15. For-Each Loops, Codespaces, Live Code

CPSC 120: Introduction to Programming
Pratishtha Soni ~ CSU Fullerton

Agenda

0. Sign-in sheet
1. Technical Q&A
2. For-Each Loops
3. Codespaces
4. Live Code

1. Technical Q&A

Technical Q&A

Let’s hear your noted questions about...

- This week’s Lab
- Linux
- Any other technical issues

Reminder: write these questions in your notebook during lab

2. For-Each Loops

Review: Ideal Division of Labor

- **Business Logic:** the human meaning of algorithm data
- Programs
  - **Cannot** understand business logic or design algorithms
  - Can perform tedious, repetitive work flawlessly, quickly, cheaply
- Humans
  - **Can** understand business logic and design algorithms
  - Busy-work is tedious, error-prone, expensive
- Division of Labor Best Practice
  - Humans think about business logic and algorithms
  - Computer programs do repetitive work

Loops

- **Loop:** repeat statements to handle multiple things
- Replace manual repetition
    - Writing many emails vs...
    - Algorithm:

```
PostCanvasAnnouncement(roster, message):
    for each student email in roster:
        send message to current email
```

Loop Terminology

- **Loop** (n): control flow statement that **repeats**
- Loop **body**: statement that is repeated, usually a compound statement
- **Iterate** (v): repeat
- **Iteration** (n): one individual repetition

Syntax: For-Each Loop

statement:

> **for (** *for-range-decl* **:** *container***)**
> *body-statement*

container: expression for a container object

for-range-decl: elt-type elt-identifier

Semantics:

- *elt-type* must match base type of *container*
- for each element in *container*:
  - initialize new *elt-identifier*{ current element }
  - execute *body-statement*
  - *elt-identifier* destroyed

```
// prints -2-7-8-2-0-1-1
std::vector<int> digits{ 2, 7, 8, 2, 0,
  1, 1 };
for (int d : digits) {
   std::cout << "-" << d;
}
std::cout << "\n";

// prints Mon Tue Wed Thu Fri
std::vector<std::string> weekdays{"Mon",
"Tue", "Wed", "Thu", "Fri"};

for (std::string today : weekdays) {
   std::cout << today << " ";
}
std::cout << "\n";
```

Tracing a Loop

For Loop:

```
std::vector<int> area_code{ 6, 5, 7 };
for (int x : area_code) {
   std::cout << x << "~";
}
std::cout << "\n";
```

Equivalent statements:

```
std::vector<int> area_code{ 6, 5, 7 };
{
  int x{ 6 };
  std::cout << x << "~";
}
{
  int x{ 5 };
  std::cout << x << "~";
}
{
  int x{ 7 };
  std::cout << x << "~";
}
std::cout << "\n";
```

Output:

```
6~5~7~
```

Example: Loop Through Command Line Arg’s

```cpp
std::vector<std::string> arguments(argv, argv + argc);

for (std::string argument : arguments) {
   std::cout << "[" << argument << "]";
}
std::cout << "\n";
```

```
$ ./a.out one two three
[./a.out][one][two][three]
$ ./a.out
[./a.out]
```

3. Codespaces

Codespaces Demo

1. Starting
2. Shell Commands
3. Stopping
4. Deleting

Codespaces Resource Limits

- Cloud service
- Each GitHub username has a monthly limit:
  - 180 core-hours = 90 hours
  - 20 GB-month
- Enough for labs
  - 4 labs × (2 hr class + 2? hr outside) = 16 hours/month
  - ≈ 1.5 GB/codespace ≈ 13 active at a time
- **If you don't waste it**
  - 90 hours nonstop < 4 days

