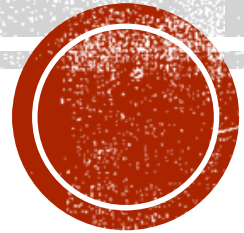
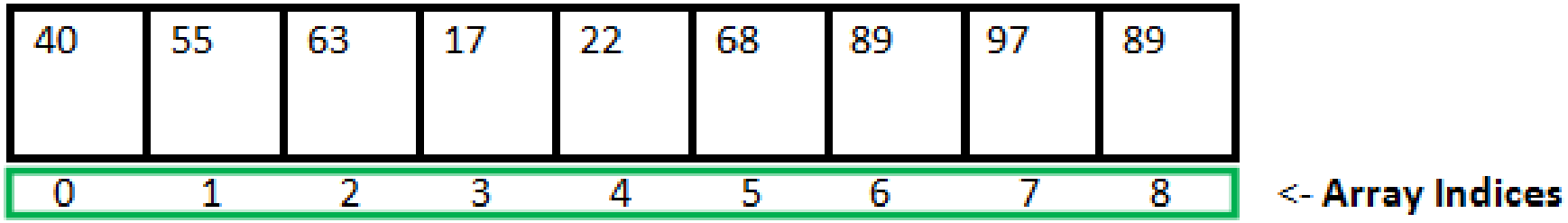
CHAPTER 10: ARRAYS (AND FOR LOOPS)

Fall 2019 – CSC 180 – Introduction to Programming



ARRAYS

image source: google images



Array Length = 9
First Index = 0
Last Index = 8

ARRAYS

- An array is a set of values where each value is identified by an index.
  - I would say that an array is a contiguous block of memory where each "spot" has the same type and can be accessed using an index.

- If you need a block of 5 related integers, one can use something like:
  - `int a, b, c, d, e;`
- But a better solution, would be to use an array of 5 integers:
  - `int[] vals = new int[5];`

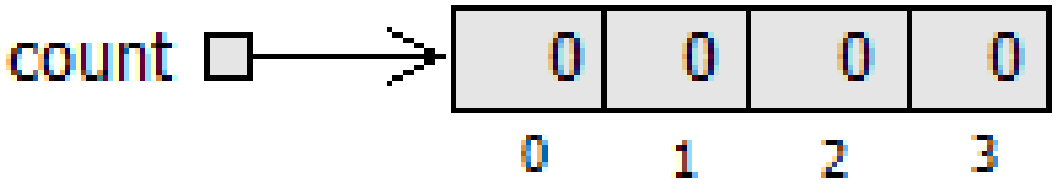
- One immediate advantage of an array (as opposed to having a bunch of independent variables) is that you can access each value from the array using loops (seen next)
  - Imagine if you need 26 related integers (one of letter), or 1500 related strings (one for each student at SMU)
  - How could I store the name of each student in this course? Using variables vs using an array …

ARRAY DECLARATION

- You can declare arrays like this (below, *count* and *values* are just **references,** they will eventually point to a block of memory, but initially they point to nowhere):
  - `int[] count;`
  - `double[] values;`

- Until you initialize these variables, they are set to **null**. This means they point to nowhere, no memory has been reserved for them, the arrays have not yet been created.

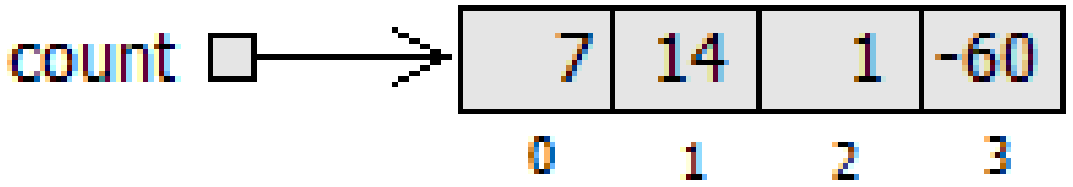
- To create the array itself, use new.
  - `count = new int[4];`
  - `values = new double[size];`
  - `bool[] flags = new bool[20];`
  - `int[] nums = {1, 2, 3, 4, 5}; //creates an initializes an array`



ACCESSING ELEMENTS

- To access (read or write) values in the array, use the [] operator.
  - For example count[0] refers to the first ("zeroeth") element of the array,
  - and count[1] refers to the second ("oneth") element.

