરે (multi-dimensional arrays)ની સગવડ પૂરી પાડતી નથી. 2-D એરે બનાવવા માટે આપણે એરેનો એરે (array of array) બનાવવો પડે. પરિમાણની સંખ્યા ઉપર કોઈ મર્યાદા નથી.

marks [5][3]માં એક 5 ઘટકોનો 1-D એરે-ઓબ્જેક્ટ બનાવે છે અને આકૃતિ 9.9માં દર્શાવ્યા પ્રમાણે આ દરેક ઘટક એ 3 પૂર્ણાંક સંખ્યાનો એક 1-D એરે-ઓબ્જેક્ટ છે.

```
int marks = new int [5][3];
```

Element Reference using indexes



Reference
Variables

આકૃતિ 9.9 : 2-D એરે એ એક 1-D એરેના એરે તરીકે

```
File Edit Search View Tools Options Language Buffers Help
1 Array2D.java *
/* creating and initializing 2-D array */
class Array2D
{
    public static void main (String [] s)
    {
        int marks1[] { };
        marks1 = new int [ 5 ] [ 3 ];
        int marks2[] [] = new int [ 5 ] [ 3 ];
        int [] [] marks3 = new int [ 5 ] [ 3 ];
        int marks4[] [] = { { 50, 60, 70 }, { 35, 30, 50 },
                             { 70, 75, 80 }, { 80, 85, 90 },
                             { 50, 50, 55 } };
        int [] [] marks5 = { { 50, 60, 70 }, { 35, 30, 50 },
                              { 70, 75, 80 }, { 80, 85, 90 },
                              { 50, 50, 55 } };

        System.out.print ("2-D Array marks1\n");
        display(marks1,5,3);
        System.out.print ("2-D Array marks2\n");
        display(marks2,5,3);
        System.out.print ("2-D Array marks3\n");
        display(marks3,5,3);
        System.out.print ("2-D Array marks4\n");
        display(marks4,5,3);
        System.out.print ("2-D Array marks5\n");
        display(marks5,5,3);
    } // main
    static void display(int arr[][], int rows, int cols)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<cols; j++)
            {
                System.out.print (arr[i] [j] + " ");
            }
            System.out.println();
        }
    } // display
} // class
```

```
File Edit Search View Tools Options Language Buffers Help
1 Array2D.java
>javac Array2D.java
>Exit code: 0
>java -cp . Array2D
2-D Array marks1
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
2-D Array marks2
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
2-D Array marks3
0 0 0
0 0 0
0 0 0
0 0 0
0 0 0
2-D Array marks4:
50 60 70
35 30 50
70 75 80
80 85 90
50 50 55
2-D Array marks5.
50 60 70
35 30 50
70 75 80
80 85 90
50 50 55
>Exit code: 0
```

જાવામાં 2-D એરેને એરેનો એરે ગણવામાં આવે છે. આથી, 2-D એરેની હારમાં ગમે તેટલી સંખ્યામાં (પરિવર્તનશીલ - variable) સ્તંભ હોઈ શકે. આકૃતિ 9.12માં કમ્પ્યુટર-પ્રોગ્રામિંગની પાંચ ભાષાઓનાં નામનો સંગ્રહ કરવા માટે અક્ષરોના 2-D એરેનો ઉપયોગ દર્શાવ્યો છે.

Elements from `names[i][0]` to `names[i][names[i].length-1]`

Reference Variable	String
<code>names</code>	<code>names[0]</code> → J a v a
	<code>names[1]</code> → C
	<code>names[2]</code> → C + +
	<code>names[3]</code> → B a s i c
	<code>names[4]</code> → P e r l

Reference Variables

કમ્પ્યુટર-અધ્યયન : 12

દરેક નામ અલગ-અલગ હારમાં સંગ્રહવામાં આવ્યા છે અને દરેક નામમાં અક્ષરોની સંખ્યા અલગ-અલગ છે. આ રીતે દરેક હારનું કદ અલગ-અલગ છે. દરેક હારનું કદ 1-D એરેની 'length' નામની પ્રોપર્ટી વડે જાણી શકાય છે.

આકૃતિ 9.13માં અલગ-અલગ માપના સંલવાળા 2-D એરેનો ઉપયોગ કેવી રીતે કરી શકાય તે દર્શાવ્યું છે. અહીં આપણે 1-D એરેના ઘટકોની સંખ્યા જાણવા માટે length પ્રોપર્ટીનો ઉપયોગ કરેલો છે. 2-D એરે માટે તે તેની હારની સંખ્યા પરત કરે છે. 1-D એરે માટે તે આપેલી હારમાં સંલની સંખ્યા પરત કરે છે. અહીં યાદ રાખો કે 2-D એરેમાં ઘટકોની સંખ્યા એ તેમાં રહેલી હારની સંખ્યા છે અને દરેક હાર એ 1-D એરે છે. ટૂંકમાં, length પ્રોપર્ટી જ્યારે ફક્ત એરેના નામ સાથે વાપરવામાં આવે, ત્યારે તે તેના પ્રથમ પરિમાણનું કદ પરત કરે છે.

```

Array2Dchar.java - SdTE
File Edit Search View Tools Options Language Buffers Help
1 Array2Dchar.java
/* creating and initializing 2-D array */
class Array2Dchar
{
    public static void main (String [] s)
    {
        char names[] [] = {
            {'J','a','v','a'},
            {'C'},
            {'C','+', '+'},
            {'B','a','s','i','c'},
            {'P','a','s','c','a','l'}
        };

        System.out.println("Number of elements in 2-D array: "
            + names.length + "\n");
        System.out.print
            ("Five names stored in 2-D Array of characters:\n");
        display(names,5);
    } // main
    static void display(char arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            System.out.print ("Row " + i + " have " + arr[i].length +
                " character elements: ");
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i][j] + " ");
            }
            System.out.println();
        }
    } // display
} // class

