COMPSCI 121: ARRAYS & REVIEW

SPRING 20

BYTE FROM COMPUTING HISTORY

Sir Tim Berners-Lee invented the World Wide Web in 1989. He also invented the first web browser and protocols and algorithms for the Web.

GOALS FOR TODAY'S CLASS

- **Introduction to your first data structure!**

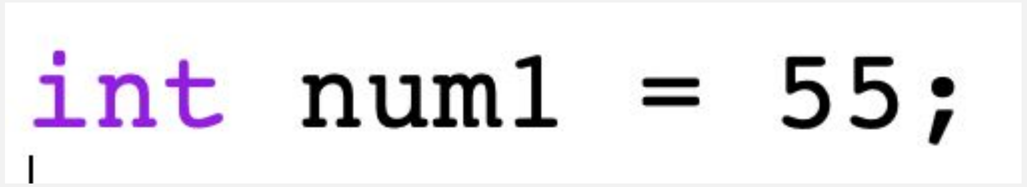
Introduce and explain how Arrays are

- **declared**
- **accessed**
- **modified**
- **added to**

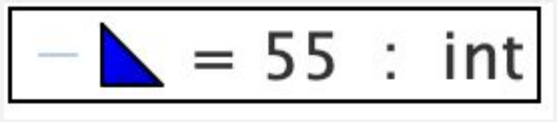
Review of some concepts from zyBooks chapters 4 to 6

STORING DATA

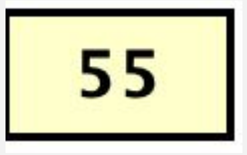
Think of how we stored the value for an `int` variable or a `String`.

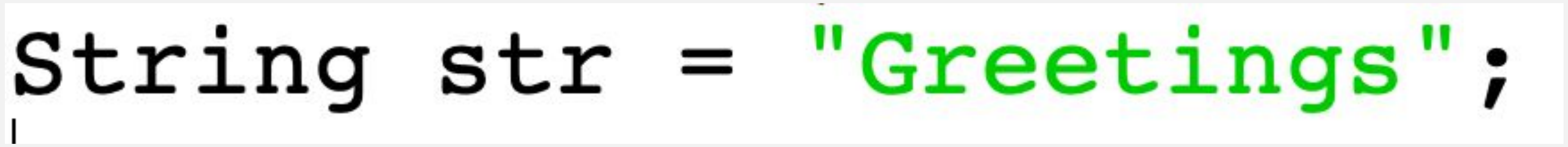


```
int num1 = 55;
```

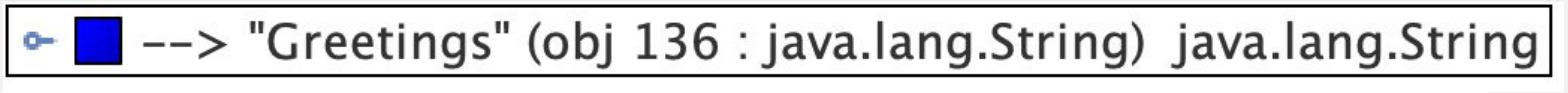


```
= 55 : int
```

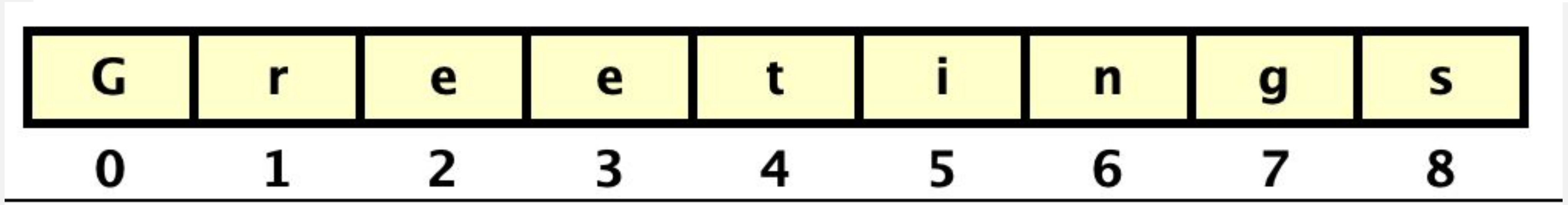




```
String str = "Greetings";
```



```
--> "Greetings" (obj 136 : java.lang.String)  java.lang.String
```



G	r	e	e	t	i	n	g	s
0	1	2	3	4	5	6	7	8

How do we store many `ints` or `Strings`?

