Introductory Programming and Object-oriented Concepts Using Java

Units 2, 4

Arrays, methods

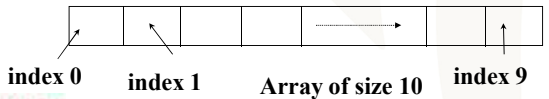
Problem

- To read in marks of 10 students, and ONLY after reading all, to print each of the marks.
- How many variables to declare to store marks?

```
int m1, m2;          //need storage for all 10 marks
m1 =55;
m2=65;
…
System.out.println(m1);
System.out.println(m2);
```


What is an Array?

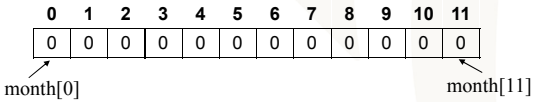
- An array is a group of contiguous memory locations that all have the same name and same type.
- It uses an integer called index/subscript to reference an element in the array. Index starts with 0.



Declaring an array

int[ ] month = new **int**[12];

- 12 integer locations created
- When you create an integer array, all the elements are initialised to 0.
- Index starts with 0



Assigning values

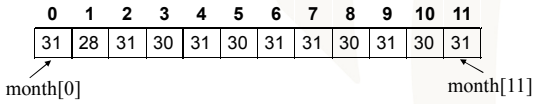
```
month[0] = 31;      // assigns 31 days to January
month[1] = 28;      // assigns 28 days to February
   …
month[11] = 31;     // assigns 31 days to December
```


```
System.out.println( month[2] );     //prints March
```


Another way to initialise an array

```
int[ ] month = {31, 28, 31, 30, 31, 30, 31,
                                    31, 30, 31, 30, 31};
```

- Size of array is determined by the number of values inside the braces.



Iterating an array

- for loop is commonly used.

```
for ( int index = 0; index < 12; index++)
{
  System.out.println ("Month " + (index+1) + " has "
                      + month[index] + " days.");
}
```

Output: Month 1 has 31 days.
Month 2 has 28 days.
….

Length of an array

- month.length returns the size of the array

```
for ( int index =0; index < month.length; index++)
{
   System.out.println ("Month " + (index+1) + " has "
                + month[index] + " days.");
}
```

Example: Searching the array

- Algorithm to search an array for a number

```
int[] nums = {2, 18, 1, 27, 16};
int toFind = 27;

boolean found = false;
for (int i = 0; i < nums.length; i++){
     if ( nums[i] == toFind){
       found = true;
       break;
     }
}
If ( found)
   System.out.println("found!");
else
   System.out.println("not found!");
```


Example: Finding the max

```
int i, max;
int[] nums = {2, 18, 1, 27, 16};
max = nums[0];
for (i = 1; i < nums.length; i++){
     if (max < nums[i]){
       max = nums[i];
}
System.out.println("Max value is " + max);
```


Array out of Bounds!

- **The most common array error is accessing a nonexistent element!**

  double[ ] data = new double[10];   // 10 elements

  data[10] = 5.4;   // the 11th element does not exist

- Subscript/index ranges from 0 to 9.
- No compilation error! But there will be a Runtime Error!
- Programmer to ensure that user does not access array out of bounds!

main(String[] args) method

```
public class Test{
        public static void main (String[ ] args){
                for (int i=0;i<args.length;i++){
                        System.out.println(args[i]);
                }
        }
}
```

String Manipulation

String s = “programming”;

0	1	2	3	4	5	6	7	8	9	10
p	r	o	g	r	a	m	m	i	n	g

- charAt(1) returns ‘r’
- s.length() returns 11

```
for ( int i=0; i< s.length(); i++) {
   System.out.println( s.charAt(i) );
}
```

Displays one letter per line

String Manipulation Example

- Count the number of occurrence of a letter in a String

```
String str = “Java Programming”;
char c = ‘a’;              //look for letter a
int count=0;
for (int i =0; i < str.length(); i++) {
   if ( c == str.charAt(i) )
      count++;
}
System.out.println( count );
```


Multi-Dimensional Array

- Arrays can have more than 1 dimension
- E.g. A 2 dimension array representing a matrix

  int[][] matrix = new int[3][4];
- Assigning a value

  matrix[1][2] = 5;

	0	1	2	3
0				
1			5	
2				

Iterating a 2D Array

```
for (int row=0; row<matrix.length; row++) {
   for ( int col=0; col < matrix[row].length; col++)
        System.out.print( matrix[row][col] );
   System.out.println();
}
```

Methods

- Currently, all statements are put in the main method

```
public static void main(String[] args)
{
    //input statements
    //processing; e.g. sorting, searching etc
    //output statements
}
```


Methods

- Also called functions (in C, C++)
- Modular approach
- Divide and conquer
- Separates logical tasks

Methods

- The syntax for a method is:

```
public static <return type> <method name>(<arguments>)
{
    // operations that this method perform
}
```

- <return type> the data type of the value returned by the method
- <method name> is the name that you give to the method
- <arguments> is the information needed by the method to do its work
- Keywords public and static will be explained in another topic

Types of methods

- A method that does not return any value has a return type void
  - public static void displayHeader()
  - public static void displayString(String s)
- If a method returns a value, the datatype of the value is specified
  - public static String getDateTime()
- Arguments are passed in the brackets after the method name
  - public static boolean isEven(int num)

Example

- Display a rectangle of symbols

```
display(2, 3, "*");
display(4, 5, "#");


public static void display(int row, int col, String sym) {
    for (int i=1;i<=row; i++) {
      for (int j=1;j<=col; j++)
         System.out.print(sym);
      System.out.println();
    }
}
```


Example – isEven()

- A method that returns true if a number is even and false otherwise.

```
public static boolean isEven(int num){
    if ( num %2 == 0)
      return true;
    else
      return false;
}
```


Scope of Variables in Methods

- What is the value of x, y, z, a before and after calling the increment method?

```
public static void main(String[] args){
   int x = 10;
   int y = increment( x);
   System.out.println( x + " " + y);
}
public static int increment(int a) {
   int z = ++a;
   return z;
}
```


Passing a Value

- What happens to x in the main and the method?

```
public static void main(String[] args){
   int x = 10;
   int y = increment( x);
   System.out.println( x + " " + y);
}
public static int increment(int x) {
   int z = ++x;
   return z;
}
```

Passing By Value

```
public static void main(String[] args){
    int x = 10;
    int y = increment( x);
    System.out.println( x + " " + y);
}
public static int increment(int x) {
    int z = ++x;
    return z;
}
```

x 10 y

x 11 z 11

Passing An Array

- What happens to array x in the main and array y in the method?

```
public static void main(String[] args){
    int[] x = {1,2,3,4};
    increment( x);
}
public static void increment(int [] y) {
    for(int i=0; i<y.length; i++)
        y[i] += 1;
}
```

x 1 2 3 4

26

Passing By Reference

- Address of array is passed to the method

```
public static void main(String[] args){
    int[] x = {1,2,3,4};
    increment( x);
}
public static void increment(int [] y) {
    for(int i=0; i<y.length; i++)
        y[i] += 1;
}
```

x 2 3 4 5

y addr of x