```

>javac Array2Dchar.java
 >Exit code: 0
 >java -cp . Array2Dchar
 Number of elements in 2-D array: 5
 Five names stored in 2-D Array of characters:
 Row 0 have 4 character elements: Java
 Row 1 have 1 character elements: C
 Row 2 have 3 character elements: C++
 Row 3 have 5 character elements: Basic
 Row 4 have 6 character elements: Pascal
 >Exit code: 0

આકૃતિ 9.13 : અલગ-અલગ સંખ્યામાં સંલ ધરાવતા 2-D એરેનો પ્રોગ્રામમાં ઉપયોગ

હવે પછી આપણે જોઈશું કે અક્ષરોના 1-D એરેને જાવામાં ઉપલબ્ધ string ક્લાસના ઉપયોગથી વ્યાખ્યાયિત કરી શકાય છે. આકૃતિ 9.14માં આપણે નામનો સંગ્રહ કરવા માટે બાઈટનો 2-D એરે વાપર્યો છે. અહીં તે દર્શાવે છે કે char પ્રકારના ડેટાને બાઈટ પ્રકારમાં પરિવર્તિત કર્યા છે અને નામના અક્ષરોને તેની અનુરૂપ ASCII કિંમત આપવામાં આવી છે.

```

File Edit Search View Tools Options Language Buffers Help
1 Array2Dbyte.java
/* creating and initializing 2-D array */
class Array2Dbyte
{
    public static void main (String [] s)
    {
        byte names[] [] = {
            {'J','a','v','a'},
            {67},
            {'C','+', '+'},
            {'B','a','s','i','c'},
            {'P','a','s','c','a','l'}
        };

        System.out.print ("Five names stored in 2-D Array of bytes:\n");
        display(names,5);
    } // main
    static void display( byte arr[][], int rows)
    {
        for (int i=0; i<rows; i++)
        {
            for (int j=0; j<arr[i].length; j++)
            {
                System.out.print ( arr[i][j] + " ");
            }
            System.out.println();
        }
    } // display
} // class

```

>java -cp . Array2Dbyte
 Five names stored in 2-D Array of bytes:
 74 97 118 97
 67
 67 43 43
 66 97 115 105 99
 80 97 115 99 97 108
 >Exit code: 0

આકૃતિ 9.14 : 2-D એરે : અક્ષરોને અનુરૂપ પૂર્ણાંક સંખ્યાનો ઉપયોગ કરી બાઈટમાં સંગ્રહ

યાદ રાખવાના મુદ્દાઓ :

- એરે એક્સમાન પ્રકારના ડેટાનો સંગ્રહ છે.
- એરેના ઘટકોને તેના દરેક પરિમાણના ઇન્ડેક્સને કૌંસમાં [] આપીને વાપરી શકાય છે.
- ઇન્ડેક્સની કિંમત શૂન્યથી શરૂ થાય છે.
- બહુ-પરિમાણીય એરેમાં બીજા, ત્રીજા,...પરિમાણોનું કદ અલગ-અલગ હોઈ શકે છે.
- પરિમાણનું કદ જાણવા માટે 'length' એટ્રિબ્યૂટનો ઉપયોગ થાય છે.
- એરેને ઓબ્જેક્ટ તરીકે ગણવામાં આવે છે.
- એરેને તેની પ્રારંભિક કિંમતો આપ્યા વિના ઘોષિત કરવાથી એરે ઓબ્જેક્ટ બનતો નથી.

સ્ટ્રિંગ (Strings)

સ્ટ્રિંગ એ અક્ષરોની એક શ્રેણી સિવાય બીજું કંઈ જ નથી. આથી અક્ષરોના 1-D એરેને સ્ટ્રિંગ તરીકે ગણી શકાય. આપણે અગાઉ string લિટરલ વાપર્યા છે અને તેનો અભ્યાસ પણ કર્યો છે, કે જેમાં અક્ષરોની શ્રેણીને બેવડા અવતરણ-ચિહ્ન (double quotes) વચ્ચે લખવામાં આવે છે.

સ્ટ્રિંગનો સંગ્રહ કરી શકે તેવા ચલ વાપરવા માટે જાવામાં બે પ્રકારની સ્ટ્રિંગ આપેલી છે, જે 'String' અને 'StringBuffer' નામના બે ક્લાસ વડે નિયંત્રિત થાય છે. આ પ્રકરણમાં આપણે પ્રથમ પ્રકારની સ્ટ્રિંગનો અભ્યાસ કરીશું.

સ્ટ્રિંગ ઓબ્જેક્ટ બનાવવા માટે વપરાતાં કેટલાંક કન્સ્ટ્રક્ટર (constructor) નીચે મુજબ છે :

- String () – ચલ રહિત String () એક પણ અક્ષર વગરનો સ્ટ્રિંગ ઓબ્જેક્ટ બનાવે છે.
- String (char ary[]) – ary ચલનો પ્રારંભિક કિંમત તરીકે ઉપયોગ કરીને સ્ટ્રિંગ ઓબ્જેક્ટ બનાવે છે.
- String (char ary[], int start, int len) – આપેલા પ્રથમ ચલ aryના start સ્થાનથી len જેટલા ઘટકોનો સ્ટ્રિંગ ઓબ્જેક્ટ બનાવે છે.
- String (String strObj) – ચલ તરીકે આપેલા ઓબ્જેક્ટ જેવો જ સ્ટ્રિંગ ઓબ્જેક્ટ બનાવે છે.
- String (string literal) – ચલ તરીકે આપેલા string literalને નિર્દેશ કરતો સ્ટ્રિંગ ઓબ્જેક્ટ બનાવે છે.

