**Массивтер**

Массив – белгілі бір ұзындықтағы бір типтегі мәндер жиынынан тұратын объект контейнер. Массив ұзындығы массив құрылған кезде орнатылады. Құрылғаннан кейін оның ұзындығы бекітіледі.



Массивтегі әрбір жеке бөлшектер элемент деп аталады, және әрбір элементке оның сандық реттік номері арқылы қол жеткізуге болады. Суретте көрсетілгендей номерлеу 0-ден басталады. Сондықтан, 9-шы элементке 8-ші реттік номері арқылы қатынайды.

Төмендегі бағдарлама бүтін сандар массивін құрады, массивке кейбір мәндерді енгізеді, және массивтің әрбір элементін стандартты шығысқа басып шығарады.

class ArrayDemo {

public static void main(String[] args) {

// declares an array of integers

int[] anArray;

// allocates memory for 10 integers

anArray = new int[10];

// initialize first element

anArray[0] = 100;

// initialize second element

anArray[1] = 200;

// and so forth

anArray[2] = 300;

anArray[3] = 400;

anArray[4] = 500;

anArray[5] = 600;

anArray[6] = 700;

anArray[7] = 800;

anArray[8] = 900;

anArray[9] = 1000;

System.out.println("Element at index 0: "

+ anArray[0]);

System.out.println("Element at index 1: "

+ anArray[1]);

System.out.println("Element at index 2: "

+ anArray[2]);

System.out.println("Element at index 3: "

+ anArray[3]);

System.out.println("Element at index 4: "

+ anArray[4]);

System.out.println("Element at index 5: "

+ anArray[5]);

System.out.println("Element at index 6: "

+ anArray[6]);

System.out.println("Element at index 7: "

+ anArray[7]);

System.out.println("Element at index 8: "

+ anArray[8]);

System.out.println("Element at index 9: "

+ anArray[9]);

}

}

Бағдарламаның нәтижесі:

Element at index 0: 100

Element at index 1: 200

Element at index 2: 300

Element at index 3: 400

Element at index 4: 500

Element at index 5: 600

Element at index 6: 700

Element at index 7: 800

Element at index 8: 900

Element at index 9: 1000

**Массивті белгілеу үшін айнымалы жариялау**

Бағдарламада төмендегі тармақты кодпен массив жариялау:

// declares an array of integers

int[] anArray;

Басқа типтегі айнымалыны жариялау секілді массивті де жариялау екі бөліктен тұрады: массив типі және массив атауы. Массив типі type[]арқылы жазылады, мұнда, type массив элементерінің деректер типі, ал тік жақшалар айнымалының массивтен тұратындығын көрсететін арнайы таңбалар. Ал массив ұзындығы оның типінің бөлігі емес, сондықтан тік жақшалар бос болады. Массив атауы айнымалы атауы заңдалықтарына сәйкес кез-келген болуы мүмкін.

Сол секілді басқа типтегі массивтерді де құруға болады:

byte[] anArrayOfBytes;

short[] anArrayOfShorts;

long[] anArrayOfLongs;

float[] anArrayOfFloats;

double[] anArrayOfDoubles;

boolean[] anArrayOfBooleans;

char[] anArrayOfChars;

String[] anArrayOfStrings;

Сонымен қатар тік жақшаларды массив атауынан кейін де қойса болады.

// this form is discouraged

float anArrayOfFloats[];

Сонда да, келісім (конвенция) бойынша бұндай формада қолдануға кеңес бермейді. Тік жақшалар массивтің түрін айқындауы және қосылып бірге қолдануы қажет.

**Массивті құру, инициализациялау және элементтеріне қатынау**

Массивті құрудың бірден-бір жолы new операторы арқылы жүзеге асады. Алдыңғы ArrayDemo бағдарламасы мысалында орналасқан төмендегі мәлімдеме 10 бүтін санды сақтауға жетерлік жады бөледі және массивті anArray айнымалысына тағайындайды.

// create an array of integers

anArray = new int[10];

Егер бұл мәлімдеме жоқ болса, онда компилятор төмендегідей қате шығарады және компиляция іске аспайды:

ArrayDemo.java:4: Variable anArray may not have been initialized.

Келесі бірнеше жолдар массивтің әрбір элементіне мән тағайындайды:

anArray[0] = 100; // initialize first element

anArray[1] = 200; // initialize second element

anArray[2] = 300; // and so forth

Массивтің әрбір элементіне оның сандық индексі арқылы қол жеткізуге болады.

System.out.println("Element 1 at index 0: " + anArray[0]);

System.out.println("Element 2 at index 1: " + anArray[1]);

System.out.println("Element 3 at index 2: " + anArray[2]);

Сонымен қатар массивті құру және инициализация жасау үшін мынадай қысқа синтаксис қолдана аласыз:

int[] anArray = {

100, 200, 300,

400, 500, 600,

700, 800, 900, 1000

};

Бұл жерде массивтің ұзындығы фигуралы жақшада көрсетілген және үтір арқылы бөлінген мәндер санымен анықталады.

Сонымен қатар бірнеше екі немесе одан көп тік жақшаларды пайдалана отырып, мынадай String[][] атаулы массивтер массивін (көп өлшемді массив) жариялауға болады. Соған сәйкес, массивтің әрбір элементі мәндер санына сәйкес индексі арқылы қол жетімді.

Java бағдарламалау тілінде көп өлшемді массив дегеніміз компонеттерінің өзі массив болып табылатын массив. Бұл C және Fortran тіліндегіден өзгешелеу. Себебі, төмендегі MultiDimArrayDemo бағдарламасында көрсетілгендей массивтер ұзындығы әртүрлі болуы мүмкін:

class MultiDimArrayDemo {

public static void main(String[] args) {

String[][] names = {

{"Mr. ", "Mrs. ", "Ms. "},

{"Smith", "Jones"}

};

// Mr. Smith

System.out.println(names[0][0] + names[1][0]);

// Ms. Jones

System.out.println(names[0][2] + names[1][1]);

}

}

Бағдарламаның орындалу нәтижесі:

Mr. Smith

Ms. Jones

Және соңына, кез-келген массивтің өлшемін анықтау үшін length сипатын қолдана аласыз. Келесі код стандартты баспаға массив ұзындығын шығарады:

System.out.println(anArray.length);

**Массивтерді көшіру**

System класында бір массивтен екінші бір массивке деректерді тиімді көшіру үшін қоллдануға болатын arraycopy әдісі бар:

public static void arraycopy(Object src, int srcPos,

Object dest, int destPos, int length)

Екі Object аргументтері, біріншісі деректері көшірілетін массивті және екіншісі деректерді көшіру керек массивті айқындайды. Үш int аргументтері, біріншісі бастапқы массивтің көшіру басталатын, ал екіншісі көшірілетін массивке көшіру басталатын позициялары және үшіншісі көшірілетін массив элементтерінің санын көрсетеді.

Келесі ArrayCopyDemo бағдарламасында таңбалар тізімінен тұратын массив жарияланады. Бұл бағдарлама массивтің ішкі компоненттерінің тізімін екінші бір массивке көшіруге System.arraycopy әдісін қолданады:

class ArrayCopyDemo {

public static void main(String[] args) {

char[] copyFrom = { 'd', 'e', 'c', 'a', 'f', 'f', 'e',

'i', 'n', 'a', 't', 'e', 'd' };

char[] copyTo = new char[7];

System.arraycopy(copyFrom, 2, copyTo, 0, 7);

System.out.println(new String(copyTo));

}

}

Бағдарламаның орындалу нәтижесі:

caffein

**Массивтер манипуляциясы**

Массивтер бағдарламалауда жиі қолданылатын өте пайдалы түсінік. Java SE массивтерге байланысты ең көп тараған манипуляцияларды орындау үшін әдістер ұсынады. Мысалыға, ArrayCopyDemo бағдарламасы көшірілетін массивтің әрбір элементін алып, оларды көшірілетін массивке салу кодын тікелей жазу орнына орнына, System класының arraycopy әдісін қолданады. Бұл бағдарламашыға әдісті шақыру үшін бір жолды кодты қолдануға мүмкіндік береді және негізгі үрдіс сахна артында жүзеге асырылады.

Ыңғайлы болу үшін Java SE массивке манипуляциялар орындау үшін java.util.Arrays класында (көшіру, сұрыпиау және іздеу секілді жалпы міндеттері) бар бірнеше әдістерді ұсынады. ArrayCopyOfDemo бағдарламасында көрсетілгендей, алдыңғы бағдарламаны java.util.Arrays класындағы copyOfRange әдісін қолдану арқылы өзгертуге болады. copyOfRange әдісін қолданудың айырмашылығы: шаыру әдісін қолданар алдында деректер көшірілетін массивті құруды талап етпейді, себебі ол массив әдістің өзімен қайтарылады.

class ArrayCopyOfDemo {

public static void main(String[] args) {

char[] copyFrom = {'d', 'e', 'c', 'a', 'f', 'f', 'e',

'i', 'n', 'a', 't', 'e', 'd'};

char[] copyTo = java.util.Arrays.copyOfRange(copyFrom, 2, 9);

System.out.println(new String(copyTo));

}

}

Көріп отырғаныңыздай, бірнеше ғана жол кодты талап етсе де, бұл бағдарламаның орындалу нәтижесі алдыңғы бағдарлама нәтижесімен бірдей. Ескеріңіз, copyOfRange әдісінің екінші параметрі көшіру басталатын бастапқы аралық индексі, қоса есептегенде, ал үшінші параметрі аралықтың соңғы қамтылмайтын индексі. Бұл мысалда, көшірілуге тиіс массив элементтері индексі 9 элементті (а таңбасын) қамтымайды.

java.util.Arrays класы әдістерімен қамтамасыз ететін кейбір басқа да пайдалы операциялар:

* Массивтің белгелі бір мәні бойынша оның орналасқан индексін анықтау (binarySearch әдісі)
* Екі массивті салыстыру, тең немесе тең еместігін анықтау, (equals әдісі)
* Белгелі бір мәнді әрбір индекс элементіне орналастырып, массивті толтыру (fill әдісі)
* Өсу реті бойынша массивті сұрыптау. Бұл sort әдісін қолданатын кезекті немесе Java SE 8-де енгізілген parallelSort әдісін қолданатын қосарлы жасалуы мүмкін.

Үлкен массивтерді көп процессорлы жүйелерде параллельді сұрыптау массивті кезекті сұрыптауға қарағанда жылдамырақ.