

CS 113 – Computer Science I

Lecture 13 – Loops

Tuesday 10/24/2023

Announcements

- HW 05 – Due Monday 10/30 (after fall break)
 - Implement Blackjack!
 - Paired assignment – can work with a partner
- Midterm 1:
 - Mostly graded, should be done by end of week

Agenda

- Announcements
- **While Loops**
- For Loops
- Arrays of Arrays

Exercise

Suppose we wanted to ask the user for 6 numbers (int) and output their sum?

Loops

- Easy way to repeat some computation
- Two kinds of loops:
 - While
 - For
- Loops repeat block of code until the condition becomes false

Example: While Loop

```
int val = 0;
int sum = 0;

int count = 0;
while (count < 6) {
    System.out.print("Enter a number: ");
    val = sc.nextInt();
    sum = sum + val;
    count = count + 1;
}
System.out.println("The sum is "+sum);
```

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum
0			

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum
0	T	0	1

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum
0	T	0	1
1	T	1	3

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum
0	T	0	1
1	T	1	3
2	T	2	5

Tracing Loops

```
int sum = 1;
int count = 0;
while (count < 3) {
    sum = sum + 2;
    count = count + 1;
}
```

Iteration	Count < 6	count	sum
0	T	0	1
1	T	1	3
2	T	2	5
3	T	3	7

Exercise: Tracing loops

```
int sum = 10;  
int count = 0;  
while (count < 6) {  
    sum = sum - 1;  
    count = count + 2;  
}
```

Iteration	Count < 6	count	sum

Exercise: Tracing loops

```
int sum = 10;  
int count = 0;  
while (count < 6) {  
    sum = sum - 1;  
    count = count + 2;  
}
```

Iteration	Count < 6	count	sum
0	T	0	10
1	T	2	9
2	T	4	8
3	T	6	7
4	F		

Accumulator pattern

Idea: Repeatedly update a variable (typically in a loop)

Pattern:

1. Initialize accumulator variable
2. Loop until done
 1. Update the accumulator variable

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

`sum = sum + 2`

`count = count + 1`

`count = count - 1`

`product = product * 2`

`divisor = divisor / 2`

`message = message + "lol!"`

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

sum = sum + 2	
count = count + 1	
count = count - 1	
product = product * 2	
divisor = divisor / 2	
message = message + " lol"	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

sum = sum + 2	sum += 2
count = count + 1	
count = count - 1	
product = product * 2	
divisor = divisor / 2	
message = message + " lol"	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

sum = sum + 2	sum += 2
count = count + 1	count += 1
count = count - 1	
product = product * 2	
divisor = divisor / 2	
message = message + " lol"	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

<code>sum = sum + 2</code>	<code>sum += 2</code>
<code>count = count + 1</code>	<code>count += 1</code>
<code>count = count - 1</code>	<code>count -= 1</code>
<code>product = product * 2</code>	
<code>divisor = divisor / 2</code>	
<code>message = message + " lol"</code>	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

<code>sum = sum + 2</code>	<code>sum += 2</code>
<code>count = count + 1</code>	<code>count += 1</code>
<code>count = count - 1</code>	<code>count -= 1</code>
<code>product = product * 2</code>	<code>product *= 2</code>
<code>divisor = divisor / 2</code>	
<code>message = message + " lol"</code>	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

<code>sum = sum + 2</code>	<code>sum += 2</code>
<code>count = count + 1</code>	<code>count += 1</code>
<code>count = count - 1</code>	<code>count -= 1</code>
<code>product = product * 2</code>	<code>product *= 2</code>
<code>divisor = divisor / 2</code>	<code>divisor /= 2</code>
<code>message = message + " lol"</code>	

Convenience syntax: Assignment

Because updating variable values is so common, language such as Java provide shorthand syntax for it

- Analogy: contractions in English

<code>sum = sum + 2</code>	<code>sum += 2</code>
<code>count = count + 1</code>	<code>count += 1</code>
<code>count = count - 1</code>	<code>count -= 1</code>
<code>product = product * 2</code>	<code>product *= 2</code>
<code>divisor = divisor / 2</code>	<code>divisor /= 2</code>
<code>message = message + " lol"</code>	<code>message += " lol"</code>

