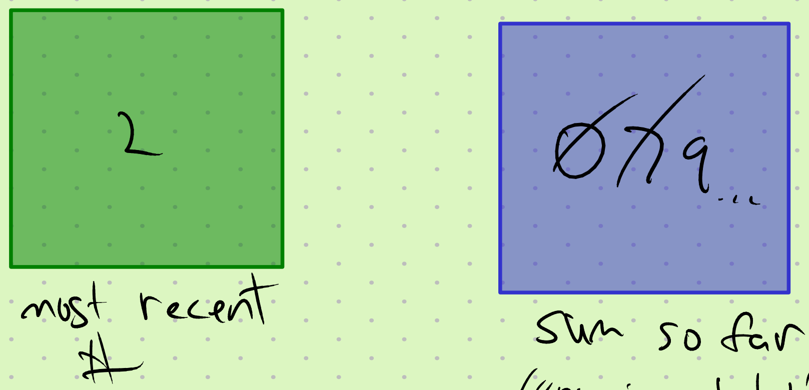
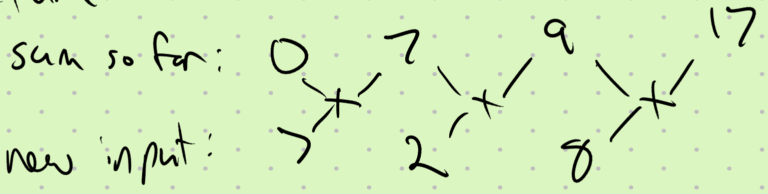
Exercise: compute sum of all integers on standard input (cin)



* upon getting a new #, just add to running total...

Picture



<u>Note</u>: this is a special case of the "fold" pattern.

General setup:

have list of values, $x_1, x_2, x_3, \ldots x_n$,

and a <u>binary operation</u> $\boxed{?}$.

Goal is to compute $s = x_1 \boxed{?} x_2 \boxed{?} \ldots \boxed{?} x_n$.

Can use the following "meta solution":

Say e is "neutral" for $\boxed{?}$, I.e.,

$x \boxed{?} e = e \boxed{?} x = x$ for any x.

(E.g. $e = 1$ for $\boxed{?} = *$, $e = 0$ for $\boxed{?} = + \ldots$)

```
s = e; // set running total to neutral
while(cin >> x)
        s = s [?] x;

cout << s;
```

Note: largest value (day 1 lecture) also fits!

$x \boxed{?} y = \max(x, y)$

$e = -\infty$ $\quad$ ($\max(x, e) = x \quad \forall x$.)

"for all"

($\approx$ INT_MIN in C/C++)

Other examples:

$\boxed{?} = *$, $e = 1$ (product)

$\boxed{?} = \min$, $e = \infty$ (min)

Quick review of all relevant tools.

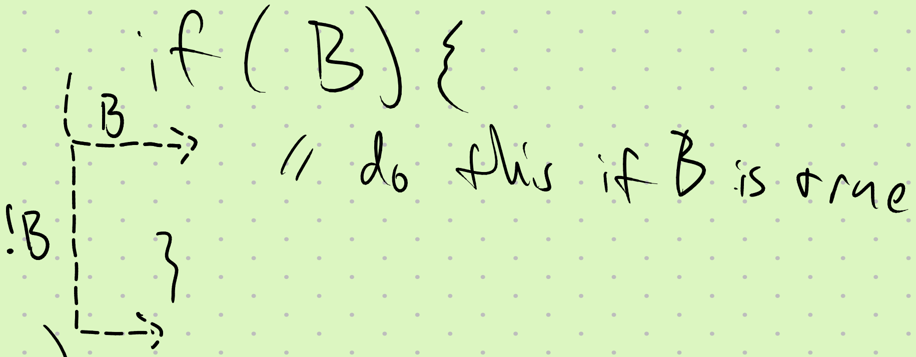
variables

```
int x;
```

(Note: datatypes so far: int, long, float, double, char, bool, string...)

