

Operation: Fill an array with values

Example: Fill an array with the values 1 through the length

```
int[] myNumbers = new int[50];  
  
for(int i=0; i<myNumbers.length; i++) //it's better to use .length instead of 50... why?  
{  
    myNumbers[i] = i+1;  
}
```

1) Create an array of size 25 to hold integers. Store in the array the values 0 through 24 in **reverse** order.

```
int[] array = new int[25];  
int nextNumber = 24;  
  
for(int i = 0; i < array.length; i++)  
{  
    array[i] = nextNumber;        //or: array[i] = array.length - 1 - i;  
    nextNumber--;  
}
```

2) Create an array of size 10 to hold integers. Store in the array 10 values read in from the user.

```
EasyReader reader = new EasyReader();  
  
int[] array = new int[10];  
  
for(int i = 0; i < array.length; i++)  
{  
    System.out.print("Enter a number");  
    array[i] = reader.readInt();  
}
```

3) Create an array of size 30 to hold integers. Store in the array 30 values chosen randomly between 2 and 10.

```
int[] array = new int[30];  
  
for(int i = 0; i < array.length; i++)  
{  
    array[i] = (int)(Math.random()*9) + 2;  
}
```

Operation: Print out values in the array

Ex: Print out all the numbers in the array

```
int[] myArray = ...; //I'm intentionally not giving you the length of the array - do you know why?  
for(int i=0; i<myArray.length; i++)  
{  
    System.out.print(myArray[i] + " "); //I put a space between each printed number  
}  
System.out.println(); //I'm adding a new line afterwards
```

4) Print out all the numbers in the array that are even.

```
int[] array = ...;  
for(int i = 0; i < array.length; i++)  
    if(array[i] % 2 == 0)  
        System.out.println(array[i]);
```

5) Print out all the numbers in the array that are at an even **index**.

```
int[] array = ...;  
for(int i = 0; i < array.length; i++)  
    if(i % 2 == 0)  
        System.out.println(array[i]);
```

```
int[] array = ...;  
for(int i = 0; i < array.length; i+=2)  
    System.out.println(array[i]);
```

6) Print out all the values in the array that are less than the value after them in the array.

```
int[] array = ...;  
for(int i = 0; i < array.length-1; i++)  
    if(array[i] < array[i+1])  
        System.out.println(array[i]);
```

Operation: Perform a calculation involving all the elements in the array

Ex: Sum (add) up all the numbers in the array

```
int[] theData = ....; //you can't see the length or what's in the array!
int sum = 0;

for(int i=0; i< theData.length; i++)
{
    sum += theData[i];
}
System.out.println(sum); //I'm printing the info - you might want to return this data in a method
```

7) Calculate the sum of all numbers in the array that are odd

```
int[] theData = ....;
int sum = 0;

for(int i=0; i< theData.length; i++)
{
    if(theData[i] % 2 == 1)
        sum += theData[i];
}
```

8) Calculate the **average** of all numbers in the array that are at odd **indexes**.

```
int[] theData = ....;
int sum = 0;
int count = 0;

for(int i=0; i< theData.length; i++)
{
    if(i % 2 == 1)
    {
        sum += theData[i];
        count++;
    }
}

System.out.println(sum / (double)count);
```

9) Calculate which number is the biggest in the array

```
int[] array = ...;
int big = array[0];

for(int i = 1; i < array.length; i++)
    if(array[i] > big)
        big = array[i];
```

10) Calculate at which index the biggest number in the array is at (if multiple, pick the first)

```
int[] array = ...;
int big = array[0];
int bigIndex = 0;

for(int i = 1; i < array.length; i++)
    if(array[i] > big)
    {
        big = array[i];
        bigIndex = i;
    }
```

11) CHALLENGE: Fill an array with the first 50 Fibonacci numbers (0, 1, 1, 2, 3, 5, 8, ...).

```
int[] fib = new int[50];
fib[0] = 0;
fib[1] = 1;

for(int i = 2; i < fib.length; fib++)
    fib[i] = fib[i-1] + fib[i-2];
```