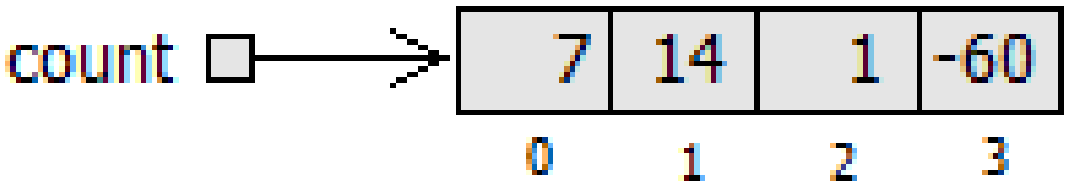
- For example:
  - `count[0] = 7;`
  - `count[1] = count[0] * 2;`
  - `count[2]++;`
  - `count[3] -= 60;`



- One can use a loop to display all values from a given array: using while loops, using for loops, or using foreach loops

ACCESSING ELEMENTS

- For example:
  - `count[0] = 7;`
  - `count[1] = count[0] * 2;`
  - `count[2]++;`
  - `count[3] -= 60;`



- To display all values from an array:
  - Using a **while** loop:

    .....

  - Using a **for** loop:
    -
```
for(int i=0;i<count.Length; i++)
        Console.WriteLine(count[i]);
```
  - Using a **foreach** loop:
    -
```
foreach(int value in count)
        Console.WriteLine(value);
```

while loops and for loops: must know

foreach loops: optional

THE FOR LOOP

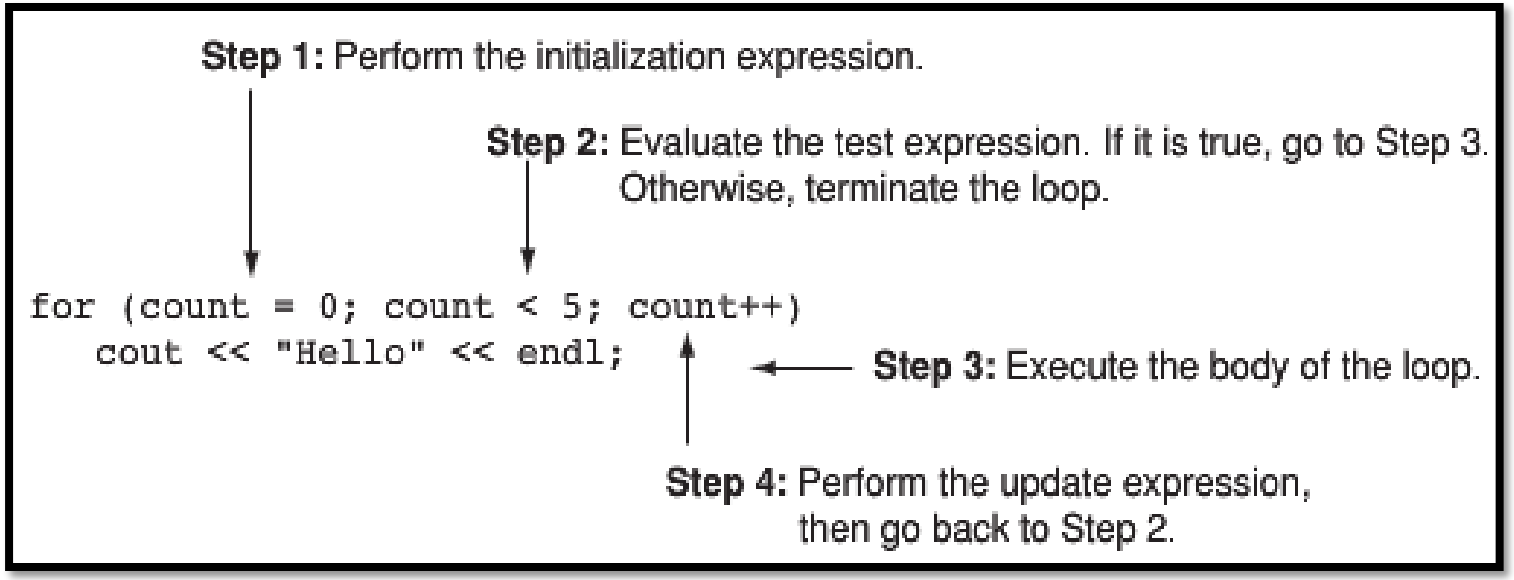
- General format of the while loop:

```
for(initialization; test; update){
        statement(s);
}
```

  - Ideal for performing a known number of iterations.

- Example: display "Hello" five times:
  -

Source: Starting out with C++, by Tony Gaddis, Pearson …



FOR LOOP BY EXAMPLE …

- What is the output of each of the following?

```
for(int x = 1; x <= 10; x++)
{
    Console.WriteLine(x);
}
```

```
for(int x = 1; x < 10; x++)
{
    Console.WriteLine(x);
}
```

```
for(int x = 20; x <= 10; x++)
{
    Console.WriteLine(x);
}
```

```
for(int x = 10; x != 0; x--)
{
    Console.WriteLine(x);
}
```

```
for(int x = 1; x != 10; x+=2)
{
    Console.WriteLine(x);
}
```

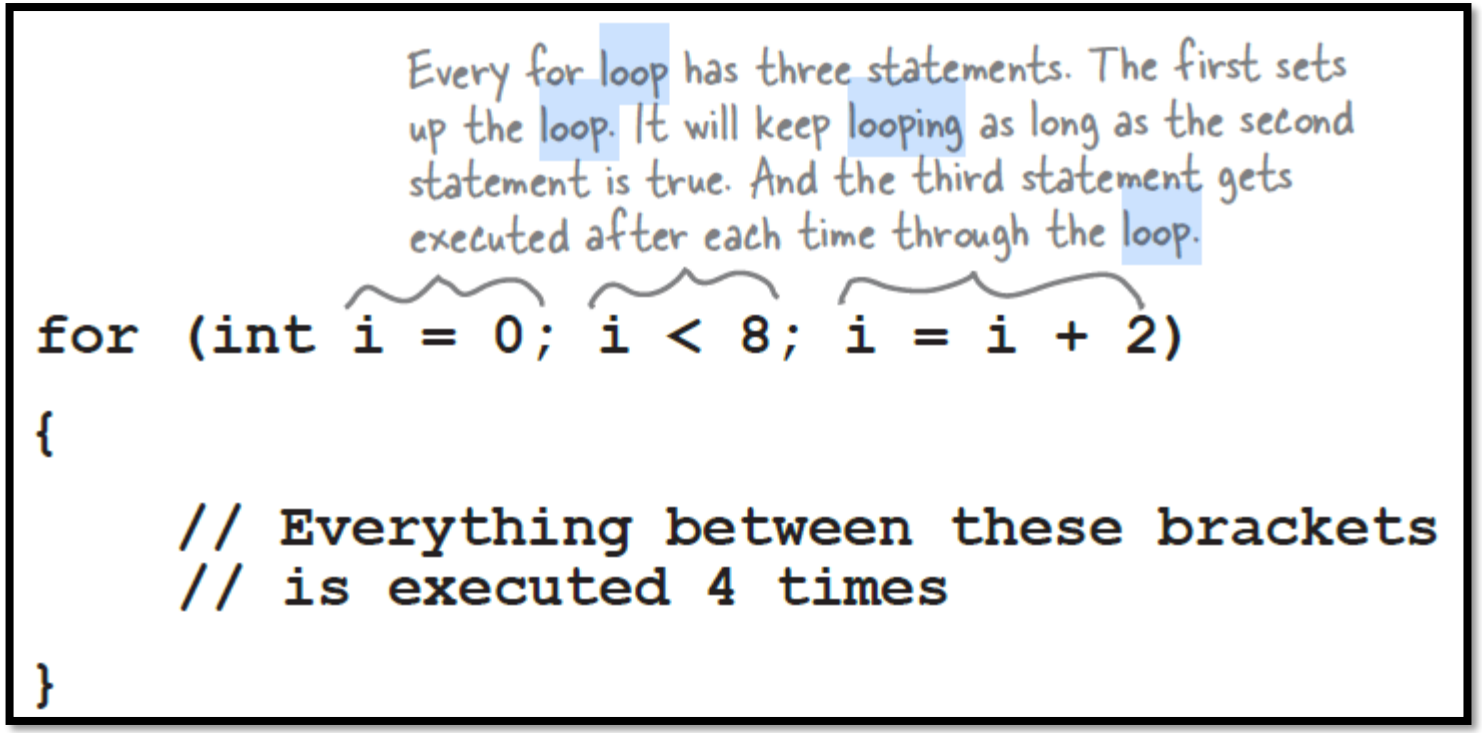


Image source: Head First C# 3rd Edition, by Andrew Stellman, Jennifer Greene (O'Reilly Media, Inc., 2013)

FOR LOOP BY EXAMPLE …

- Ask the user to enter a positive integer **num**. Then, display "Hello World!" **num** many times.

FOR LOOP BY EXAMPLE …

- **TABLES**

- Display a conversion table from F to C, for the following values of F: -10, -5, 0, … , 120.
  - Use the formula: C = 5/9*(F-32)

FOR LOOP BY EXAMPLE …

- Write a program that will display all the divisors of a positive integer n, given by the user. Validate n.

FOR LOOP BY EXAMPLE …

- **RUNNING TOTAL**

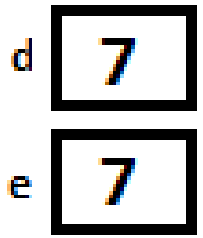
- Write a program that will compute the following sum: 1^4+2^4+…+2019^4.

- Change the program above so it computes the more general problem: 10^4+11^4+…+n^4, where n is a positive integer given by the user. Validate n.

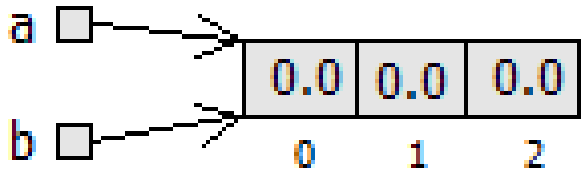
COPYING ARRAYS

- References (such as: arrays and objects) behave differently than value types (such as int, bool, char, double, etc.)
  - **Note:** Strings are an exception – they are references but because they are immutable

- Value type:
  - `int d = 7;`
  - `int e = d;`



- Reference types:
  - `double[] a = new double[3];`
  - `double[] b = a;`



- To create a separate copy of a, one typically has to use a loop
  - `double[] b = new double[a.Length];`

```
for (int i = 0; i < b.Length; i++)          //looping through arrays is very similar to strings!
        b[i] = a[i];
```

EXAMPLES

- Program: write a C# program that would display to the console the contents of an array.

- Program: write a C# program that would ask the user to give you values and put them into an array. You should first ask the user how many values he/she wants to input.

- Program: write a C# program that find the sum of a given array.

EXAMPLES – IF TIME

- Write a C# program that find the largest (max) value in a given array
- Write a C# program that would create an array for 10 random numbers
- Write a C# program that would create a random shuffle of an array containing numbers 1-10.
- Write a C# program that would create a random anagram of a given word (string)
  - Hint: use ToCharArray to create an array of characters
  - Hint: use `new string(charArray)` to recreate the string from the array of characters

FOR LOOP BY EXAMPLE … – IF TIME

- Write a program that will display whether or not a positive integer given by the user is prime.

2D ARRAYS – IF TIME

- Examples involving 2D arrays – if time

HOMEWORK FOR CHAPTER 10

- Requirements: see moodle for details
- Deadline: see moodle

- **Reminder: If your code does not compile, crashes at start, or contains no meaningful comments, it will automatically be graded with 0!**