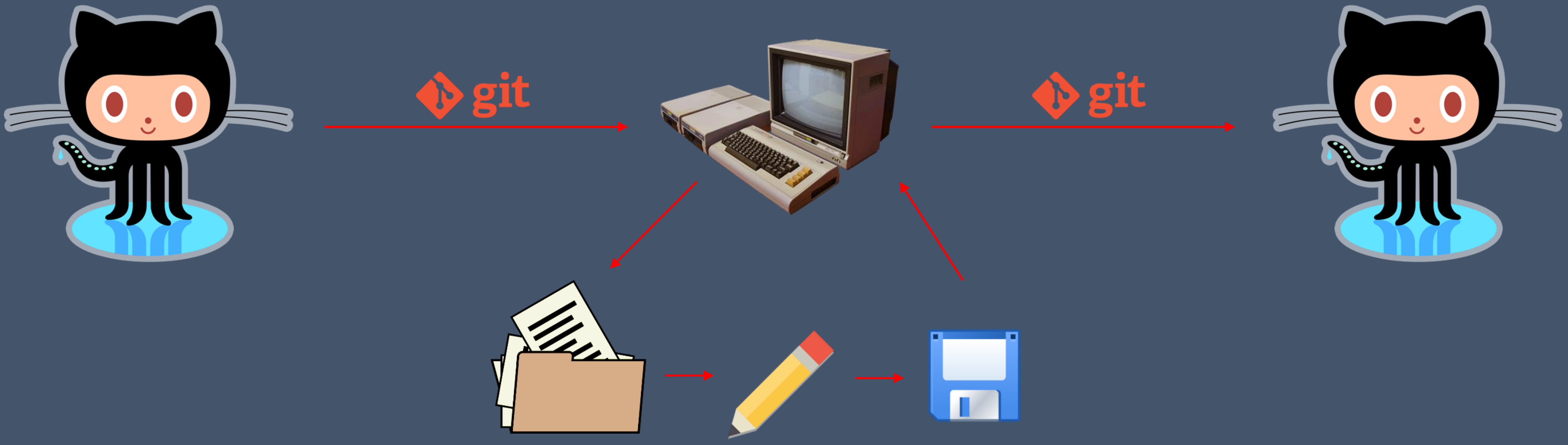
CMPSC 100

Computational Expression

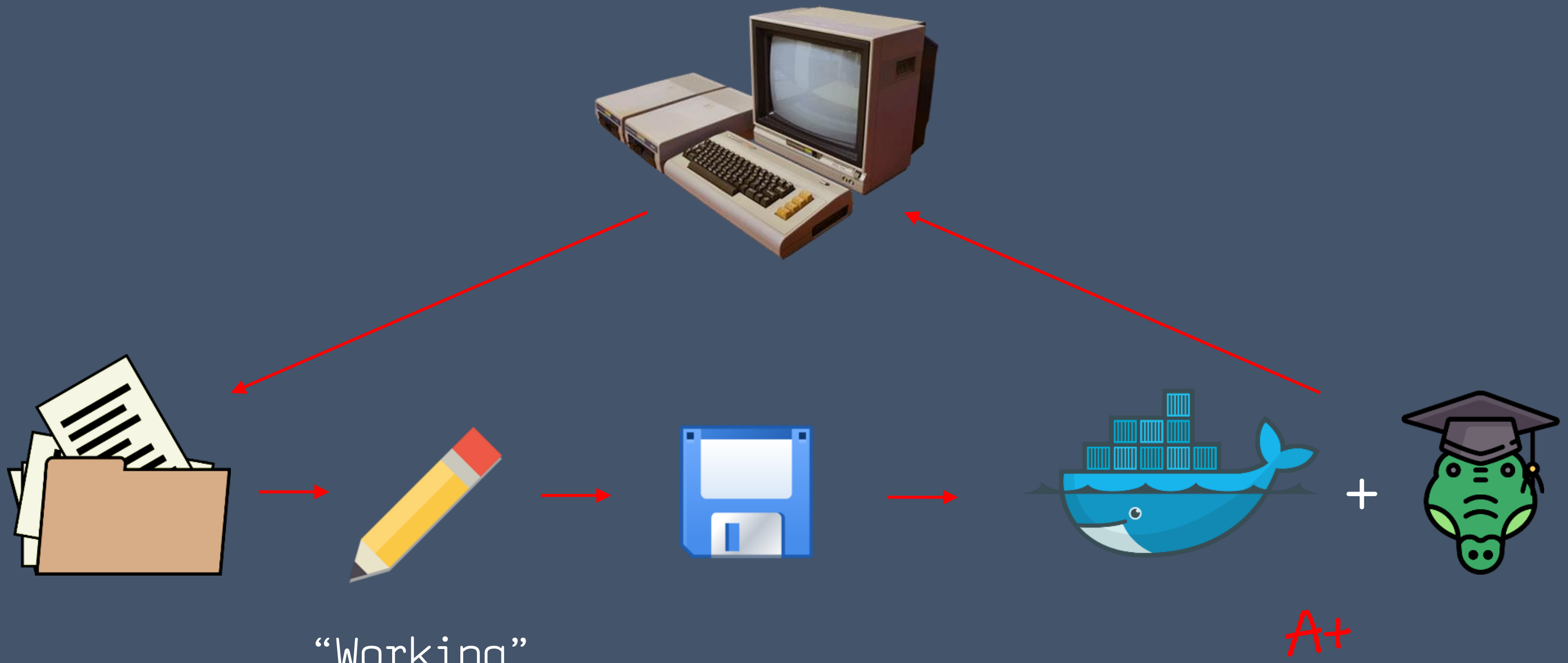


GitHub

Your PC

GitHub

“Working” directory