SOLUTION: USE AN ARRAY

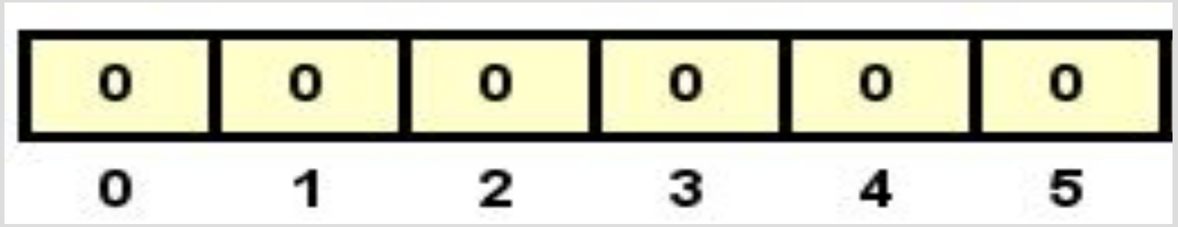
A ***data structure*** that stores and allows access to data.

Operations we can perform on an array:

1. **Declare** (create) and **initialize** it.
2. **Read** data from it.
3. **Modify** existing data.
4. **Traverse** an array.
5. **Add** to an array

1. DECLARE AND INITIALIZE ARRAY

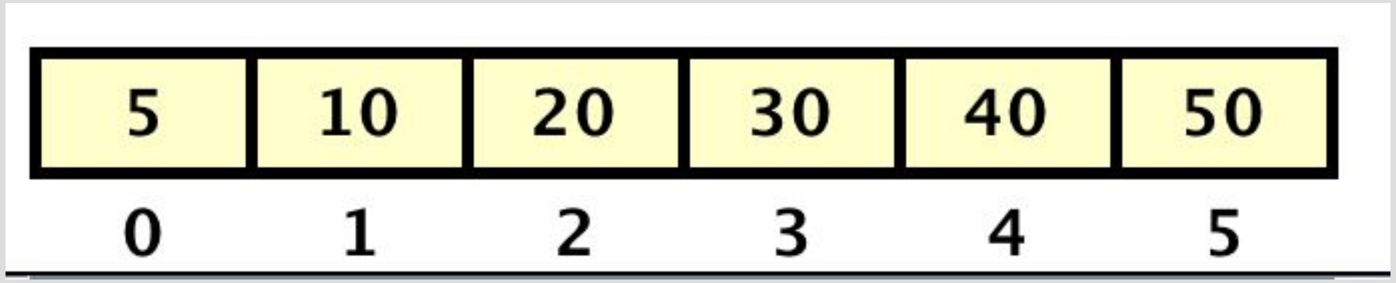
```
int[] firstArray = new int[6];
```



new **operator creates space in memory to store array with the specific type and number of elements**

Or Shorthand

```
int[] firstArray = {5, 10, 20, 30, 40, 50};
```



Note
[ ] brackets
{ } braces

Default values:

```
boolean : false
int : 0
double : 0.0
String : null
```
User defined type: `null`

1.1 ARRAY INDICES

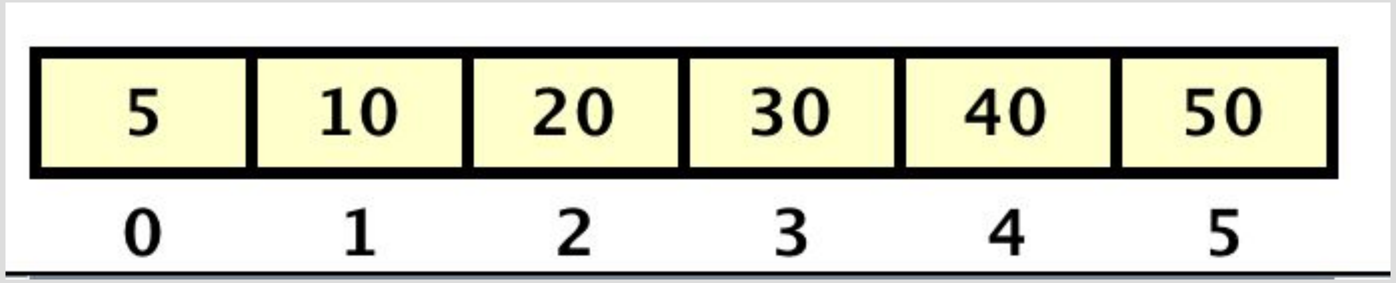
The size of an array is determined when "`new`" is invoked:

```
int[] someArray = new int[66];
int[] nums;
// this is ok - variable is named,
// does not create an array object
// or allocate any space for array
nums[3]  //refers  to index 3
```

Array indices always `int` and always start at 0.

Array indices end at cell # (length - 1): same as `String` indexing!

2. READ DATA FROM AN ARRAY



Assume `firstArray` is declared and holds above values.

```
int curVal = firstArray [4];
```

The value of `curVal` is?