આકૃતિ 9.15માં વિવિધ પ્રકારના String ક્લાસના કન્સ્ટ્રક્ટરનો ઉપયોગ દર્શાવેલ છે. ઓબ્જેક્ટ બનાવતા સમયે new પ્રક્રિયકનો ઉપયોગ ભૂલતા નહીં.

```
String1.java - SciTE
File Edit Search View Tools Options Language Buffers Help

1 String1.java
/* creating String Objects */
class String1
{
    public static void main (String [] s)
    {
        char name [] = { 'j', 'a', 'v', 'a' };

        String str1 = new String();
        String str2 = new String(name);
        String str3 = new String(name, 1, 2);
        String str4 = new String(str3);
        String str5 = new String("java");

        System.out.println ("str1 is " + str1);
        System.out.println ("str2 is " + str2);
        System.out.println ("str3 is " + str3);
        System.out.println ("str4 is " + str4);
        System.out.println ("str5 is " + str5);
    } // main
} // class

>javac String1.java
>Exit code: 0
>java -cp . String1
str1 is
str2 is java
str3 is av
str4 is av
str5 is java
>Exit code: 0
```

આકૃતિ 9.15 : વિવિધ કન્સ્ટ્રક્ટરના ઉપયોગ વડે string ક્લાસના ઓબ્જેક્ટ બનાવવા

જાવામાં અક્ષરનો સંગ્રહ કરવા માટે અક્ષર દીઠ બે બાઈટ વપરાય છે. જગ્યા બચાવવા માટે જો ASCII અક્ષરો હોય, તો char પ્રકારના ડેટા-એરે વાપરવાને બદલે bytes પ્રકારના ડેટા-એરેનો ઉપયોગ કરવો જોઈએ. આ માટે આપણે કન્સ્ટ્રક્ટરમાં ચલ તરીકે byteનો એરે વાપરી શકીએ. આકૃતિ 9.15માં આપેલા પ્રોગ્રામના એરે char name[] ને બદલી byte name[] કરો અને પ્રોગ્રામનો અમલ કરવા કોશિશ કરો.

નોંધ : લિટરલ (literals)નો મેમરીમાં સંગ્રહ કરવામાં આવે છે. જ્યારે બે સ્ટ્રિંગ-ઓબ્જેક્ટ એક્સરખા સ્ટ્રિંગ લિટરલ વાપરીને બનાવ્યા હોય, ત્યારે બીજા ઓબ્જેક્ટ માટે મેમરીમાં જગ્યા ફાળવાતી નથી. બંને ઓબ્જેક્ટ એક જ મેમરી સ્થાનનો ઉલ્લેખ કરે છે.

જ્યારે સ્ટ્રિંગ ઓબ્જેક્ટ new પ્રક્રિયક વાપરીને બનાવવામાં આવે છે, ત્યારે બંને સ્ટ્રિંગ એક્સમાન હોય, તોપણ મેમરીમાં અલગ જગ્યા ફાળવવામાં આવે છે. આકૃતિ 9.16માં આપેલું કોડલિસ્ટિંગ આ પ્રકારનું ઉદાહરણ છે.

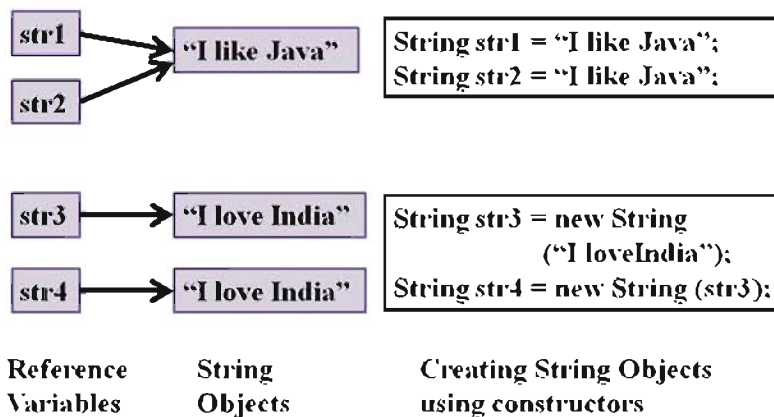
```
String2.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 String2.java
/* String Objects */
class String2
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String str2 = "I like Java";
        String str3 = new String("I love India");
        String str4 = new String(str3);

        System.out.println ("str1==str2: "
            + (str1==str2));
        System.out.println ("str3==str4: "
            + (str3==str4));

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str3: " + str3);
        System.out.println ("str4: " + str4);
    } // main
} // class
> javac String2.java
> Exit code: 0
> java -cp . String2
str1==str2: true
str3==str4: false
str1: I like Java
str2: I like Java
str3: I love India
str4: I love India
> Exit code: 0
```

આકૃતિ 9.16 : સ્ટ્રિંગ લિટરલ અને new પ્રક્રિયક વાપરીને એક્સરખા સ્ટ્રિંગ ઓબ્જેક્ટ બનાવવા