“Working”
directory

Docker

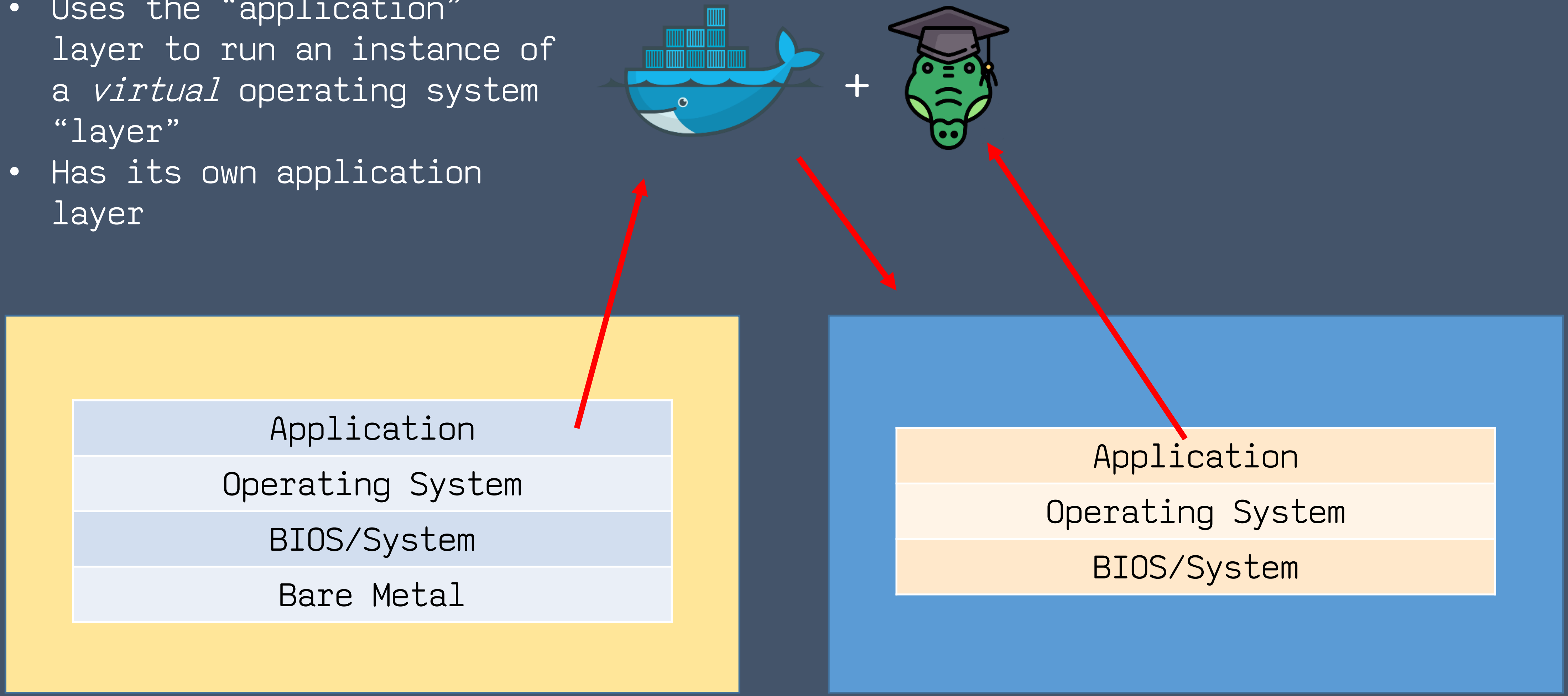
"Containerization" platform

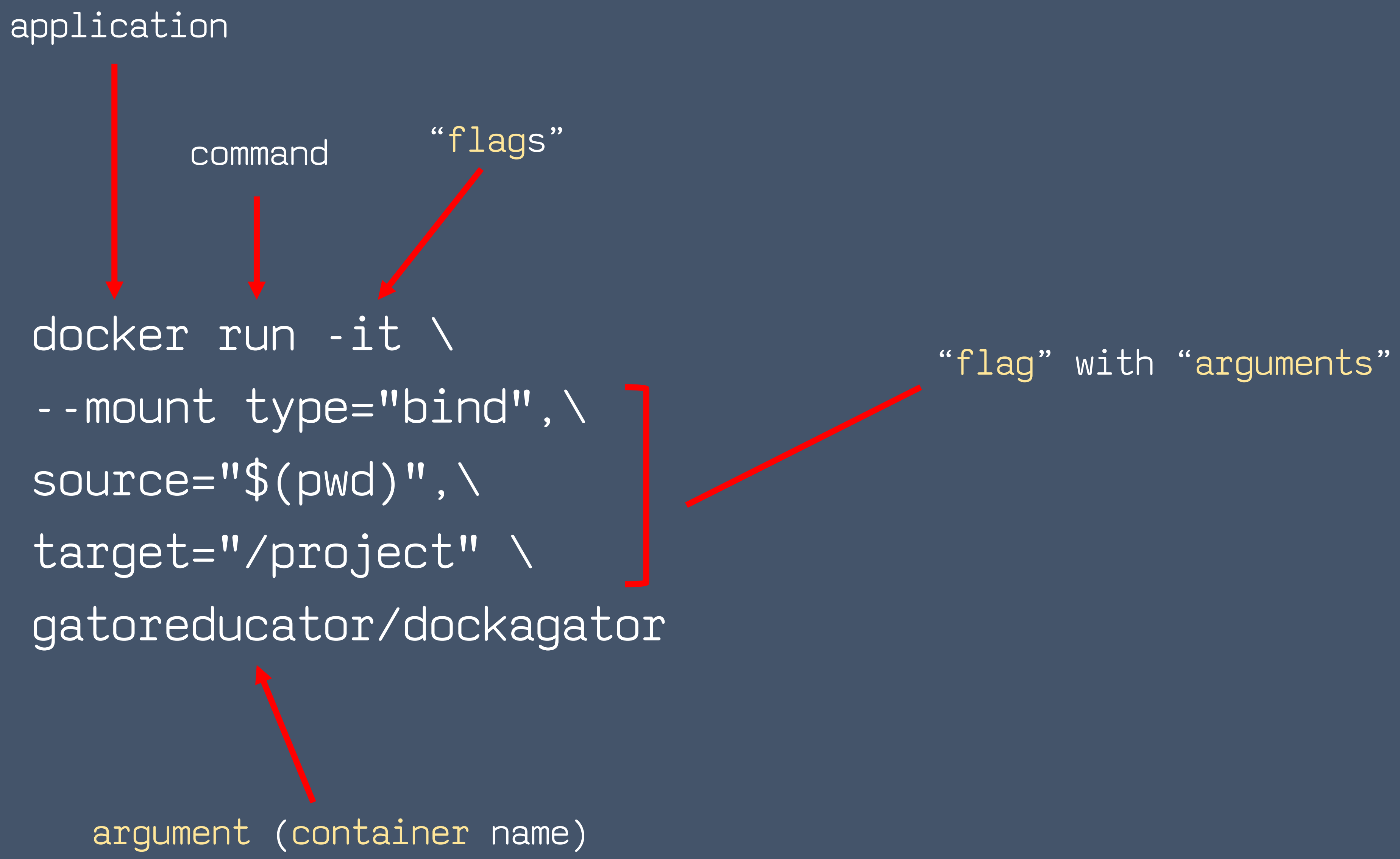
- Uses a system's "working" memory ("main" memory) to:
  - Uses the "application" layer to run an instance of a *virtual* operating system "layer"
  - Has its own application layer

GatorGrader

"Containerized" application

- Packaged into a "container": gatoreducator/dockagator





application

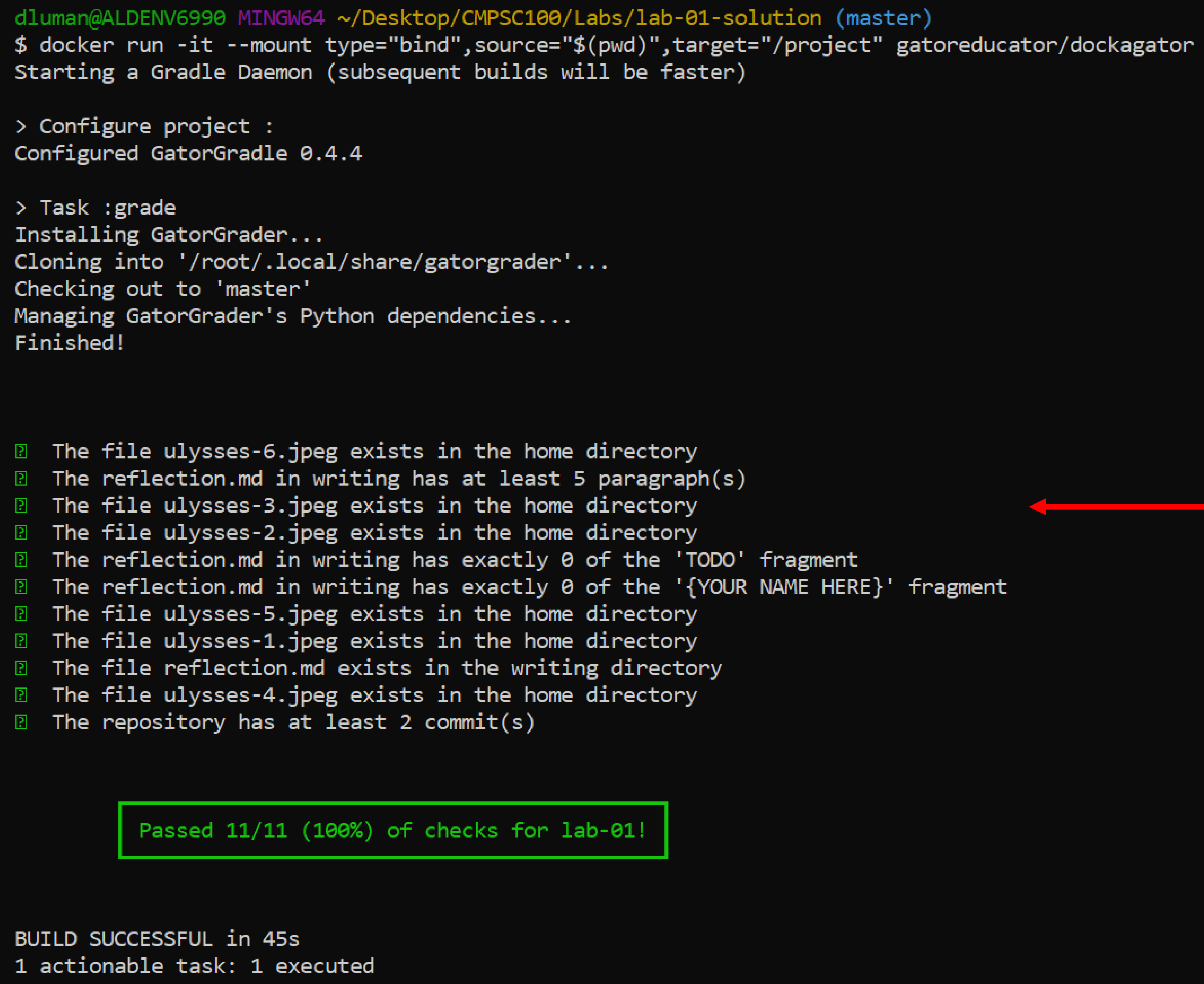
command

“flags”

```
docker run -it \
--mount type="bind",\
source="$(pwd)",\
target="/project" \
gatoreducator/dockagator
```

“flag” with “arguments”

argument (container name)



```
dluman@ALDENV6990 MINGW64 ~/Desktop/CMPSC100/Labs/lab-01-solution (master)
$ docker run -it --mount type="bind",source="$(pwd)",target="/project" gatoreducator/dockagator
Starting a Gradle Daemon (subsequent builds will be faster)

> Configure project :
Configured GatorGradle 0.4.4

> Task :grade
Installing GatorGrader...
Cloning into '/root/.local/share/gatorgrader'...
Checking out to 'master'
Managing GatorGrader's Python dependencies...
Finished!



✔  The file ulysses-6.jpeg exists in the home directory
✔  The reflection.md in writing has at least 5 paragraph(s)
✔  The file ulysses-3.jpeg exists in the home directory
✔  The file ulysses-2.jpeg exists in the home directory
✔  The reflection.md in writing has exactly 0 of the 'TODO' fragment
✔  The reflection.md in writing has exactly 0 of the '{YOUR NAME HERE}' fragment
✔  The file ulysses-5.jpeg exists in the home directory
✔  The file ulysses-1.jpeg exists in the home directory
✔  The file reflection.md exists in the writing directory
✔  The file ulysses-4.jpeg exists in the home directory
✔  The repository has at least 2 commit(s)


        Passed 11/11 (100%) of checks for lab-01!


BUILD SUCCESSFUL in 45s
1 actionable task: 1 executed
```

Example of a successful GatorGrader report for lab-01

3 steps for creating and pushing a commit

1

```
git add .
```

2

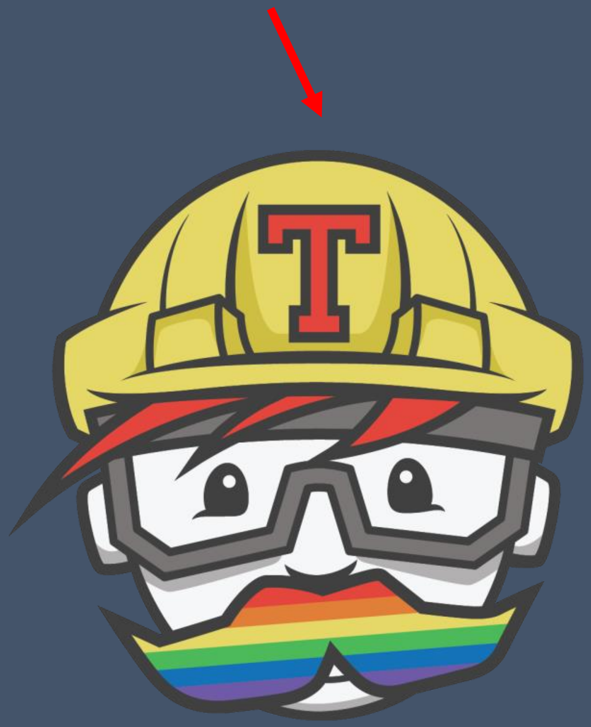
```
git commit -m “{COMMIT MESSAGE}”
```

```
git commit -m “Submitting assignment”
```

3

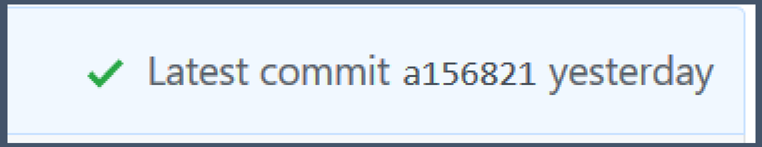
```
git push
```

This is Travis. They are your friend, too. But, unlike G. Wiz, they'll never try to trick you.



Travis is a “Continuous Integration/Continuous Deployment” (CI/CD) service. For us this means:

- Runs after every `git push`
- Reports the overall success of failure of build (incl. GatorGrader)

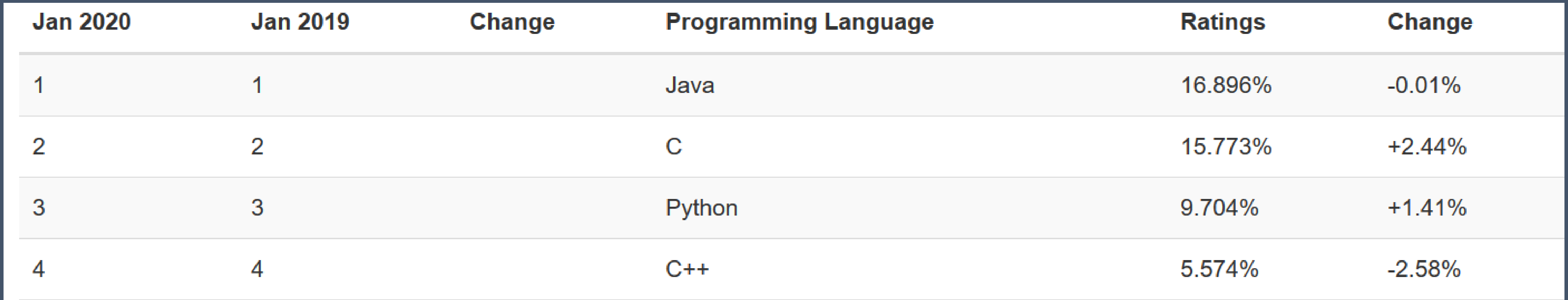


✓ Latest commit a156821 yesterday

- Success ✓
- Error ✗
- In progress ●

Java

- Created in 1996 at Sun Microsystems
- Developed to speed up software development
- As of this month (Jan. 2020), still the top-used commercial software language

