Lesson 2.5: Programming Implicitly

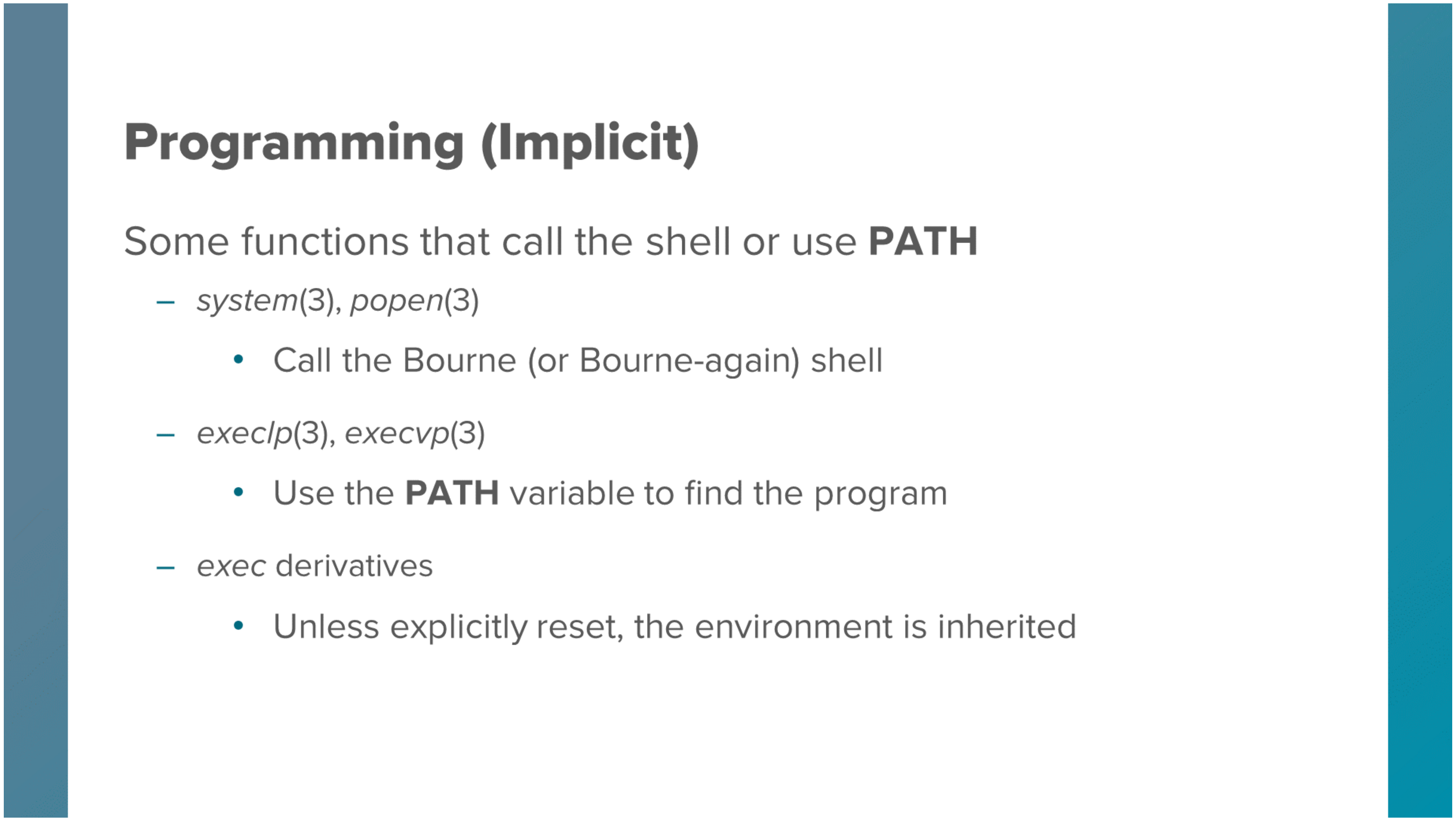
SECURITY VULNERABILITIES IN C/C++ PROGRAMMING

Programming Implicitly

Matt Bishop, Ph.D.
Professor of Computer Science,
UC Davis

UCDAVIS
Continuing and Professional Education

Slide 1: Programming (Implicit)

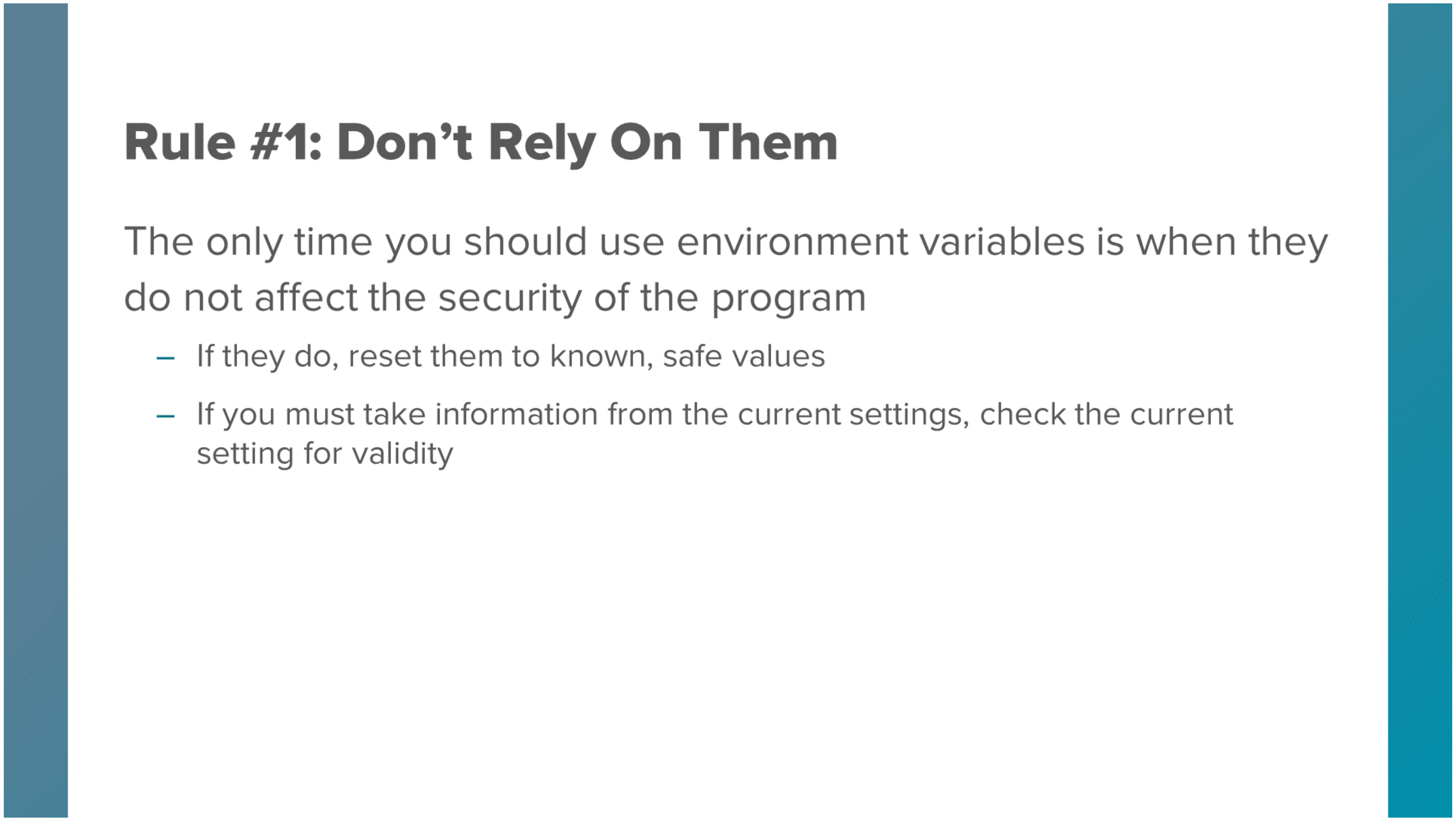


Programming (Implicit)

Some functions that call the shell or use **PATH**

- *system*(3), *popen*(3)
  - Call the Bourne (or Bourne-again) shell
- *execlp*(3), *execvp*(3)
  - Use the **PATH** variable to find the program
- *exec* derivatives
  - Unless explicitly reset, the environment is inherited

Slide 2: Rule #1: Don’t Rely On Them

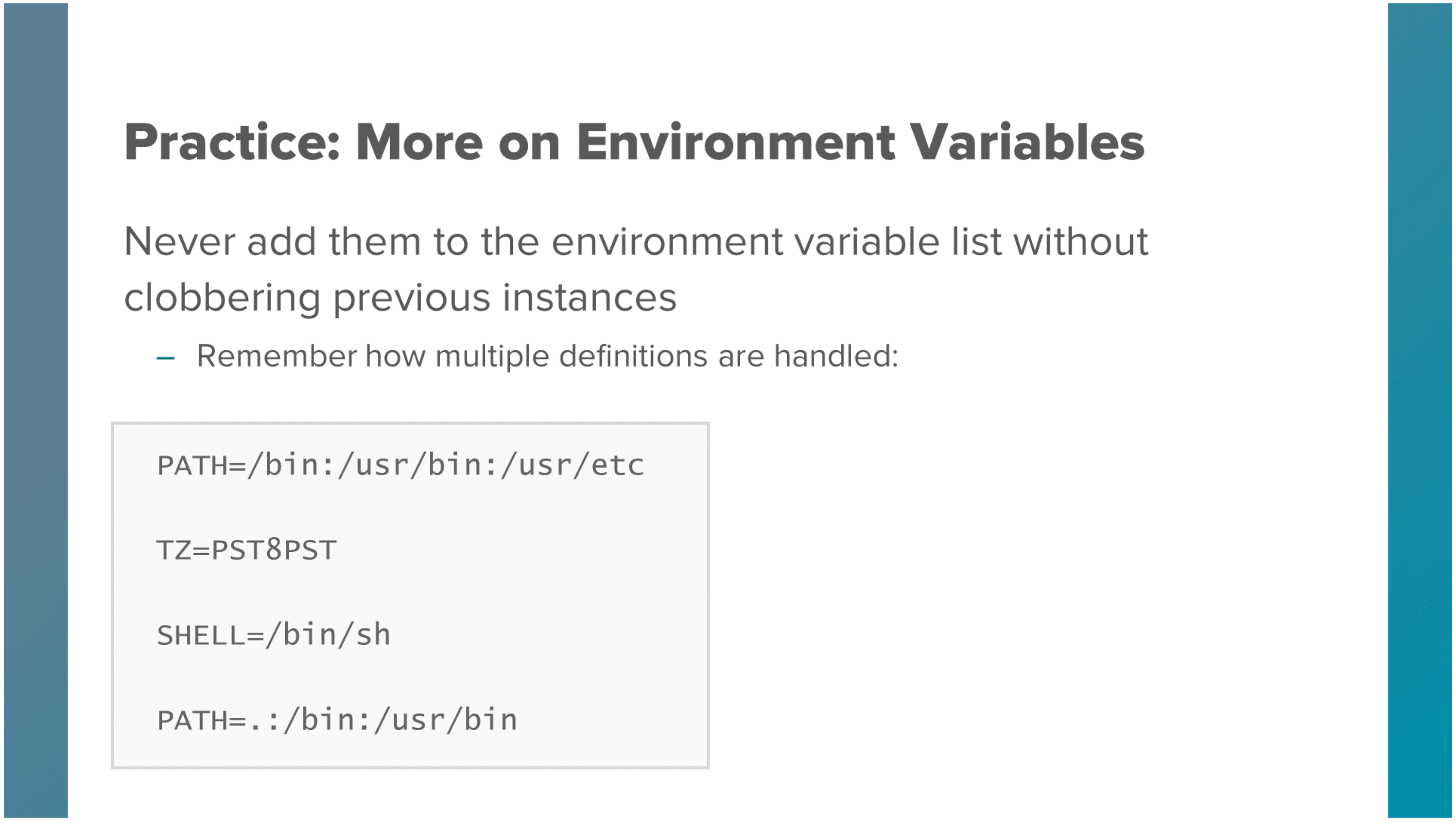


Rule #1: Don’t Rely On Them

The only time you should use environment variables is when they do not affect the security of the program

- If they do, reset them to known, safe values
- If you must take information from the current settings, check the current setting for validity

Slide 3: Practice: More on Environment Variables



Practice: More on Environment Variables

Never add them to the environment variable list without clobbering previous instances

- Remember how multiple definitions are handled:

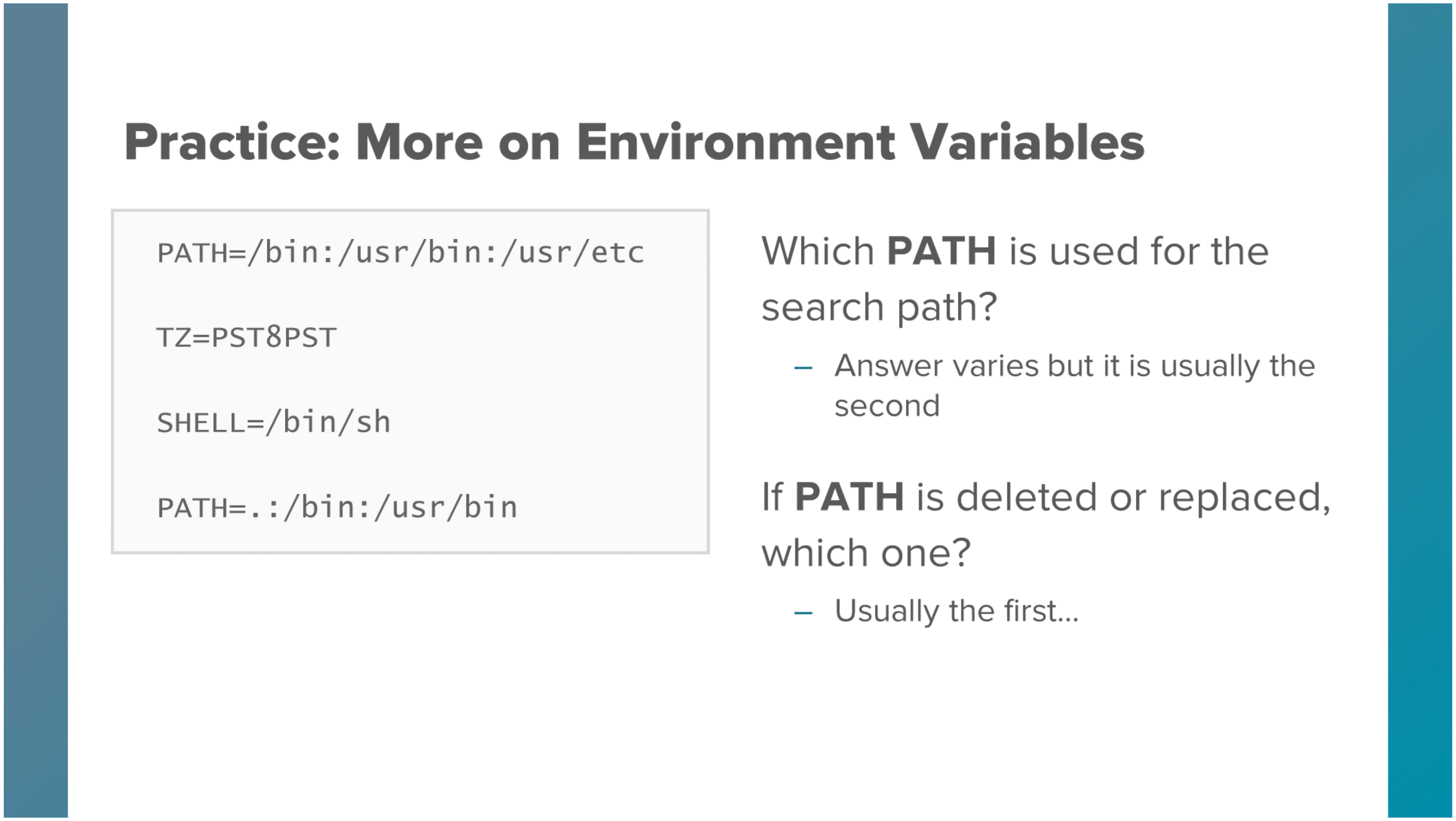
```
PATH=/bin:/usr/bin:/usr/etc

TZ=PST8PST

SHELL=/bin/sh

PATH=.:/bin:/usr/bin
```

Slide 4: Practice: More on Environment Variables



Practice: More on Environment Variables

```
PATH=/bin:/usr/bin:/usr/etc

TZ=PST8PST

SHELL=/bin/sh

PATH=.:/bin:/usr/bin
```

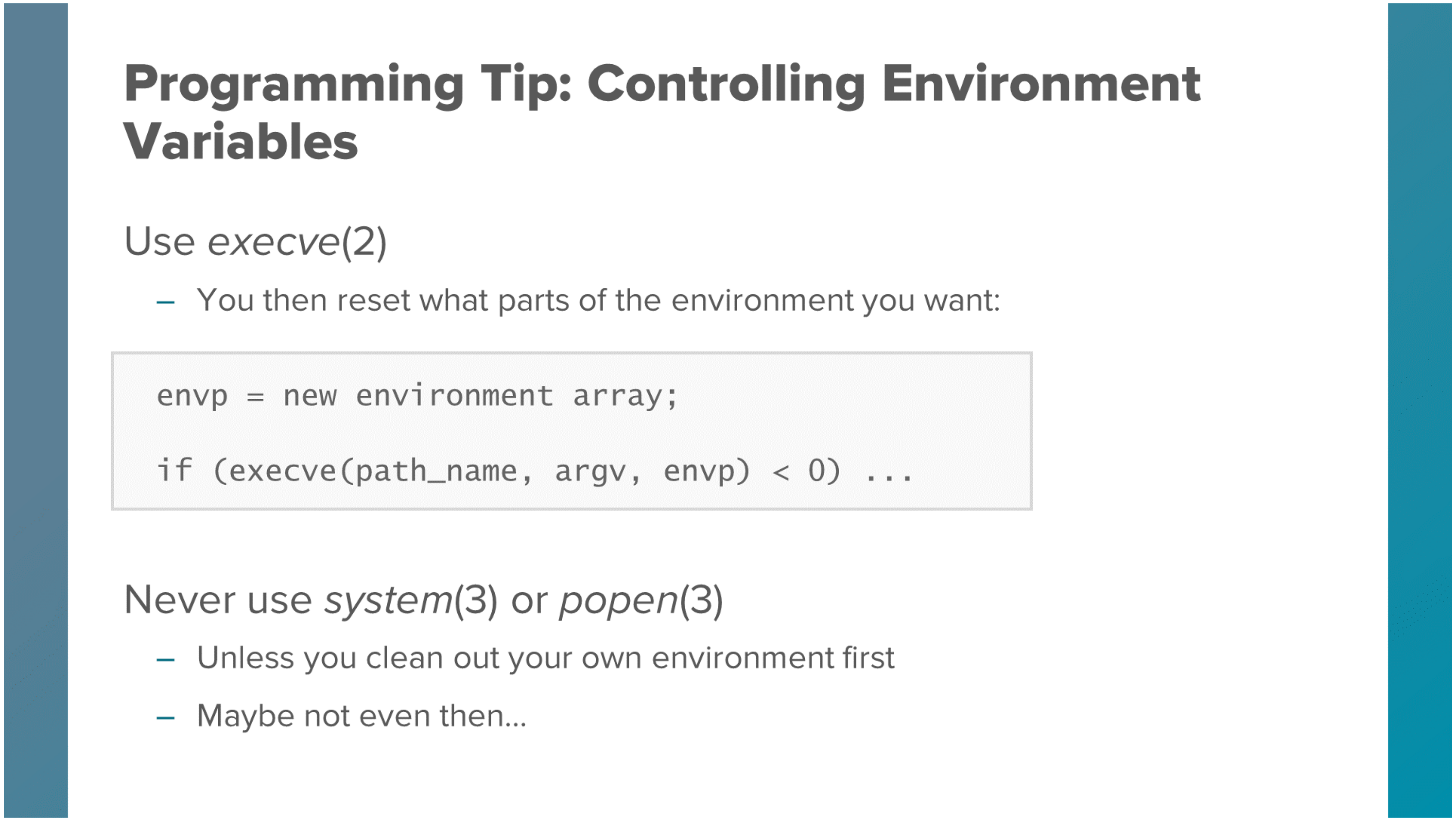
Which **PATH** is used for the search path?

- Answer varies but it is usually the second

If **PATH** is deleted or replaced, which one?

- Usually the first…

Slide 5: Programming Tip: Controlling Environment Variables



Programming Tip: Controlling Environment Variables

Use *execve*(2)

- You then reset what parts of the environment you want:

```
envp = new environment array;

if (execve(path_name, argv, envp) < 0) ...
```

Never use *system*(3) or *popen*(3)

- Unless you clean out your own environment first
- Maybe not even then…