Conserving Resources

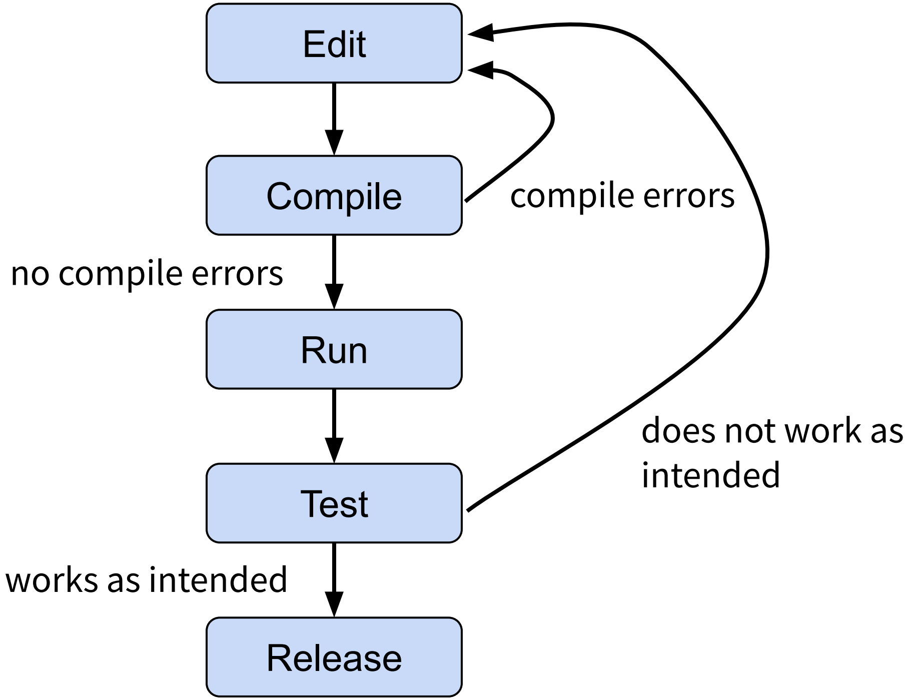
- Getting the most out of your included usage
- Stop the codespace when you pause working
- Set your idle timeout to 5 minutes
- Delete the codespace after finishing a lab
- Free to choose
  - Linux laptop with no limits
  - Codespaces, obey limits
  - Depleted resources are not an excuse

4. Live Coding

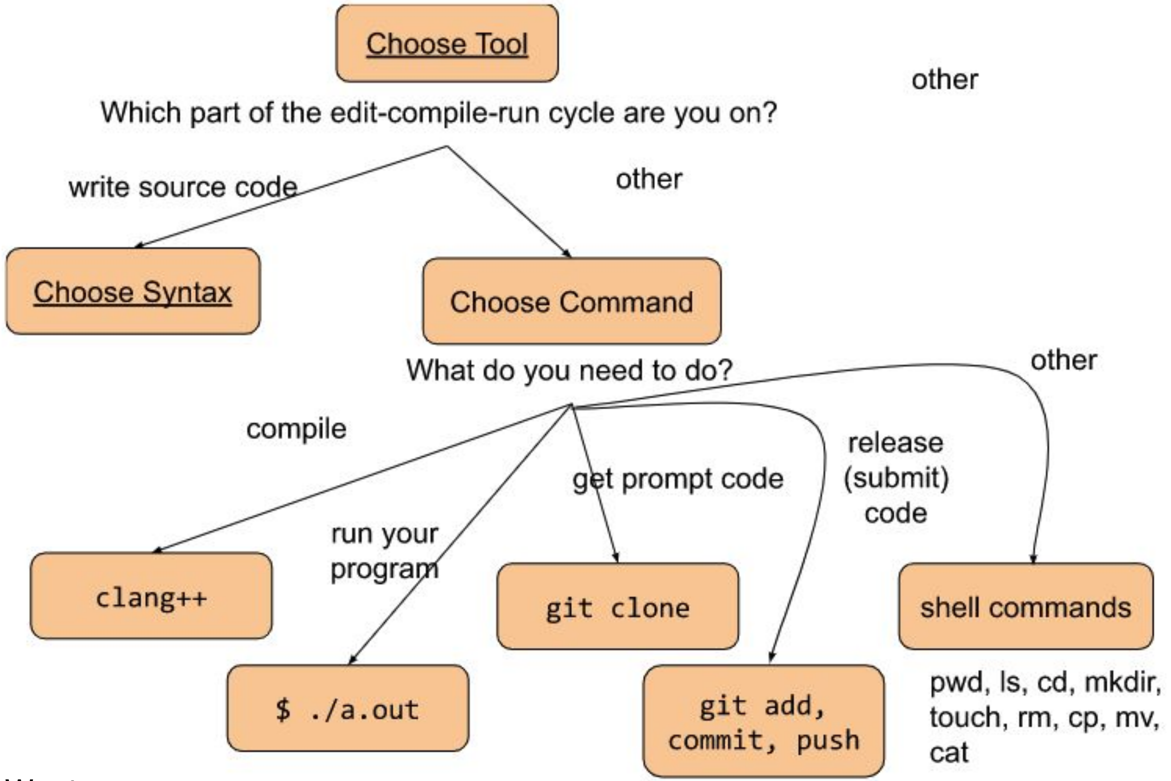
Live Coding

- Interactive
- Instructor: **driver**
- Students: **navigators**

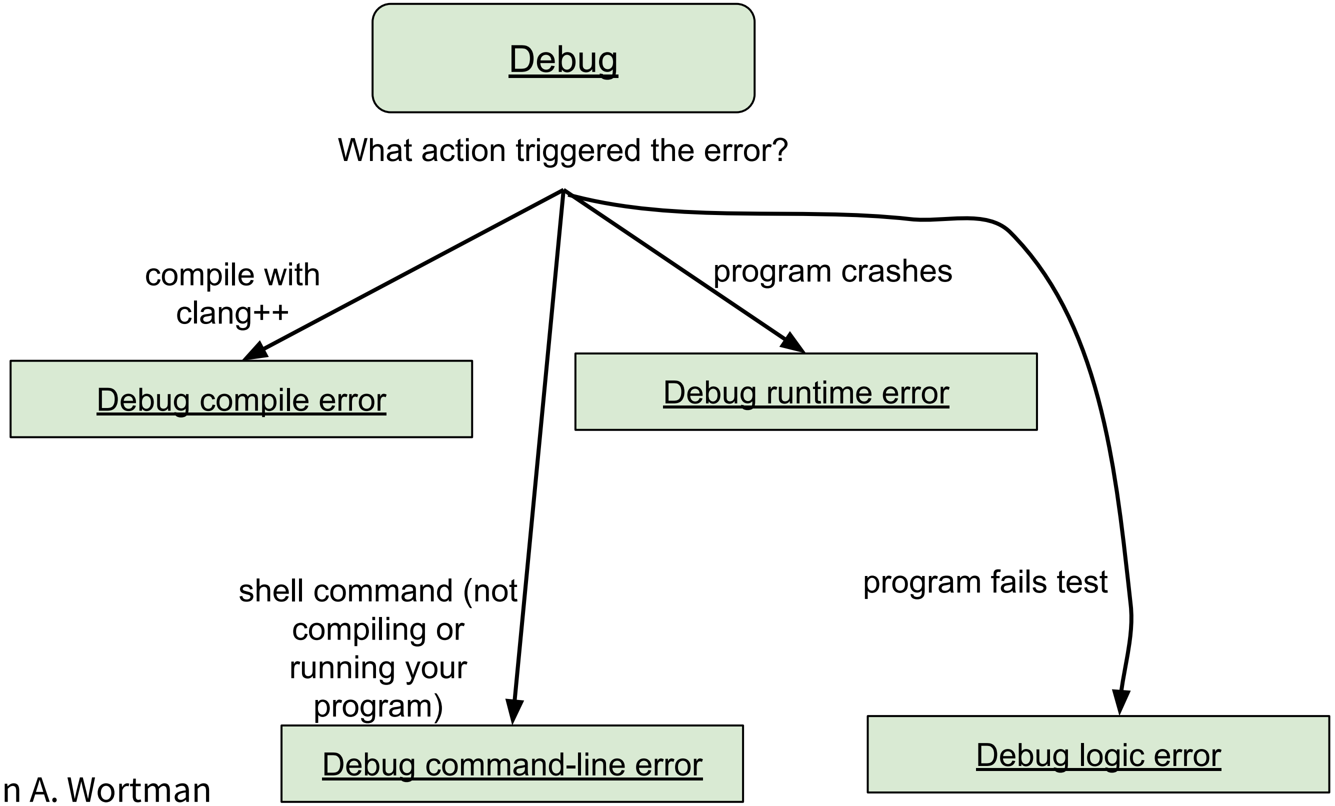
Review: The Edit-Compile-Run Cycle



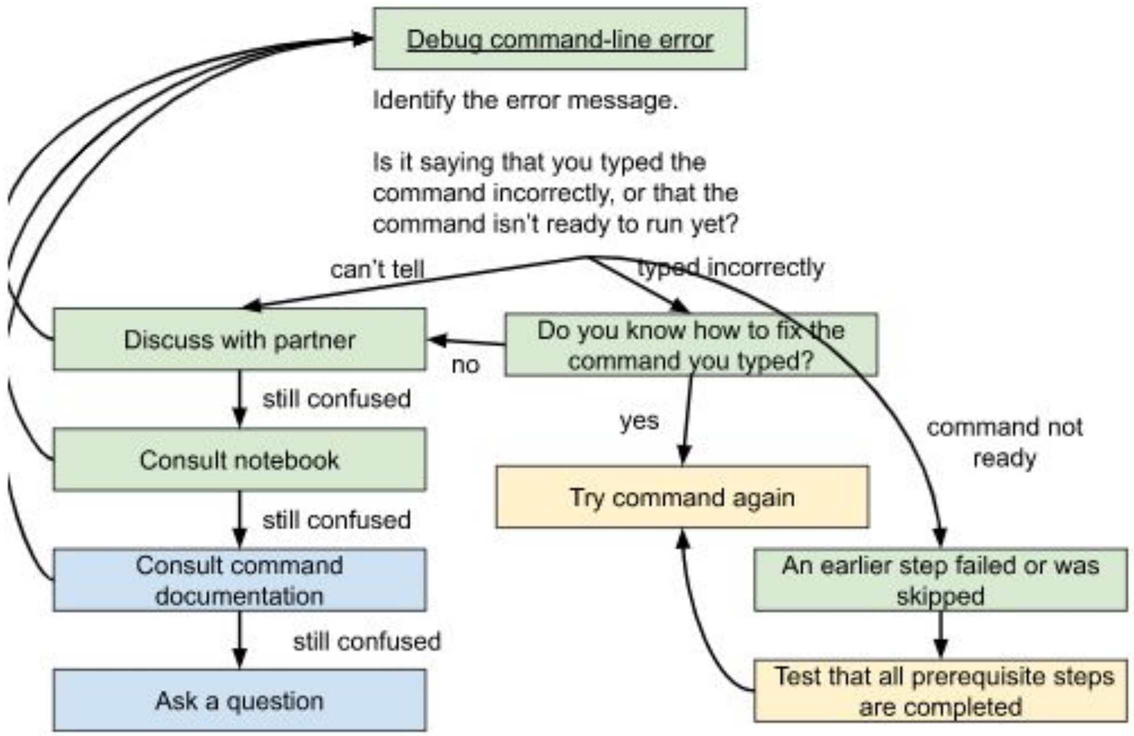
Review: Choose Tool Flowchart



Review: Debug Flowchart

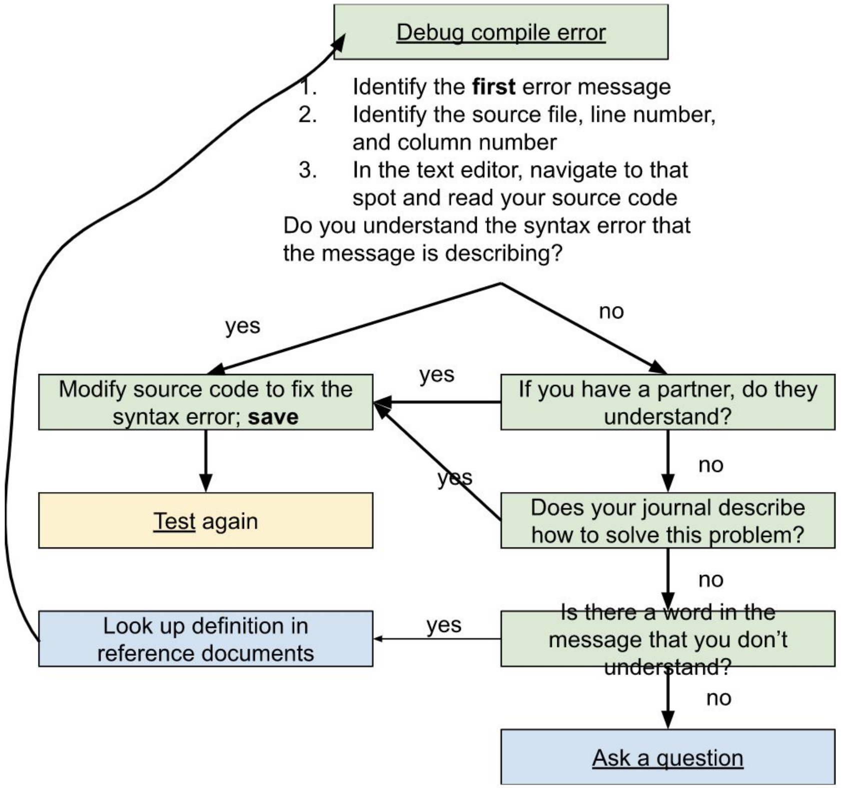


Review: Debug command error flowchart



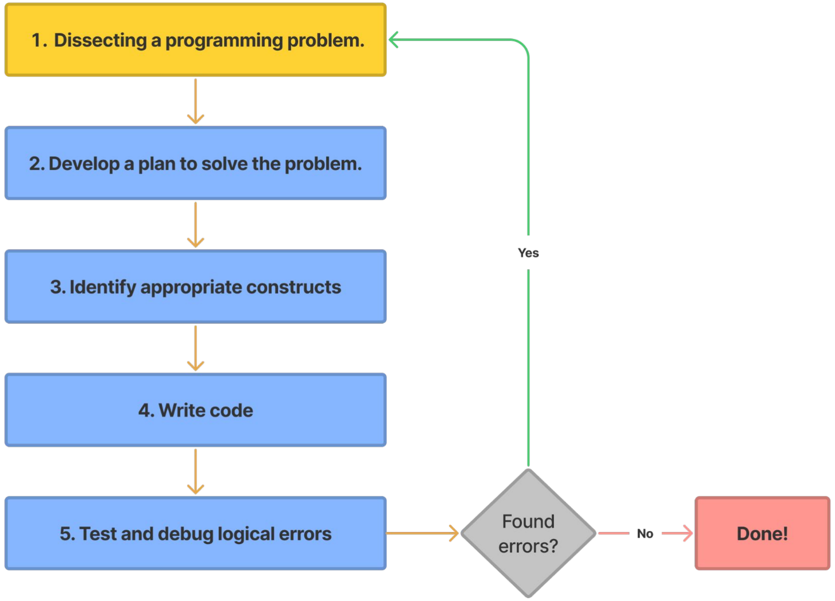
Debug command-line error
Identify the error message.
Is it saying that you typed the command incorrectly, or that the command isn't ready to run yet?
can't tell
typed incorrectly
Discuss with partner
no
Do you know how to fix the command you typed?
still confused
yes
command not ready
Consult notebook
Try command again
still confused
Consult command documentation
An earlier step failed or was skipped
still confused
Ask a question
Test that all prerequisite steps are completed