Exercise: Write a program that computes powers of 2

Write a program, LoopPow2.java, that computes powers of twos. For example,

```
$ java LoopPow2
Enter an exponent: 0
2 to the power of 0 is 1

$ java LoopPow
Enter an exponent: 1
2 to the power of 1 is 2

$ java LoopPow
Enter an exponent: 4
2 to the power of 4 is 16
```


Agenda

- Announcements
- While Loops
- **For Loops**
- Arrays of Arrays

Example: For Loop



```
int val = 0;
String valStr = "";
int sum = 0;

for (int count = 0; count < 6; count = count +1) {
    System.out.print("Enter a number: ");
    valStr = System.console().readLine();
    val = Integer.parseInt(valStr);
    sum = sum + val;
}
System.out.println("The sum is "+sum);
```

Example: For Loop

initialize condition update

```
for (int count = 0; count < 6; count = count + 1) {  
  
}
```

Exercise: Tracing loops

```
String pattern = "";  
for (int i = 0; i < 3; i++) {  
    pattern = pattern + "*";  
}  
System.out.println(pattern);
```

Iteration	i < 3	i	pattern

Exercise: Tracing loops

```
String pattern = "";  
for (int i = 0; i < 3; i++) {  
    pattern = pattern + "*";  
}  
System.out.println(pattern);
```

Iteration	i < 3	i	pattern
0	T	0	""
1	T	1	"*"
2	T	2	"**"
3	F	3	"***"

Exercise: LoopPattern.java

```
$ java LoopPattern
```

```
Enter a length: 5
```

```
*_*_*
```

```
$ java LoopPattern
```

```
Enter a length: 10
```

```
*_*_*_*_*_*
```

```
$ java LoopPattern
```

```
Enter a length: 0
```

```
$ java LoopPattern
```

```
Enter a length: 1
```

```
*
```

Exercise: Nested loops

```
$ java Square
```

```
Enter a size: 5
```

```
*****
```

```
*****
```

```
*****
```

```
*****
```

```
*****
```

```
$ java Square
```

```
Enter a size: 1
```

```
*
```

```
$ java Square
```

```
Enter a size: 0
```

Iterating through an array

Write a method called `printArray` that takes in an array of integers and prints out the values in each array:

`printArray({1,2,3,4}) -> "1 2 3 4"`

Agenda

- Announcements
- While Loops
- For Loops
- **Arrays of Arrays**

Arrays of Arrays

`int[] array1` is an array of ints

`String[] array2` is an array of Strings

What is `int[][] array3`?

An array of integer arrays

What is `String[][] array4`?

An array of String arrays

2D array example

What does `int[][] array = new int[4][3]` look like?

2D array example

What does `int[][] array = new int[4][3]` look like?

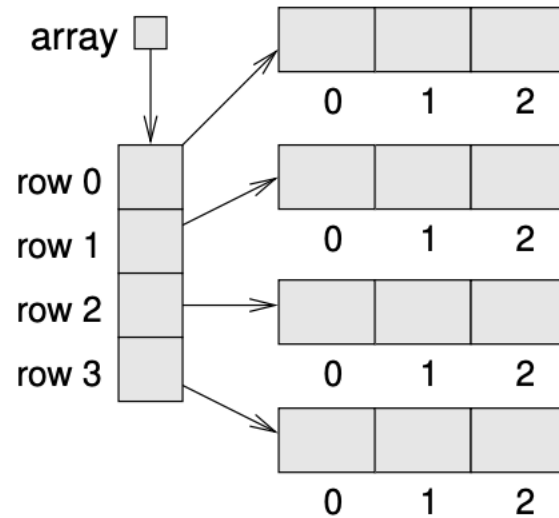


Figure 15.3: Storing rows and columns with a 2D array.

2D Array

Useful for representing a:

- Grid
- Boardgame
- Matrix
- Table
- ...

Traversing through a 2D array

What type of loop should we use?

if we know the length, then a for loop

Pseudocode/algorithm:

for array in 2D array:

for item in array: