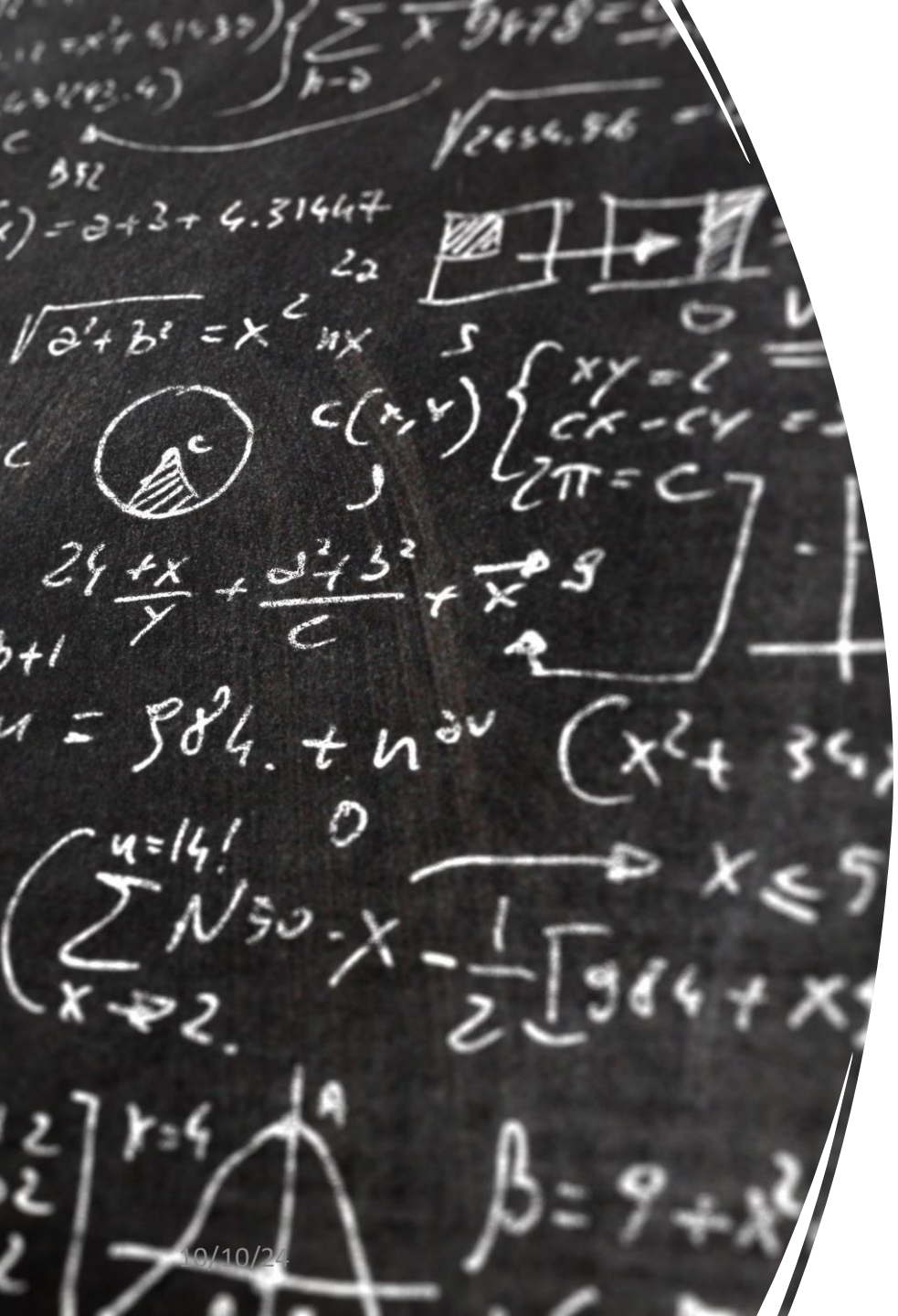

CS 113 – Computer Science I

Lecture 10 – Objects & Loops

Thursday 10/10/2024

Announcements

- HW05
  - Due Tuesday after fall break
  - Autograder will be up today

- Office hours:
  - Adam's Tuesday 2:40-4:00pm Thursday 2:40-4:00pm

HW04 feedback

Ask for help – use Piazza!

Doing well on the homeworks!

If code didn’t compile, I’m manually grading those (haven’t finished)

Midterm – Thursday 10/24

In class, closed book

Terminal commands, vim, directory structure
variables (int, double, char, bool, string, array)
Expressions
Methods
Frame diagrams
Conditionals
Recursion

Practice exam is on course website

Agenda

Strings and Arrays as Objects

Loops

Initializing empty arrays

```
int[] nums = new int[3];
```

[0, 0, 0]

```
String[] strs = new String[3];
```

[null, null, null]

Paired Exercise

Write a recursive method that takes in an array of numbers, an integer *x*, and multiples each number in the array by *x*

```java
public static void add1(int[] list, int pos) {
  if (pos >= list.length) {
    return;
  }
  list[pos] += 1;
  add1(list, pos+1);
}

public static void add1(int[] list) {
  add1(list, 0);
}

public static void main(String[] args) {

  int[] numbs = {10, 20, 30};
  printList(numbs);
  add1(numbs);
  printList(numbs);
}
```

What is numbs after we call add1?

Objects

Strings and arrays are **NOT** primitives

They are objects

Explains why we can't use "==" to compare Strings

"==" checks if two objects are the same

not if the two values are the same

Agenda

Strings and Arrays as Objects

Loops

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

While loop

While a condition is true, run a block of code

```
while(condition) {
  //run the code in this block
}
```

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

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

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
```