2.1 ARRAY LENGTH

```
int theArray = {3,5,7,9,11};
String myName = "Joe";
```

`myName.length()` vs. `theArray.length`

String: `length()` method

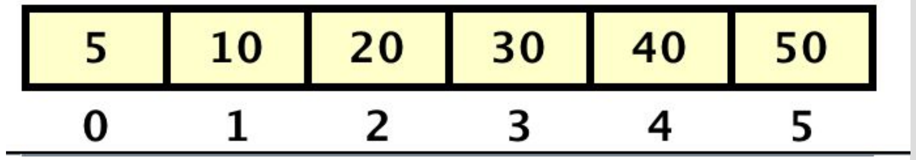
Array: `length` attribute

***`Array.length`* indicates number of array elements.**

2.2 ARRAY EXPRESSIONS

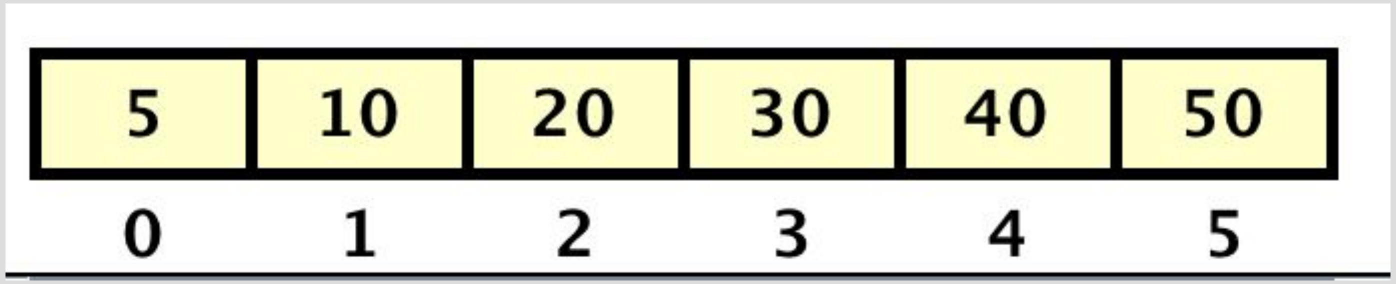
These sorts of expressions are possible:

```
firstArray[4] = 9*firstArray[4];

firstArray[3] = 11;
```

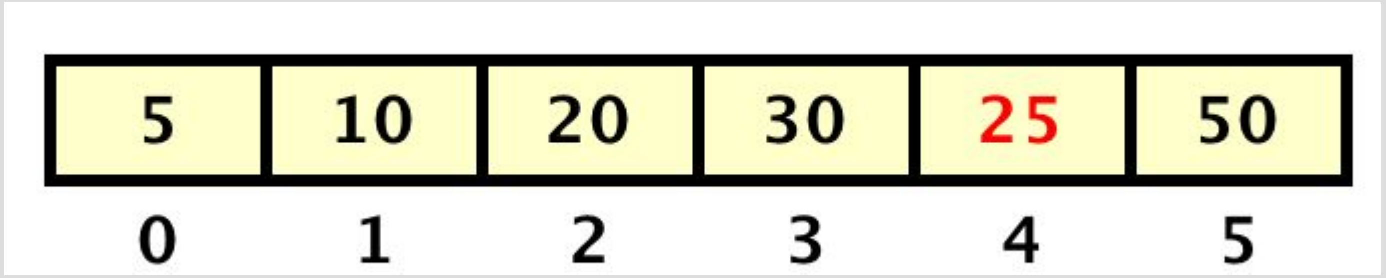


```
firstArray[4] = firstArray[4] +
firstArray[5];

int j = firstArray[3]/2;

firstArray[j] = 9*firstArray[j/2];
```

3. MODIFY EXISTING DATA IN ARRAY



```
firstArray[4] = 25;
```



4. TRAVERSE AN ARRAY

```
int[] firstArray = {1, 2, 3, 4, 5};
```

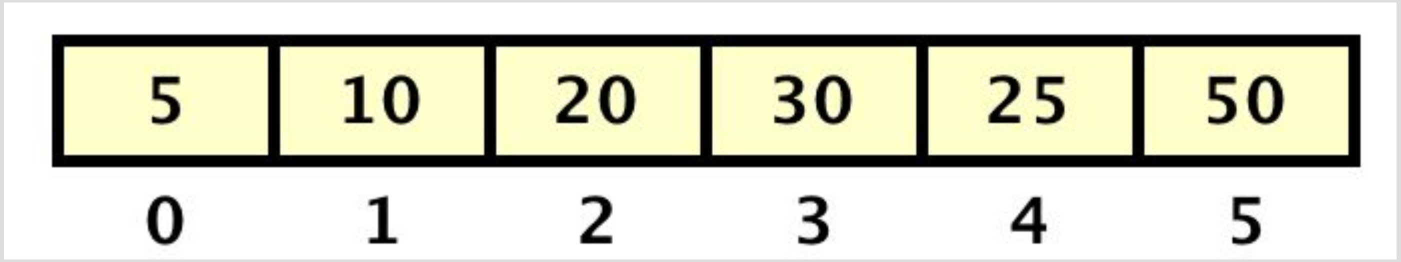
What do the loops print?

```
for(int i = 0;  i < firstArray.length; i++)
   System.out.print(firstArray[i] + " ");
```

```
for(int i = firstArray.length-1;  i >=0; i--)
    System.out.print(firstArray[i] + " ");
```