assignment: `x = y;`

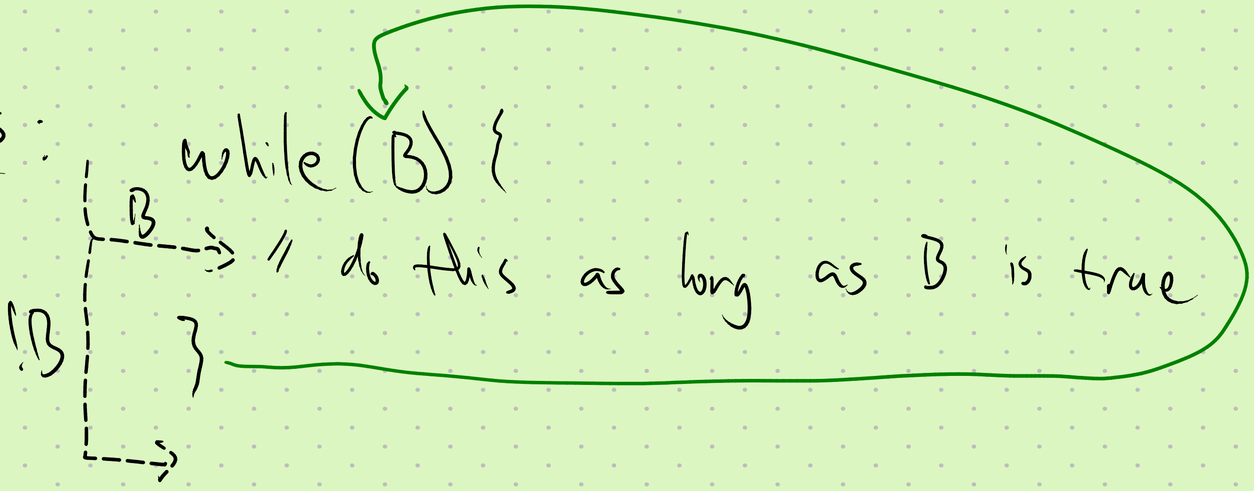
if statements
(do something at most once)



```
if (B) {
    // do this if B is true
}
```

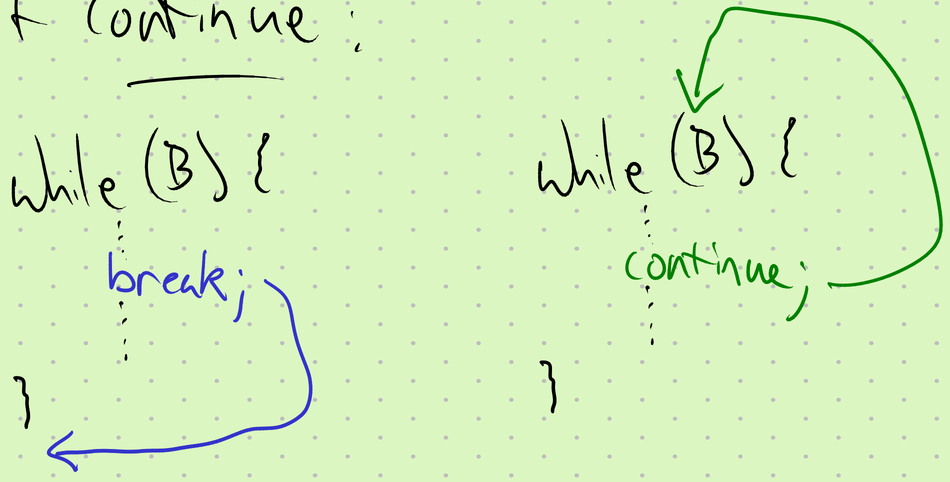
(see also if/elseif/else...)

While loops:



```
while (B) {
    // do this as long as B is true
}
```

break + continue:



```
while (B) {
    :
    break;
    :
}
```

```
while (B) {
    :
    continue;
    :
}
```