આકૃતિ 9.16માં આપેલા કોડલિસ્ટિંગમાં "str1 == str2" એ str1 અને str2 એમ બે રેફરન્સ ચલમાં સંગ્રહ કરેલી વિગતની તુલના કરે છે અને નહિ કે str1 અને str2 વડે નિર્દેશ કરેલા ઓબ્જેક્ટની. આ બાબતની નોંધ કરો. અહીં str1 અને str2 ઓબ્જેક્ટ new પ્રક્રિયક વિના અને એક્સરખા સ્ટ્રિંગ લિટરલ વાપરીને બનાવ્યા છે. આથી બંને ચલ str1 એ બનાવેલા ઇન્સ્ટન્સનો જ નિર્દેશ કરે છે, આકૃતિ 9.17માં દર્શાવ્યા પ્રમાણે str2 ઓબ્જેક્ટ માટે અલગ મેમરીની ફાળવણી કરવામાં આવી નથી. આ જ પ્રોગ્રામમાં str3 અને str4 સ્ટ્રિંગ ઓબ્જેક્ટ new પ્રક્રિયકનો ઉપયોગ કરીને બનાવ્યા છે. બંને ઓબ્જેક્ટમાંની માહિતી સરખી છે, પરંતુ બંને અલગ-અલગ મેમરીસ્થાનનો સંદર્ભ ધરાવે છે. આ બંને ઓબ્જેક્ટ માટે અલગ-અલગ મેમરીની ફાળવણી કરવામાં આવી છે, જુઓ આકૃતિ 9.17.



આકૃતિ 9.17 : એક્સમાન સ્ટ્રિંગ

સ્ટ્રિંગ ક્લાસમેથડ (String Class Methods)

બે સ્ટ્રિંગની તુલના કરવી, સ્ટ્રિંગની લંબાઈ મેળવવી, બે સ્ટ્રિંગને જોડવી, સ્ટ્રિંગમાંથી આંશિક સ્ટ્રિંગ (સબસ્ટ્રિંગ) મેળવવી, સ્ટ્રિંગને પરાવર્તિત કરવી, સ્ટ્રિંગનું વિભાજન કરવું, અક્ષર અથવા અક્ષરસમૂહને સ્ટ્રિંગમાં શોધવા વગેરે કાર્યો માટે સ્ટ્રિંગ-ક્લાસ કેટલીક મેથડ પૂરી પાડે છે. આપણે પ્રોગ્રામ દ્વારા કેટલીક મેથડનો ઉપયોગ કરવાના ઉદાહરણ જોઈશું. આકૃતિ 9.18માં લખેલો પ્રોગ્રામ સ્ટ્રિંગ અથવા સબસ્ટ્રિંગની સરખામણી કરવા માટેની વિવિધ રીત દર્શાવે છે.

```

class StringComparison
{
    public static void main (String [] s)
    {
        String str1 = "India is great";
        String str2 = "INDIA is Great";

        System.out.println ("str1: " + str1);
        System.out.println ("str2: " + str2);
        System.out.println ("str1.equals(str2): "
            + (str1.equals(str2)));
        System.out.println ("str1.equalsIgnoreCase(str2): "
            + (str1.equalsIgnoreCase(str2)));
        System.out.println ("str1.compareTo(str2): "
            + (str1.compareTo(str2)));
        System.out.println ("str1.compareToIgnoreCase(str2): "
            + (str1.compareToIgnoreCase(str2)));
    } // main
} // class

```

```

> javac StringComparison.java
> Exit code: 0
> java -cp . StringComparison
str1: India is great
str2: INDIA is Great
str1.equals(str2): false
str1.equalsIgnoreCase(str2): true
str1.compareTo(str2): 32
str1.compareToIgnoreCase(str2): 0
> Exit code: 0

```

આકૃતિ 9.18 : સ્ટ્રિંગ ક્લાસની મેથડનો ઉપયોગ કરી સ્ટ્રિંગની તુલના

કોષ્ટક 9.1માં સ્ટ્રિંગની તુલના કરવા માટેની વિવિધ મેથડનું વર્ણન અને વાક્યરચના દર્શાવી છે.

મેથડ	વર્ણન
boolean equals (String str)	મેથડકોલ કરતી સ્ટ્રિંગ અને પ્રાયલ str (ઓબ્જેક્ટ) સરખાં હોય, તો true પરત કરે છે.
boolean equalsIgnoreCase (String str)	મેથડકોલ કરતી સ્ટ્રિંગ અને પ્રાયલ str ઓબ્જેક્ટના અક્ષરોના કેસ (case) (કેપિટલ અને બીજી એબીસીડી)ની અવગણના કરીને તુલના કરે છે. જો બંને સરખાં હોય, તો true પરત કરે છે. (case insensitive)
int compareTo (String str)	જો મેથડકોલ કરનાર સ્ટ્રિંગ ઓબ્જેક્ટ પ્રાયલ strની તુલનાએ સમાન હોય, મોટી હોય અથવા નાની હોય, તો અનુક્રમે 0, >0, <0 પૂર્ણાંક સંખ્યા પરત કરે છે.
int compareToIgnoreCase (String str)	compareTo મેથડ પ્રમાણે જ પરંતુ case પ્રત્યે અસંવેદનશીલ

કોષ્ટક 9.1 : સ્ટ્રિંગની તુલના માટે સ્ટ્રિંગ ક્લાસની વિવિધ મેથડ

સ્ટ્રિંગ ક્લાસ નીચે જણાવેલાં અન્ય કાર્યો કરવા માટે અનેક મેથડ પૂરી પાડે છે. આમાંની કેટલીક કોષ્ટક 9.2માં આપેલી છે.

- સ્ટ્રિંગનો અમુક ભાગ અલગ કરવો.
- અક્ષરો કે શબ્દસમૂહ (સબસ્ટ્રિંગ) બદલવા.
- સ્ટ્રિંગનું સબસ્ટ્રિંગમાં વિભાજન કરવું.
- અક્ષરોની સંખ્યા મેળવવી.
- આપેલા ઈન્ડેક્સસ્થાન પરનો અક્ષર મેળવવો.

- સ્ટ્રિંગને bytes પ્રકારના એરેમાં પરિવર્તિત કરવી.
- સ્ટ્રિંગના અક્ષરોને સ્મોલ અથવા કેપિટલ કરવા.
- સ્ટ્રિંગના અંતમાં બીજી સ્ટ્રિંગ ઉમેરવી.
- સ્ટ્રિંગ કે તેના ભાગની નકલ કરવી.

મેથડ	વર્ણન
int length()	મેથડકોલ કરતી સ્ટ્રિંગ ઓબ્જેક્ટમાં રહેલા અક્ષરોની સંખ્યા પરત કરે છે.
char indexAt(int index)	મેથડકોલ કરતી સ્ટ્રિંગ ઓબ્જેક્ટમાં પ્રાયલ ઇન્ડેક્સ સ્થાને આવેલા અક્ષરને પરત કરે છે. ઇન્ડેક્સ શૂન્યથી ગણાય છે.
byte [] getBytes()	મેથડકોલ કરતી સ્ટ્રિંગના અક્ષરોને byte એરેમાં પરત કરે છે.
void getChars (int fromIndx, int toIndx, char target [], int targetIndx)	મેથડકોલ કરતી સ્ટ્રિંગ ઓબ્જેક્ટના fromIndx સ્થાનથી toIndx-1 સ્થાન સુધીના અક્ષરોની લક્ષ્ય એરેમાં targetIndx સ્થાન પછી નકલ કરે છે.
String concat(String str)	મેથડકોલ કરતી સ્ટ્રિંગના અંતમાં પ્રાયલ strના અક્ષરોને ઉમેરીને સ્ટ્રિંગ ઓબ્જેક્ટ પરત કરે છે.
String toLowerCase()	મેથડકોલ કરતી સ્ટ્રિંગના બધા જ અક્ષરોને સ્મોલ અક્ષરોમાં બનાવીને સ્ટ્રિંગ પરત કરે છે.
String toUpperCase()	મેથડકોલ કરતી સ્ટ્રિંગના બધા જ અક્ષરોને કેપિટલ અક્ષરમાં પરિવર્તિત કરીને સ્ટ્રિંગ પરત કરે છે.

કોષ્ટક 9.2 : સ્ટ્રિંગકલાસની અન્ય મેથડ

આકૃતિ 9.19માં આપેલાં કોડલિસ્ટિંગમાં આમાંની કેટલીક મેથડનો ઉપયોગ દર્શાવ્યો છે.

```

StringConversion.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 StringConversion.java
/* String conversions */
class StringConversion
{
    public static void main (String [] s)
    {
        String str1 = "I like Java";
        String strAry[] = new String[5];
        byte byteAry[] = new byte[20];

        System.out.println ("Given string is \"" + str1 + "\"");
        System.out.println ("String in lowercase: \""
            + str1.toLowerCase()+"\"");
        System.out.println ("String in uppercase: \""
            + str1.toUpperCase()+"\"");

        System.out.println("\nString converted to array of bytes");
        byteAry = str1.getBytes();
        for (int i=0; i<byteAry.length; i++)
            System.out.println ( byteAry[i] );
    } // main
} // class

```

```

>javac StringConversion.java
>Exit code: 0
>java -cp . StringConversion
Given string is "I like Java"
String in lowercase: "i like java"
String in uppercase: "I LIKE JAVA"

String converted to array of bytes
73
32
108
105
107
101
32
74
97
118
97
>Exit code: 0

```

આકૃતિ 9.19 : સ્ટ્રિંગને પરિવર્તિત કરતી મેથડનો ઉપયોગ

નોંધ : એરે ચલ માટે length એ એટ્રિબ્યૂટ કે પ્રોપર્ટી છે, જ્યારે સ્ટ્રિંગ ઓબ્જેક્ટ માટે length એ મેથડ છે. આથી સ્ટ્રિંગ ઓબ્જેક્ટ સાથે length મેથડ વાપરતા સમયે () કૌંસનો ઉપયોગ કરવો જરૂરી છે.

જાવામાં સ્ટ્રિંગ ક્લાસમાં સ્ટ્રિંગને ઉલટાવવા માટે કોઈ મેથડ ઉપલબ્ધ નથી. આથી, આપણે સ્ટ્રિંગને ઉલટાવવા માટે પ્રોગ્રામ લખીએ. આકૃતિ 9.20માં આપેલા કોડલિસ્ટિંગ અને તેના આઉટપુટનો અભ્યાસ કરો.

```

StringReverse.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 StringReverse.java
/* Reverse String */
class StringReverse
{
    public static void main (String [] s)
    {
        String str = new String("I Like Java");
        System.out.println ("Given String: " + str);
        System.out.println ("Reverse String: " + reverseStr(str));
    } // end main

    static String reverseStr (String str) // reverse string
    {
        int len = str.length();
        byte byteAry[] = new byte[len];
        byteAry = str.getBytes(); // convert to byte array
        for (int i=0, j=len-1; i<len/2; i++, j--) // exch half array
        {
            byte tmp = byteAry[i];
            byteAry[i] = byteAry[j];
            byteAry[j]=tmp;
        }
        return new String(byteAry); // used constructor
    }
} // class
>javac StringReverse.java
>Exit code: 0
>java -cp . StringReverse
Given String: I Like Java
Reverse String: avaJ ekiL I
>Exit code: 0

```

આકૃતિ 9.20 : સ્ટ્રિંગને ઉલટાવવી

અહીં, આપણે વપરાશકર્તા વ્યાખ્યાયિત reverseStr નામની મેથડ લખી છે, જે નીચે જણાવેલાં પગલાં પ્રમાણે સ્ટ્રિંગને ઉલટાવે છે :

1. સ્ટ્રિંગક્લાસની length() મેથડ વાપરીને સ્ટ્રિંગમાં રહેલા અક્ષરોની સંખ્યા (સ્ટ્રિંગની લંબાઈ) નક્કી કરો.
2. સ્ટ્રિંગક્લાસની getBytes() મેથડના ઉપયોગથી સ્ટ્રિંગને byte એરેમાં પરિવર્તિત કરો.
3. byte એરેમાં ડાબી બાજુના અડધા ઘટકોની (ડાબી બાજુથી જમણી બાજુ તરફ) જમણી બાજુના અડધા ઘટકો (છેલ્લે જમણી બાજુથી ડાબી બાજુ તરફ) સાથે અદલાબદલી કરો.
4. કન્સ્ટ્રક્ટર વાપરીને byte એરેમાંથી સ્ટ્રિંગ ઓબ્જેક્ટ બનાવો અને તેને પરત કરો.

Date ક્લાસ (Date Class)

આપણે જાવામાં ઉપલબ્ધ String ક્લાસ અને Arrays ક્લાસનો અભ્યાસ કર્યો. જાવાલાઈબ્રેરીમાં java.util નામના પેકેજમાં Date ક્લાસ પૂરો પાડે છે. Date ક્લાસ તારીખ અને સમય બંનેનો સમાવેશ કરે છે અને મિલિસેકન્ડ સુધીની ચોકસાઈ સહિત કિંમત જણાવે છે. આકૃતિ 9.21માં કોડલિસ્ટિંગ અને તેના આઉટપુટ સાથે Date ક્લાસનો ઉપયોગ દર્શાવ્યો છે.

```

Date1.java - SciTE
File Edit Search View Tools Options Language Buffers Help
1 Date1.java

import java.util.Date;

class Date1
- {
    public static void main(String args[])
    {
        Date date1 = new Date(); // current date, time
        // if not used: import java.util.Date; use next statement
        java.util.Date date2 = new java.util.Date();
        System.out.println ("date1: Date & Time: " + date1);
        System.out.println ("date2: Date & Time: " + date2);

        System.out.println (
            "Elapsed time since Jan 1, 1970 is\n\t"
            + date1.getTime() + " milliseconds");
        date1.setTime(date1.getTime() + 10000000);
        System.out.println ("DateTime after 1 crore milliseconds\n\t"
            + date1.toString());
    } // end main()
} // end class

>javac Date1.java
>Exit code: 0
>java -cp . Date1
date1: Date & Time: Thu Oct 31 08:50:59 IST 2013
date2: Date & Time: Thu Oct 31 08:50:59 IST 2013
Elapsed time since Jan 1, 1970 is
1383189659935 milliseconds
DateTime after 1 crore milliseconds
Thu Oct 31 11:37:39 IST 2013
>Exit code: 0

```

આકૃતિ 9.21 : Date ક્લાસનો ઉપયોગ

કોષ્ટક 9.3માં Date ક્લાસની કેટલીક મેથડની યાદી આપેલી છે.

મેથડ	વર્ણન
Date()	સિસ્ટમના તત્કાલીન સમયનો ઉપયોગ કરીને Date ઓબ્જેક્ટ બનાવે છે.
Date(long elapsedTime)	જાન્યુઆરી, 1, 1970 GMTથી પ્રાયલમાં આપેલા સમય વચ્ચેનો સમયગાળો મિલિસેકન્ડમાં ગણીને Date ઓબ્જેક્ટ બનાવે છે.
String toString()	Date ઓબ્જેક્ટના તારીખ અને સમયને સ્ટ્રિંગમાં રજૂ કરી તે સ્ટ્રિંગ પરત કરે છે.
long getTime()	જાન્યુઆરી 1, 1970 GMTથી અત્યાર સુધીની મિલિસેકન્ડ પરત કરે છે.
void setTime(long elapsedTime)	પ્રાયલમાં આપેલા વિતેલા સમયનો ઉપયોગ કરી નવી તારીખ અને સમય સેટ કરે છે.

કોષ્ટક 9.3 : Date ક્લાસની વિવિધ મેથડ

Calendar ક્લાસ (Calendar Class)

Date ક્લાસની જેમ Calendar ક્લાસ પણ java.util પેકેજમાં આપવામાં આવ્યો છે. આ ક્લાસ વર્ષ, માસ, તારીખ, કલાક, મિનિટ અને સેકન્ડ જેવી કેલેન્ડરની વિગતવાર માહિતી મેળવવા માટે વાપરી શકાય છે. અહીં, આપણે calendar ક્લાસના સબક્લાસ GregorianCalendarના ઉપયોગ બાબત જોઈશું.