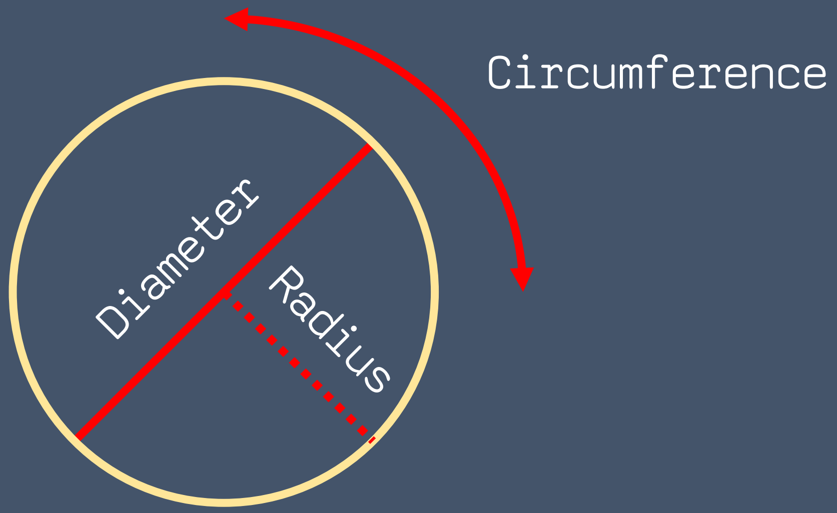


Jan 2020	Jan 2019	Change	Programming Language	Ratings	Change
1	1		Java	16.896%	-0.01%
2	2		C	15.773%	+2.44%
3	3		Python	9.704%	+1.41%
4	4		C++	5.574%	-2.58%

Java

- One of several "object-oriented programming" languages
  - Everything in Java is a object that has various properties or attributes

Circumference	$2\pi r$ (πd)
Diameter	$2r$
Area	πr^2

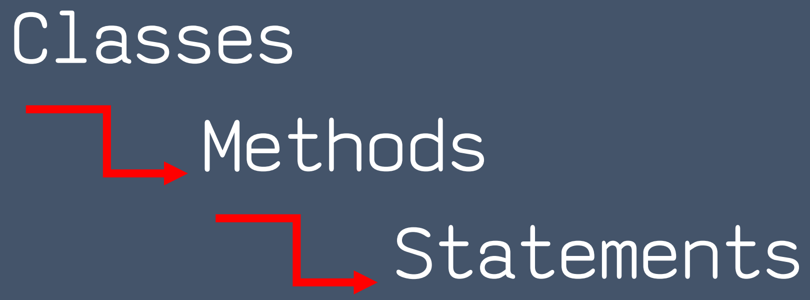


Java

- Programs are composed of:
  - Definitions
    - “Classes”
    - “Methods”
  - Statements
    - “Method” calls
    - Assignments
- Each definition or statement has appropriate syntax
  - Syntax is collected in *.java files (“inert”)
    - “Blueprints”
  - When compiled and loaded into “working” memory (when executed), programs become functional

Java

- Every program requires a “driver” class
- This contains a “main” method
  - This class/method is place where the code begins



