A bit more about by value vs reference:

Terminology:

- "Formal parameter" — the variable used in the function definition:

```
int f(int x) { ... }
```

↑ formal parameter.

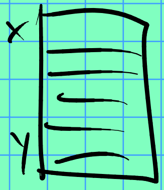
- "Actual parameter" — the expression/variable on which you evaluate the function:

```
int f(int);
int main() {
    :
    int y;
    f(y);
    :
```

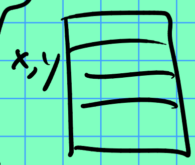
← y is the actual param. for this call to f.

Note: the relationship between the formal & actual params differs for by value & by reference calls:

By value: formal parameter is a copy of actual



By reference: formal parameter is a synonym/alias of the actual parameter.

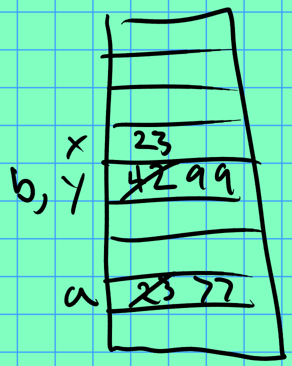


```
int f(int a, int& b) {
    :
    :
```

value ref.

```
        a = 77;
        b = 99;
          ;
    }

int main() {
    int x, y;
    x = 23; y = 42;
    f(x, y);
```

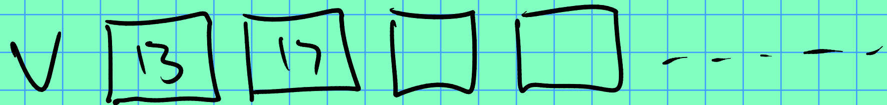


Vectors: in our lame analogy, they are like an (almost) inexhaustable array of post it notes.

Motivation: how could you write a program that reads integers from stdin & prints them in reverse order.

Problem: we don't know beforehand (compile-time) how many variables we need!

Vectors solve this problem with ease:



as program runs, you can attach new variables to the end of it.

Example: print stdin in reverse:
(See cpp file..)