5. ADD TO AN ARRAY

Let's say we want to add `77` to `firstArray`:

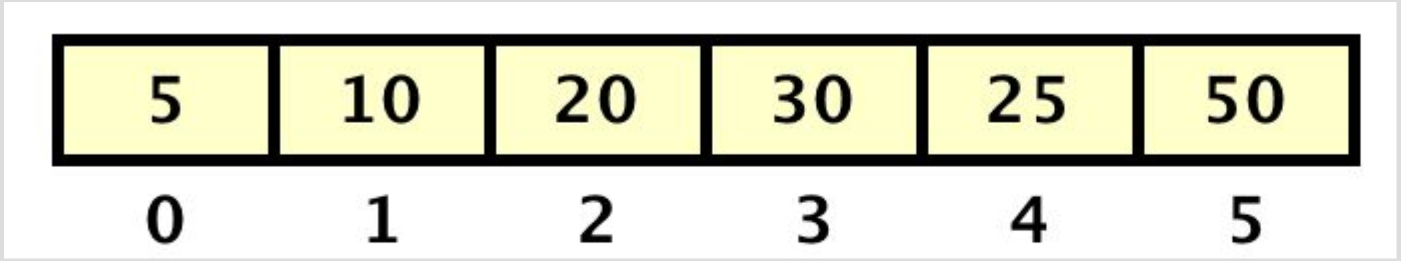


Problem: Array stores only fixed size of elements. It doesn't grow its size at runtime. So,

1. **make a new array with an extra cell.**
2. **copy data from old array to new array.**
3. **add the new data.**
4. **reassign variable.**

5.1. ADD TO AN ARRAY

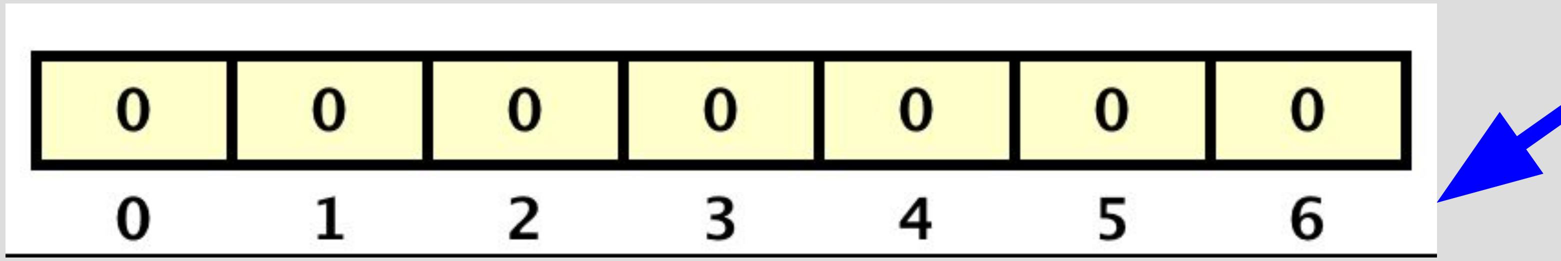
`firstArray:`



Make a new array with an extra cell :

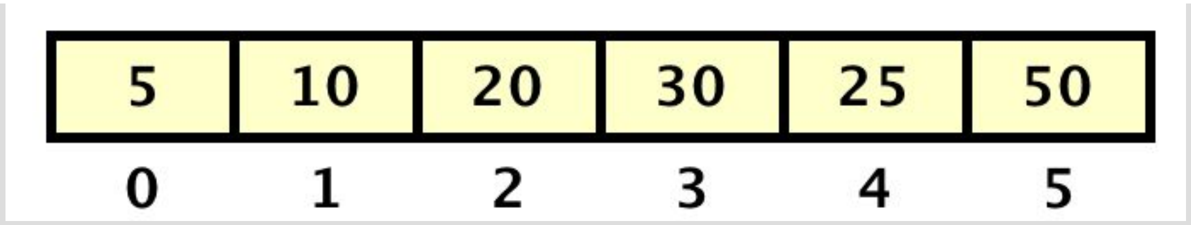
```
int[] tempArray = new int [firstArray.length + 1];
```

`tempArray:`



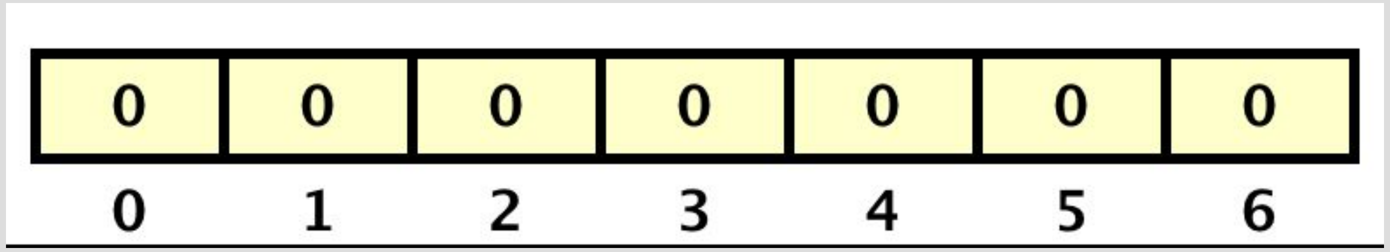
5.2. COPY DATA FROM OLD TO NEW ARRAY

firstArray:



5	10	20	30	25	50
0	1	2	3	4	5

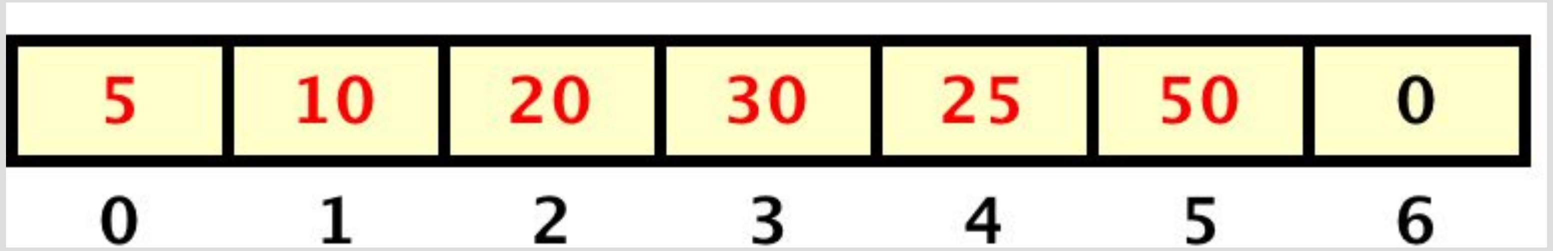
tempArray:



0	0	0	0	0	0	0
0	1	2	3	4	5	6

```
for(int i = 0; i < firstArray.length; i++)
tempArray[i] = firstArray[i];
```

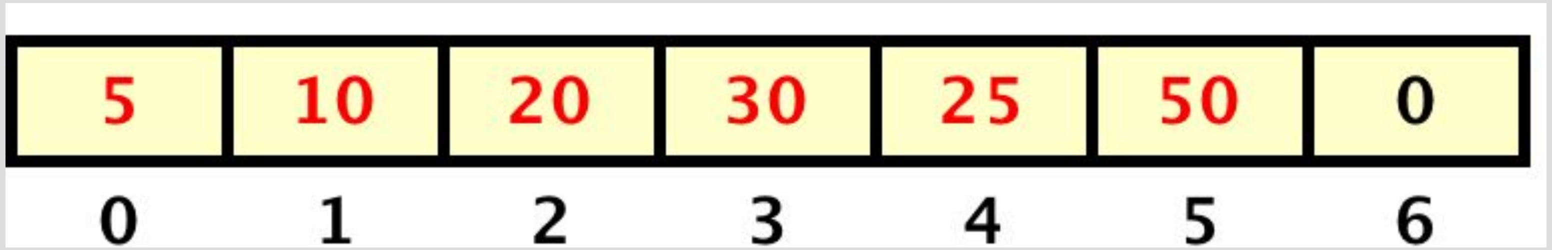
tempArray:



5	10	20	30	25	50	0
0	1	2	3	4	5	6

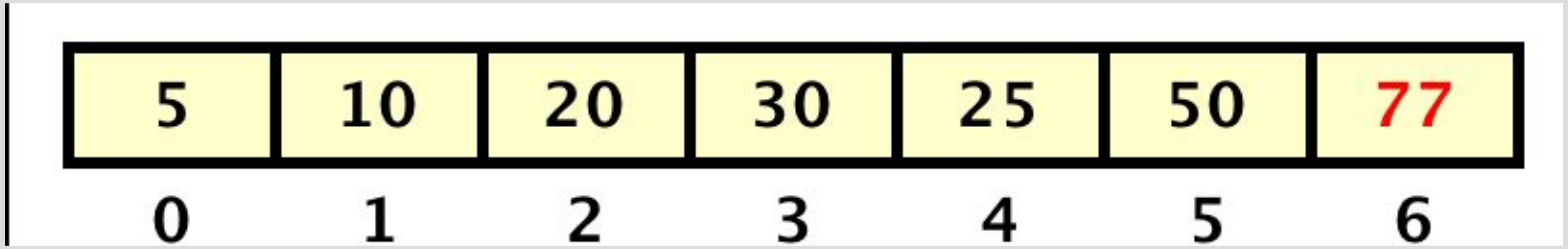
5.3. ADD DATA TO NEW ARRAy

tempArray:



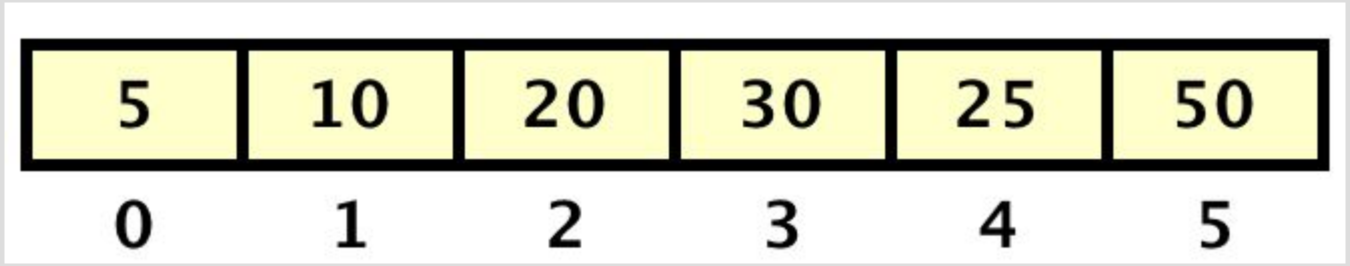
```
tempArray [tempArray.length - 1] = 77;
```

tempArray:

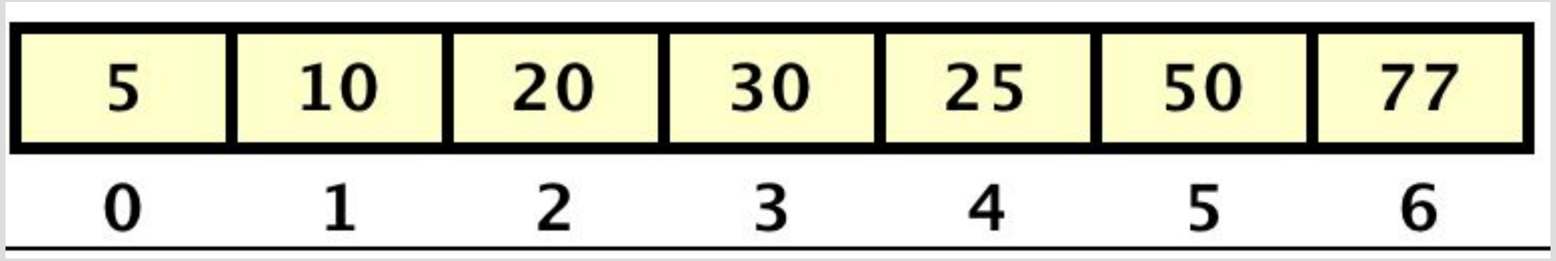


5.4. REASSIGN VARIABLE

firstArray:

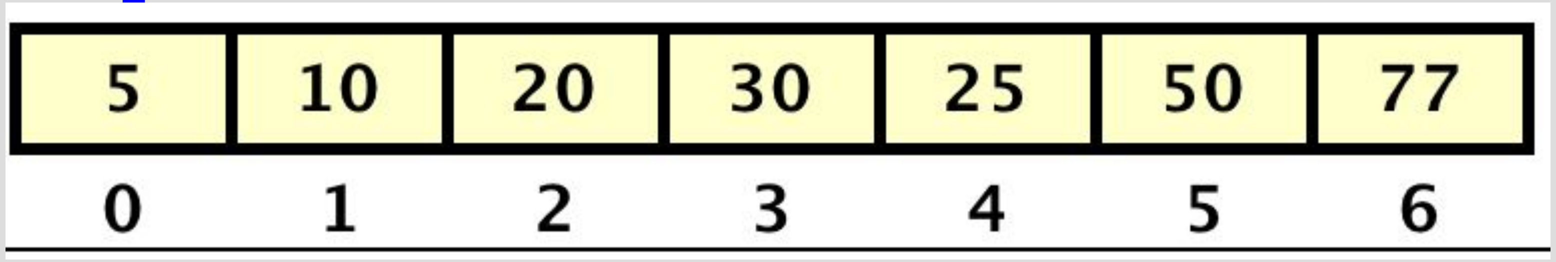


tempArray:



```
firstArray = tempArray;
```

firstArray:



NEXT WEEK

- Arrays of Strings
- Arrays with Objects
- Methods with Arrays

NOW – Review Questions.

REVIEW 1: RANDOM NUMBERS-ANSWERS

Assume a random number `randGen`

What do the following generate?

```
1. randGen.nextInt(6)
2. randGen.nextInt(6) + 10
```

Generate Random integers in the range 10 to 15.

```
3. randGen.nextInt(6) + ______
```

Generate Random integers in the range 16 to 25.

```
4. randGen.nextInt( ______ ) + 16
```

REVIEW 1: RANDOM NUMBERS - ANSWERS

Assume a random number `randGen`

POINTS /4

What do the following generate?

1. `randGen.nextInt(6)` possible value 0 to 5
2. `randGen.nextInt(6) + 10` possible value 10 to 15

Generate Random integers in the range 10 to 15.

3. `randGen.nextInt(6) + 10`

Generate Random integers in the range 16 to 25.

4. `randGen.nextInt(10) + 16`

REVIEW 2: MOD OPERATOR %

Assume a random number `randNum`.

What do the following generate?

1. `randNum % 10`
2. `randNum % 51`
3. `(randNum % 9) + 1`
4. `(randNum % 11) + 20`

REVIEW 2: % MOD OPERATOR

1. `randNum % 10`	Yields 0 - 9
2. `randNum % 51`	Yields 0 - 50
3. `(randNum % 9) + 1`	Yields 1 - 9
4. `(randNum % 11) + 20`	Yields 20 - 30

POINTS /4

% 50 would yield 0 - 49.

% 9 yields 9 possible values 0 - 8, so the + 1 yields 1 - 9.

% 11 yields 11 possible values 0 - 10, so the + 20 yields 20 - 30.

REVIEW 3: LOGICAL OPERATORS

Assume x = 7, y = 9. **Evaluate true /false**

```
1. (x > 0) && (y < 10)

2. (x < 0) && (y < 5  )

3. (x > 0) || (y > 10)

4. (x < 0) || (y > 5)

5. !(x < 0)

6. !(x > 0)
```

REVIEW 3: LOGICAL OPERATORS

Assume x = 7,   y = 9

POINTS /6

```
                                      Answer
1. (x > 0) && (y < 10)
   true         true                  true
2. (x < 0) && (y < 5  )
   false        false                 se
3. (x > 0) || (y > 10)
   true       false                   rue
4. (x < 0) || (y > 5)                 true
    false     true
5. !(x < 0)
     false                              true
6. !(x > 0)
      true                            false
```

Shortcut!

Shortcut!

Shortcut means the second expression is not evaluated!

REVIEW 3: LOGICAL OPERATORS

Given these assignment statements:

```
int a = 1;
int b = 4;
int c = 4;
```

Evaluate the following to true or false

A. !((b == c) || !(a != b))

B. !!((b == c) || !(a != b))

C. !((b == c) && !(a != b))

D. !(!(a <= b) && !(a != b))

REVIEW 3: LOGICAL OPERATORS - ANSWERS

Given these assignment statements:

```
int a = 1; int b = 4; int c = 4;
```

Evaluate the following to true or false **Answer**

A. !((b == c) || !(a != b)) false

NOT ((true) OR NOT (true))

B. !!((b == c) || !(a != b)) true

NOT NOT ((true) OR NOT (true))

POINTS /4

C. !((b == c) && !(a != b)) true

D. !(!(a <= b) && !(a != b)) true

Figure by yourself.
See zyBooks 5.8.1

REVIEW 4: STRING & CHARACTER METHODS

Given `myChar = 'm';`

which methods would you call to

1. make it into a capital letter?
2. check if it is an alphabet?

Given `String str` = `“What a wonderful world”`

which methods would you call to

1. return how many characters the string has?
2. return the word “`world`”?
3. return the third character in the word?
4. add ! to the end of the sentence?