Classes

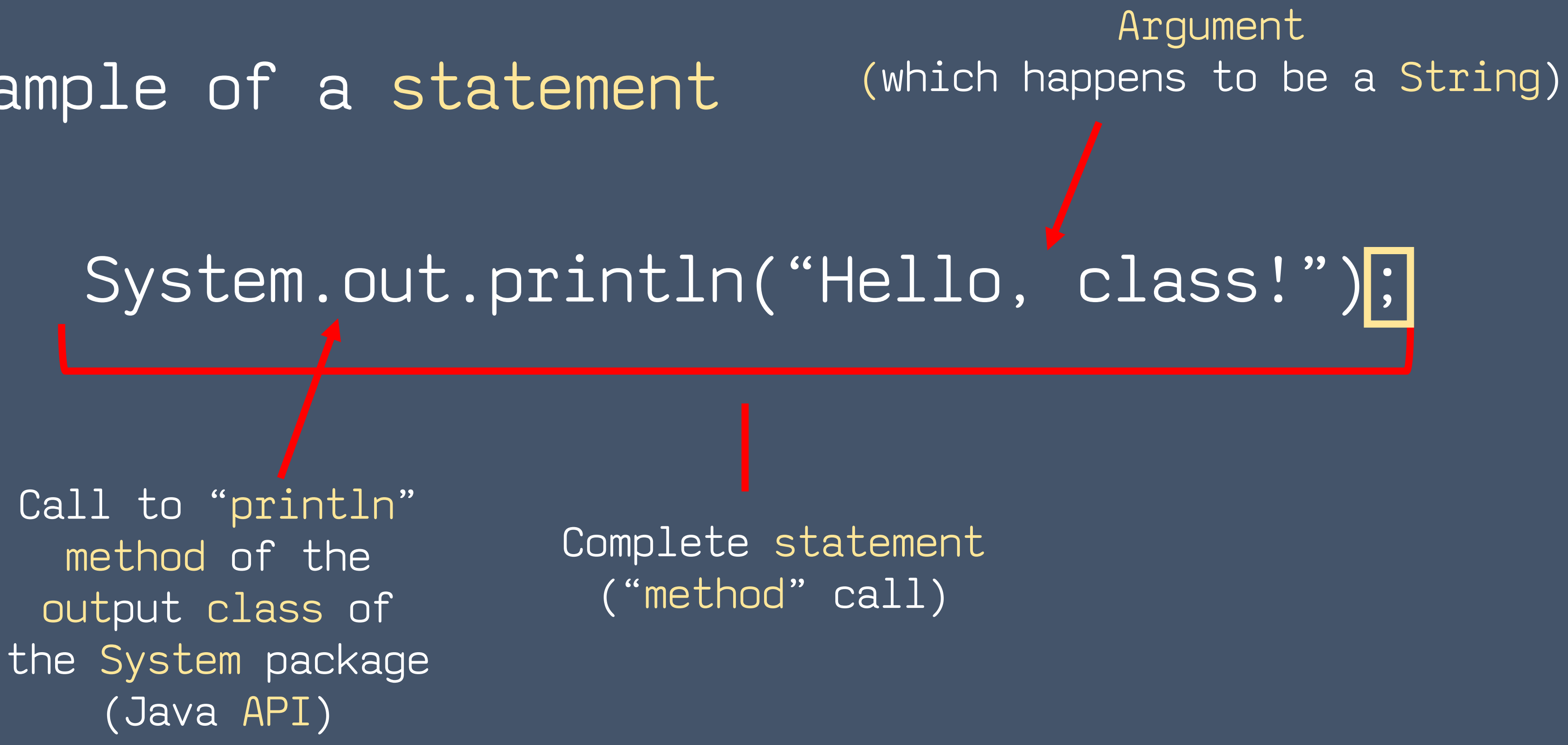
Methods

Statements

Java

- Right now, we're going to focus on statements

Example of a statement



Argument
(which happens to be a String)

```
System.out.println("Hello, class!");
```

Call to "println" method of the output class of the System package (Java API)

Complete statement ("method" call)

Java

Example of a statement

A single-line comment

```
// The below statement prints “Hello, World!”
System.out.println(“Hello, class!”);
```

Comments are:

- Useful for documentation
- Non-functional (they do not execute)

Java

Application Programmer Interface

Interruption #1: the Java API

Things like System.out.println come “for free” with Java compilers because the language simply wouldn’t be useful without it.

We can “extend” the functionality of our programs by bringing in other parts of the language, but we won’t do that…yet.

Java

Interruption #2: The many ways to do things in Java

Example: Java includes multiple ways to print lines.

```
System.out.print(“Hello, ”);
System.out.print(“class!”);
```

1

```
      > Hello, class!
```

```
System.out.println(“Hello, ”);
System.out.println(“class!”);
```

2

```
      > Hello,
      > class!
```

Activity

1. cd to your CMPSC100/Activities folder

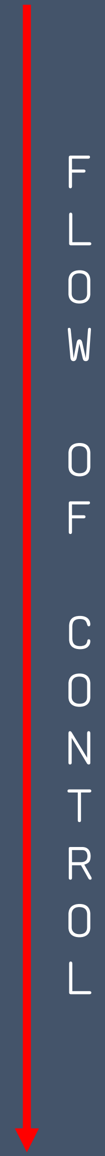
2. Visit the #activities channel in course Slack

3. Click the link and clone the repository there

4. cd to the activity-01 folder and follow along

```java
public class Poem {

  /** Entry point.
   *
   * @param args The command line arguments
   */
  public static void main(String[] args) {
    System.out.println("Harlem");
    System.out.println("Langston Hughes");
    System.out.println();
    System.out.println("What happens to a dream deferred?");
    // Remainder of code removed to fit this on a slide
  }
}
```



FLOW OF CONTROL