Review: Debug compile error flowchart



Debug compile error
1. Identify the **first** error message
2. Identify the source file, line number, and column number
3. In the text editor, navigate to that spot and read your source code
Do you understand the syntax error that the message is describing?
yes
no
Modify source code to fix the syntax error; **save**
If you have a partner, do they understand?
yes
no
Test again
yes
Does your journal describe how to solve this problem?
no
Look up definition in reference documents
yes
Is there a word in the message that you don't understand?
no
Ask a question

Steps for Solving a Programming Problem


1. Dissecting a programming problem.
2. Develop a plan to solve the problem.
3. Identify appropriate constructs
4. Write code
5. Test and debug logical errors
Found errors?
Yes
No
Done!

Rock, Paper, Scissors

- Game to choose between two people
- Also called *roshambo*
- Each player makes one of three gestures:
  - rock
  - paper
  - scissors
- Heuristic (rule) decides either
  - win, or
  - tie
- In the event of a tie, play again until a winner is determined

Win Heuristic

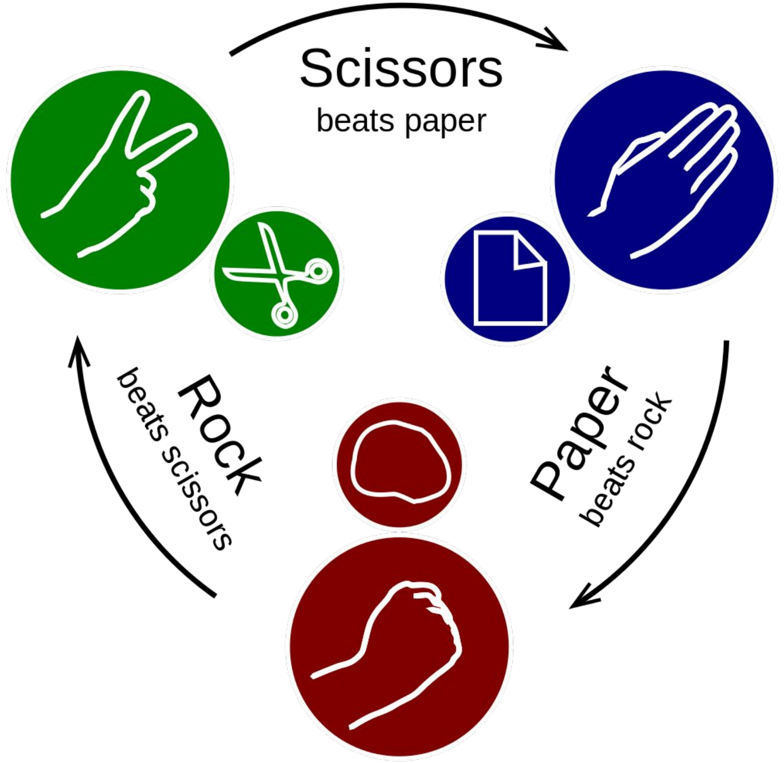


Image credit: Wikipedia

Program Requirements

1. Validate command line arguments
    - Argument 1 = player 1 move = “r”, “p”, or “s”
    - Argument 2 = player 2 move
2. Print verdict
    - player 1 wins
    - player 2 wins
    - tie

Example Input/Output

```
$ ./rps
Error: you must supply two arguments
$ ./rps a b c
Error: you must supply two arguments
$ ./rps 0 r
Error: invalid move
$ ./rps r 0
Error: invalid move
```

```
$ ./rps r s
Player 1 wins
$ ./rps s s
Tie
$ ./rps r p
Player 2 wins
```