REVIEW 4: STRING & CHARACTER METHODS : ANSWERS

POINTS /6

Given `myChar = 'm';`
which methods would you call to

1. make it into a capital letter? `toUpperCase(c)`
2. check if it is an alphabet? `isLetter(c)`

Given `String str` = `“What a wonderful world”` which methods would you call to

1. return how many characters the string has? `length()`
2. return the word “`world`”? `substring(int, int)`
3. return the third character in the word? `charAt(2)`
4. add ! to the end? `str = str + "!";` or `str += "!";`

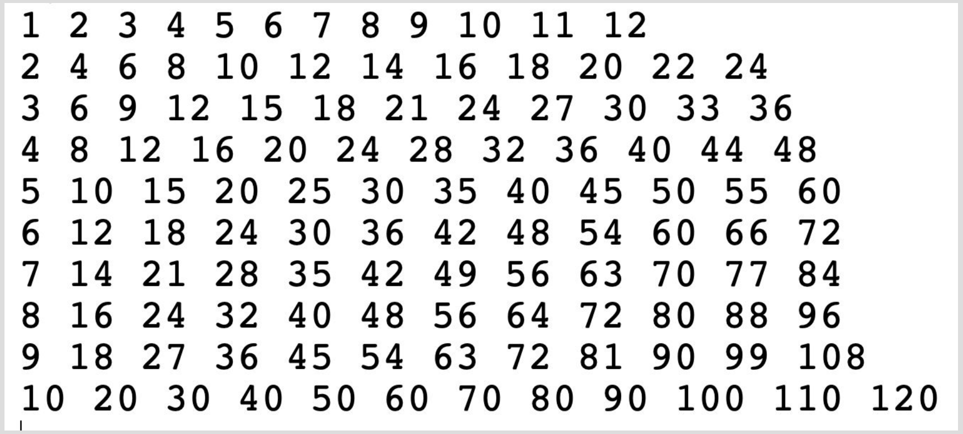
REVIEW 5: NESTED LOOPS

1. What does the following code print?
2. How many times does the outer loop run?
3. How many times does the inner loop run?

```
for (int rowNumber = 1; rowNumber <= 10; rowNumber++) {
   for (int n = 1; n <= 12; n++) {
      System.out.print(n * rowNumber + "  ");
   }
   System.out.println();
 }
```

REVIEW 5: NESTED LOOPS: ANSWERS

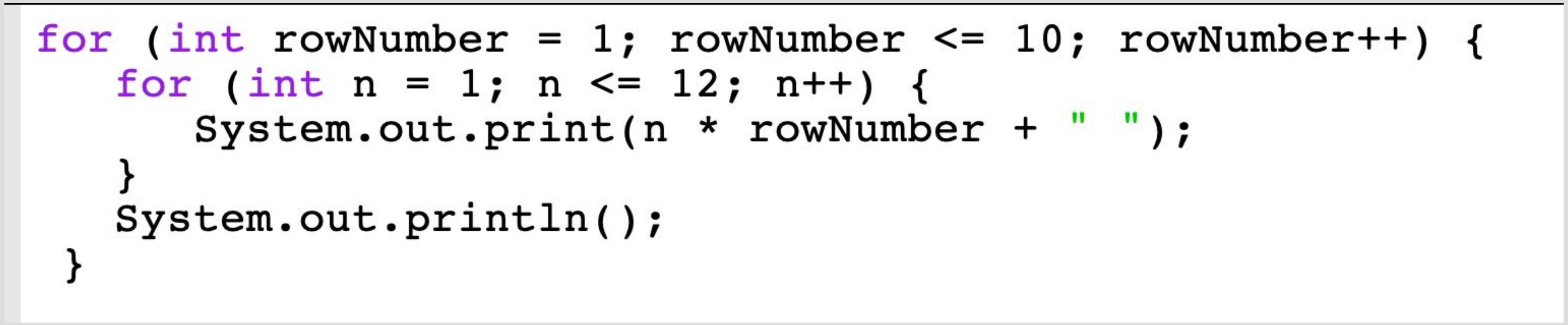
1. What does the following code print? Multiples of 1 to 10 (up to 12 each)
2. How many times does the outer loop run? 10
3. How many times does the inner loop run? 120



```
1 2 3 4 5 6 7 8 9 10 11 12
2 4 6 8 10 12 14 16 18 20 22 24
3 6 9 12 15 18 21 24 27 30 33 36
4 8 12 16 20 24 28 32 36 40 44 48
5 10 15 20 25 30 35 40 45 50 55 60
6 12 18 24 30 36 42 48 54 60 66 72
7 14 21 28 35 42 49 56 63 70 77 84
8 16 24 32 40 48 56 64 72 80 88 96
9 18 27 36 45 54 63 72 81 90 99 108
10 20 30 40 50 60 70 80 90 100 110 120
```

POINTS

/3



```java
for (int rowNumber = 1; rowNumber <= 10; rowNumber++) {
    for (int n = 1; n <= 12; n++) {
        System.out.print(n * rowNumber + " ");
    }
    System.out.println();
 }
```

NEXT WEEK

- Working with arrays of objects.
- Writing methods
  - with arrays as parameters.
  - that return arrays.

TO-DO LIST:

- Check your iClicker grades in Moodle.
- Complete zyBook chapter 7 exercises.
- Communicate with us using only Moodle forum or Piazza.
- Submit Project 3 early and *often* - seek help in office hours.
- Be ready for Exam 2.