 

**Advanced Placement Computer Science**

[Shenendehowa HS](http://www.shenet.org/shen-high-school/)[mr Hanley](http://hanley.co.nr)

**Unit 6: Arrays/ArrayLists**

**Lesson: One and Two D ArraysFILLED\_OUT**

***Last Updated:*** *11/20/2017*

Lesson: Parameter Passing Mechanisms

*Last Updated: 100/11001/1100*

Arrays are a powerful tool, they allow a programmer to store data using

a single name and a subscript(number that goes in the [ ]’s to select an element)

ARRAYS ARE ZERO BASED: they start numbering at 0 and go through n-1

To create an array of primitives

int[ ] nums = new int[10]; //or alternatively int nums[ ] = new int[10];

You create an array of doubles to store the conversion rates for 10 currencies to USD

## double[ ] rates = new double[10];

To access individual elements inside the array

rates[0] = 1.06;

NOTE: Valid indices are from 0 to 9(.length-1 )

To create an array of references

Color[ ] colors = new Color[100];

To create objects that the references point to

for(int i=0; i<100; i+){

colors[i] = new Color(20, i\*2, 100); //Note: Red, Green and Blue Intensity

}

To figure out the size of the array

.length

colors.length

Create an array of 100 doubles and write a loop that will assign the square roots of the integers from 1-100 into your array. Call your array squares

double [] squares = new double[100];

for(int i =0; i<100; i++)

squares[i] = Math.sqrt(i+1);

Create an array of 256 Color objects and write a loop that will assign an array of 256 Colors using all of the green intensities but leave the red and blue intensities at 0. (to refresh your memory, a Color object is created by using Color newCol = new Color(1,5,10);)

//where 1 = red intensity from 0-255, 5 = green intensity from 0-255, 10 = blue intensity from 0-255

Color [] cols = new Color[256];

for(int i=0, r=0, g=0, b=0; i < 256; i++, g++)

cols[i] = new Color(r, g, b);

Arrays can also be created by specifying the initial values and NOT specifying the size

int [ ] daysInMonth = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

What will System.out.println(daysInMonth[5]) output to the console?

Also, you can create an anonymous array object on the fly:

new boolean [ ] { true, false, false, true , true};

What is .length for the anonymous array above?

**5**

**Passing arrays to methods**

public class PunjabMC {

//preconditions: values.length > 0

//postconditions: returns the average of the numbers in the array

public double average(double[ ] values){ //finish the method

double sum=0;

for(int i =0; i < values.length; i++){  
 sum += values[i];  
 return sum/values.length;

}

}

public class Tester {

public static void main (String [ ] args) {

//calling a method and passing an array to it

PunjabMC pm = new PunjabMC();

double [ ] myNums = {5, 6, 7, 8};

double avg = pm.average(myNums);

System.out.println(avg);

}

}

**Two dimensional arrays**

You can also create two and n dimensional arrays

To create a 2 dimensional array

int [ ] [ ] maze = new int [40] [25];

Figuring out the size of the dimensions of the array

For the first dimension

maze.length

For the second dimension

maze[0].length

If there was a third dimension

maze[0] [0].length

If there was a fourth dimension

maze[0] [0] [0].length

//You try; assign all values in the maze to 1

for(int r=0; r < maze.length; r++)

for(int c=0; c < maze[0].length; c++)  
 maze[r][c] = 1 ;

Now, Let’s display the maze to the screen

for(int r=0; r < maze.length; r++){

for(int c=0; c < maze[0].length; c++)  
 System.out.print(maze[r][c]+”\t”);  
 System.out.println();

}

Arrays of primitives simply allocate contiguous memory (if its available) when the new command is issued.

You will recall this chart

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TYPE | CONTAINS | SIZE | MAX | MIN |
| boolean | boolean value | 8 bits | true | false |
| char | Unicode char | 16 bits | \uFFFF | \u0000 |
| byte | signed integer | 8 bits | 127 | -128 |
| short | signed integer | 16 bits | 32767 | -32768 |
| int | signed integer | 32 bits | 2147483647 | -2147483648 |
| long | signed integer | 64 bits | 8.223372 E +18 | -9.223372 E +18 |
| float | floating point | 32 bits | 3.402823 E +38  1.402398 E -45 | -3.402823 E +38  -1.402398 E -45 |
| double | floating point | 64 bits | 1.797693 E +308  4.940656 E -324 | -1.797693 E +308  -4.940656 E -324 |

How many bytes are allocated for the following arrays?

int myArray[ ] = new int[15];

**15 X 4 = 60**

float yourArray[ ] = new float[10];

**10 X 4 = 40**

double [ ] [ ]grid = new double[10][20];

**10 X 20 X 8 = 1600 or 1.6 Kilobyes**

What happens when you attempt to create an array of references?

**Java creates variables that point to objects that are NOT the objects themselves**

You need to loop through all of the references and use the new operator

Use an Array of Students as example

public class Student{

public String name;

public int numQuizzes, numPoints; //for simplicity, no private vars!

}

Create an array of 3000 Students called roster (loop through and make 3000 objects)  
**Student [ ] roster = new Student[3000];**

**for(int i =0; i < roster.length; i++)**

**roster[i] = new Student();**

Set the third student’s name to “Chester”, numQuizzes to 3 and numPoints to 297

**roster[2].name = “Chester”;**

**roster[2].numQuizzes = 3;**

**roster[2].numPoints = 297;**

Writing a method that returns an array

Write a method that returns an array of Strings and takes a String as a parameter that will break the parameter into its individual letters.

For example

String [] x = convertToLetters(“Bob”);

should return

[0] “B”

[1] “o”

[2] “b”

**public String [] convertToLetters(String word){  
 String[ ] temp = new String[word.length];**

**for(int i = 0; i < word.length; i++)  
 temp[i] = word.substring(i, i+1);**

**return temp;**

**}**