Fast Copy in Planets in Space

- Once upon a time, there was a little robot named FastCopy
  FastCopy worked in a big science lab.
- His job was to move stars in the computer sky from one place to another.
  FastCopy was very smart. He could move 8 stars at the same time
- Sometimes, only a few stars were left behind.
  FastCopy carefully moved them one by one, so the sky stayed perfect.
- FastCopy used his shiny finger to point to the next star.
  That helped him remember where to go next
- FastCopy always followed the rules made by the scientists.
  Because of this, FastCopy was trusted by everyone.
- And so, FastCopy kept the computer sky bright and beautiful,
  the fastest and kindest robot in the future

Solution:

Usually, when a robot moves stars, he picks up one star, looks at a map, puts the star down, and then looks at the map again to see if he is done. But looking at the map for **every single star** takes too much time!

Python code to make the fastest robot in the sky.

1. Moving in Big Handfuls

We need to move stars in groups of **exactly eight**. Instead of checking a map after every star, we need to move 8 stars in a row and *only then* stop to check if you need to do more. This saves you from stopping and looking down all the time!

2. Handling the "Leftovers"

What if you have 10 stars to move? You can't move 8 and then another 8, because that would be 16!

- The code calculates the "leftovers" first (in this case, 2 stars).
- It tells your **shiny finger** exactly where to start so you can pick up those 2 stars one by one.
- Once those 2 are done, you can go back to moving big groups of 8!

3. No "Shortcuts" Allowed

We want to see every single move you make. They don't want you to use a "magic bag" (like Python's bulk copy) to move them all at once. They want us to use our hand to move **one star at a time**, just like a real computer moving tiny bits of information into its memory.

4. Why use Python?

We use Python because it’s like a clear set of instructions that both humans and robots can read easily. It acts as a "practice sky" so they can time exactly how long it takes your shiny finger to move from the input buffer to the new output region.

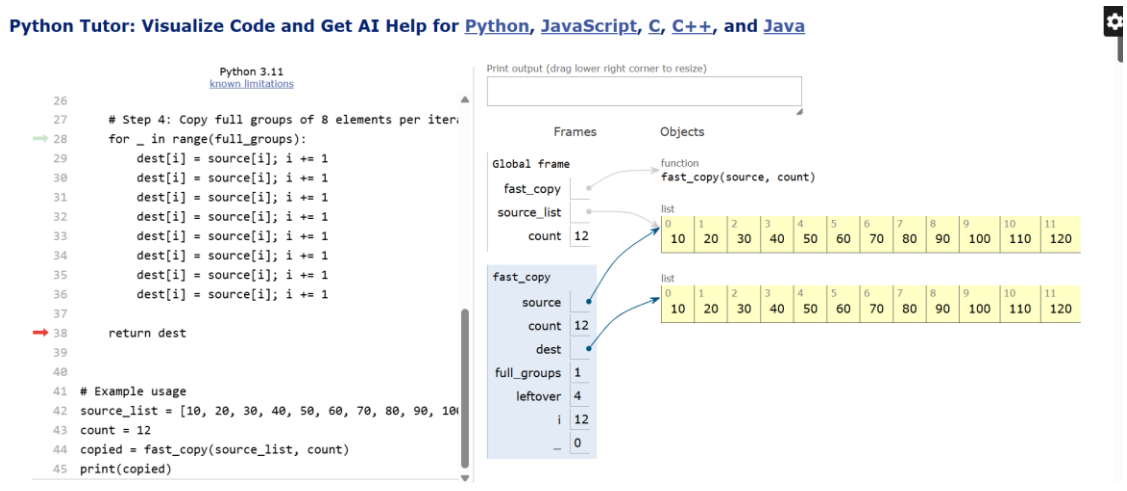
Key Differences: Python vs. C

Feature	In Python	In C (The "Real" Robot)
Control Checks	Uses if and while to count groups.	Uses a switch and do-while for even fewer checks.
The "Shiny Finger"	Uses indices like source[index].	Uses "pointers" (*source) which move automatically.
Why it's slower	Python has to "think" about every line of code.	C translates directly into physical star-moving actions.
Manual Steps	Each assignment must be written out explicitly.	The code looks "tangled" but runs very fast on a CPU.

CODE:

```
def fast_copy(source, count):
    # Step 1: Allocate destination list
    dest = [None] * count
    # Step 2: Calculate full groups of 8 and leftover
    full_groups = count // 8
    leftover = count % 8
    i = 0  # index to track position in dest/source
    # Step 3: Handle leftover elements with explicit assignments
    if leftover >= 1:
        dest[i] = source[i]; i += 1
    if leftover >= 2:
        dest[i] = source[i]; i += 1
    if leftover >= 3:
        dest[i] = source[i]; i += 1
    if leftover >= 4:
        dest[i] = source[i]; i += 1
    if leftover >= 5:
        dest[i] = source[i]; i += 1
    if leftover >= 6:
```

```
        dest[i] = source[i]; i += 1
    if leftover >= 7:
        dest[i] = source[i]; i += 1
    # Step 4: Copy groups of 8 elements per iteration
    for _ in range(full_groups):
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
        dest[i] = source[i]; i += 1
    return dest
# Example usage
source_list = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120]
count = 12
copied = fast_copy(source_list, count)
print(copied)
```



Submitted by

Ms.R.Kirthika.AP/ECE