આકૃતિ 9.22માં ઉદાહરણ રૂપે Calendar ક્લાસનો ઉપયોગ કરતો પ્રોગ્રામ અને તેનું આઉટપુટ દર્શાવ્યું છે. અહીં તારીખ અને સમયના વિવિધ ઘટક પ્રદર્શિત કરવા માટે વપરાશકર્તાએ વ્યાખ્યાયિત display() ક્લાસમેથડનો ઉપયોગ કર્યો છે, તેની નોંધ કરો.

```

Calendar1.java - ScITE
File Edit Search View Tools Options Language Buffers Help
1 Calendar1.java
import java.util.*; // for Date, Calendar, GregorianCalendar class
public class Calendar1
{
    public static void main(String s[])
    {
        System.out.println ("Current Date & Time: " + new Date()); // current date,time
        //using current date & time
        Calendar calendar1 = new GregorianCalendar();
        display(calendar1);
        //using given year,month,day
        Calendar calendar2 = new GregorianCalendar(2013, 11, 17);
        display(calendar2);
        //using given date-year, month, day and time-hour, minute, second
        Calendar calendar3 = new GregorianCalendar(2013,11,20,210,2);
        display(calendar3);
    } // end main()
    static void display(Calendar calendar)
    {
        // using Calendar constants
        System.out.println ("Year:" + calendar.get(Calendar.YEAR));
        System.out.println ("Month:" + calendar.get(Calendar.MONTH));
        System.out.println ("Date:" + calendar.get(Calendar.DATE));
        System.out.println ("Hour (12 hour):" + calendar.get(Calendar.HOUR));
        if (calendar.get(Calendar.AM_PM) == 1)
            System.out.println (" p.m.");
        else
            System.out.println (" a.m.");
        System.out.println ("Hour (24 hour):" + calendar.get(Calendar.HOUR_OF_DAY));
        System.out.println ("Minute:" + calendar.get(Calendar.MINUTE));
        System.out.println ("Second:" + calendar.get(Calendar.SECOND));
        System.out.println ("Day of Week:");
        switch (calendar.get(Calendar.DAY_OF_WEEK))
        {
            case 1: System.out.println ("Sunday"); break;
            case 2: System.out.println ("Monday"); break;
            case 3: System.out.println ("Tuesday"); break;
            case 4: System.out.println ("Wednesday"); break;
            case 5: System.out.println ("Thursday"); break;
            case 6: System.out.println ("Friday"); break;
            case 7: System.out.println ("Saturday"); break;
            default: System.out.println ("Error...");
        }
    } // end display()
} // end class

```

```

> javac Calendar1.java
> Exit code: 0
> java -cp . Calendar1
Current Date & Time: Thu Oct 31 07:48:28 IST 2013
Year: 2013
Month: 9
Date: 31
Hour (12 hour): 7 a.m.
Hour (24 hour): 7
Minute: 48
Second: 28
Day of Week: Thursday
Year: 2013
Month: 11
Date: 17
Hour (12 hour): 0 a.m.
Hour (24 hour): 0
Minute: 0
Second: 0
Day of Week: Tuesday
Year: 2013
Month: 11
Date: 28
Hour (12 hour): 6 p.m.
Hour (24 hour): 18
Minute: 2
Second: 0
Day of Week: Saturday
> Exit code: 0

```

આકૃતિ 9.22 : Calendar ક્લાસ અને તેના અચલનો ઉપયોગ

get મેથડની જેમ set મેથડ વાપરીને Calendar ક્લાસના ક્ષેત્ર અચલ (field constants)ની કિંમત આપણે સેટ કરી શકીએ છીએ. ઉદાહરણ તરીકે, જો Calendar ક્લાસનો calendar ઓબ્જેક્ટ હોય, તો 'calendar.set(Calendar.DATE, 20)' મેથડકોલનો અમલ કરતાં મહિનાની તારીખ 20 સેટ થશે.

કોષ્ટક 9.4માં Calendar ક્લાસમાં વ્યાખ્યાયિત પૂર્ણાંક સંખ્યાવાળા અચલોની યાદી આપેલી છે. આ અચલોને અર્થપૂર્ણ અને સ્વયંસ્પષ્ટ નામો આપવામાં આવેલાં છે :

અચલ	વર્ણન
YEAR	કેલેન્ડરનું વર્ષ
MONTH	કેલેન્ડરનો મહિનો (જાન્યુઆરી માટે 0 અને ડિસેમ્બર માટે 11)
DATE	મહિનાનો દિવસ
DAY_OF_MONTH	મહિનાનો દિવસ (DATE સમાન)
HOUR	12 કલાકની સંકેતલિપિ પ્રમાણે કલાક
HOUR_OF_DAY	24 કલાકની સંકેતલિપિ પ્રમાણે કલાક
MINUTE	મિનિટ
SECOND	સેકન્ડ
AM_PM	AM માટે 0 અને PM માટે 1
DAY_OF_WEEK	અઠવાડિયામાં દિવસનો ક્રમ (રવિવાર માટે 1 અને શનિવાર માટે 7)
WEEK_OF_MONTH	મહિનામાં અઠવાડિયાનો ક્રમ
WEEK_OF_YEAR	વર્ષમાં અઠવાડિયાનો ક્રમ
DAY_OF_YEAR	વર્ષમાં દિવસનો ક્રમ (પ્રથમ દિવસ માટે 1)

કોષ્ટક 9.4 : Calendar ક્લાસમાં વ્યાખ્યાયિત અચલ

સારાંશ

આ પ્રકરણ એરે, સ્ટ્રિંગ અને રેઈટનો ઉપયોગ કેવી રીતે કરવો, તે સમજાવે છે. એકસમાન પ્રકારના ચલોના સમૂહ માટે એરેનો ઉપયોગ થાય છે. જાવા મૂળભૂત રીતે ફક્ત 1-D એરે પૂરા પાડે છે. 1-D એરેનો એરે વાપરીને આપણે વ્યાવહારિક રીતે બહુ-પરિમાણીય એરે મેળવી શકીએ છીએ. એરેને Arrays ક્લાસના ઓબ્જેક્ટ તરીકે ગણવામાં આવે છે, તેથી એરેનું નામ એ સંદર્ભચલ (reference variable) છે, જે તેના ઘટકો જે જગ્યાએ સંગૃહીત કરવામાં આવ્યા છે તે મેમરી સ્થાનાંકનો ઉલ્લેખ કરે છે. એરે-ઓબ્જેક્ટ સાથે Arrays ક્લાસની તમામ મેથડ વાપરી શકાય છે. આવી મેથડનાં ઉદાહરણો છે : sort, fill. જાવામાં આપેલા String ક્લાસ અક્ષરોની શ્રેણી સાથે કામ કરવાનું સામર્થ્ય પૂરું પાડે છે. સ્ટ્રિંગ ઉપર કરી શકાતાં કાર્યોનાં ઉદાહરણો : સ્ટ્રિંગની સરખામણી, સ્ટ્રિંગમાં રહેલા અક્ષરોની સંખ્યા શોધવી, સ્ટ્રિંગમાં રહેલા અક્ષરોના કેસને રૂપાંતરિત કરવા. તારીખ અને સમય સાથે કામ કરવા માટે Date અને Calendar ક્લાસ આપવામાં આવ્યા છે. Calendar ક્લાસની મેથડ વાપરીને આપણે વર્ષ, માસ, તારીખ, કલાક, મિનિટ અને સેકન્ડ જેવા ઘટકોની કિંમત મેળવી શકીએ છીએ, તેમજ સેટ કરી શકીએ છીએ.

સ્વાધ્યાય

1. એરે વિશે ઉદાહરણ આપી સમજાવો.
2. 1-D અને 2-D એરે વચ્ચેનો તફાવત જણાવો.
3. નીચે જણાવેલા ક્લાસનો ઉપયોગ સમજાવો :
 - a. String
 - b. Date
 - c. Calendar
4. નીચે આપેલા વિકલ્પમાંથી સાચો વિકલ્પ જણાવો :
 - (1) નીચેનામાંથી શું એરેની પ્રારંભિક ઇન્ડેક્સ કિંમતનો ઉલ્લેખ કરે છે ?
 - (a) 0
 - (b) 1
 - (c) null
 - (d) ઉપરના તમામ વિકલ્પ
 - (2) sales[5][12] એરેમાં દ્વિતીય પરિમાણનું કદ કેટલું છે ?
 - (a) 5
 - (b) 12
 - (c) 60
 - (d) 10
 - (3) sales[5][12] એરે માટે sales.length પદાવલી શું પરત કરશે ?
 - (a) 5
 - (b) 12
 - (c) 60
 - (d) 120
 - (4) જો પ્રારંભિક કિંમતો આપ્યા વિના sales [5][12] એરે વ્યાખ્યાયિત કરવામાં આવે, તો sales[0][0]ની શરૂઆતની કિંમત શું હશે ?
 - (a) 0
 - (b) પૂર્વનિર્ધારિત કિંમત
 - (c) કમ્પાઈલેશન એરર
 - (d) 60
 - (5) String ક્લાસના ઓબ્જેક્ટમાં 'length' શું નિર્દેશ કરે છે ?
 - (a) એટ્રિબ્યૂટ
 - (b) મેથડ
 - (c) ક્લાસ વેરિયેબલ
 - (d) ક્લાસનું નામ
 - (6) જો 'str' એ String ક્લાસનો ઓબ્જેક્ટ હોય અને તેમાં "Thank GOD" માહિતી હોય, તો str.length()ની કિંમત કેટલી હશે ?
 - (a) 9
 - (b) 10
 - (c) 8
 - (d) 11

(7) જો આપણે Calendar ક્લાસની get મેથડમાં DAY_OF_WEEK પ્રાચલ તરીકે વાપરીએ, તો કયા પ્રકારની કિંમત પરત થશે ?

(a) int

(b) char

(c) String

(d) boolean

પ્રાયોગિક સ્વાધ્યાય

1. એક પ્રોગ્રામ લખો, જે દિવસના 6 am થી 6 pm સુધીના દરેક કલાકના તાપમાનની 12 કિંમતોનો સંગ્રહ કરવા માટે એરેનો ઉપયોગ કરે. આ કિંમતો આપવા માટે પ્રારંભિક કિંમતો અથવા એસાઇન્મેન્ટ વિધાનોનો ઉપયોગ કરો. દિવસનું મહત્તમ અને ન્યૂનતમ તાપમાન છાપો.
2. એક પ્રોગ્રામ લખો, જે પાંચ વેચનાર વ્યક્તિઓના અઠવાડિક વેચાણની માહિતીનો સંગ્રહ કરે. વેચનાર વ્યક્તિને તેના અઠવાડિક વેચાણના સરેરાશના પ્રમાણે અઠવાડિક પ્રોત્સાહન રકમ આપવામાં આવે છે. જો સરેરાશ અઠવાડિક વેચાણ ₹ 10,000 સુધી હોય, તો 10 %, સરેરાશ અઠવાડિક વેચાણ ₹ 10000 થી ₹ 30,000 સુધીમાં હોય, તો 15 % અને ₹ 30,000 થી વધુ હોય, તો 20 % પ્રોત્સાહન રકમ છે. બધી વેચનાર વ્યક્તિઓનું દરરોજનું કુલ વેચાણ શોધો. આ ઉપરાંત વેચનાર વ્યક્તિની અઠવાડિક પ્રોત્સાહન રકમની ગણતરી કરો.
3. એક પ્રોગ્રામ લખો, જે આપેલી સ્ટ્રિંગ palindrome છે કે નહીં તે જણાવે. (સ્ટ્રિંગ palindrome કહેવાય, જો તેને ઉલટાવવાથી પણ સરખી રહે, જેમકે "1771" અને "madam".)
4. આજની તારીખને "DD-MMM-YYYY" સ્વરૂપમાં છાપવાનો પ્રોગ્રામ લખો. ઉદાહરણ તારીખ, 17th November 2013ને 17-Nov-2013 તરીકે છાપો. આ ઉપરાંત અઠવાડિયાનો દિવસ શબ્દમાં છાપો.
5. તમારી જન્મતારીખ અને સમય પ્રમાણે તારીખ અને સમય સેટ કરવાનો પ્રોગ્રામ લખો. તમારા જન્મદિવસે અઠવાડિયાનો કયો દિવસ હતો તે છાપો.

