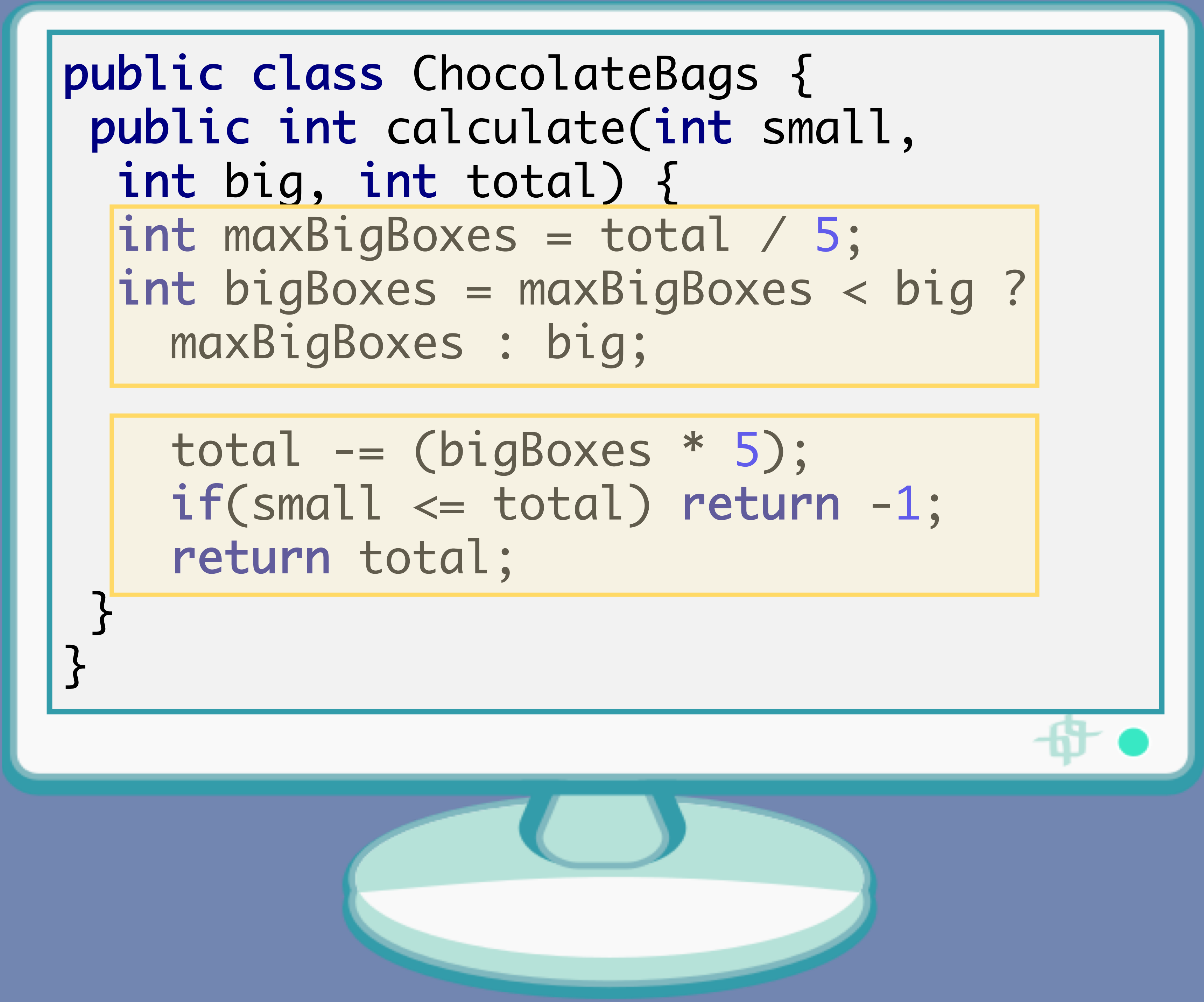
We build chocolate packages! A package should store a different number of kilos. There are small bars (1 kilo each) and big bars (5 kilos each). We should calculate the number of small bars to use, assuming we always use big bars before small bars. Return -1 if it can't be done.



```
public class ChocolateBags {
 public int calculate(int small,
  int big, int total) {
  int maxBigBoxes = total / 5;
  int bigBoxes = maxBigBoxes < big ?
    maxBigBoxes : big;

    total -= (bigBoxes * 5);
    if(small <= total) return -1;
    return total;
 }
}
```

Hmm, let me think about partitions…

☑ The total is higher than the amount of small and big bars.

Ex: small = 1, big = 1, total = 10

☑ Only big bars.

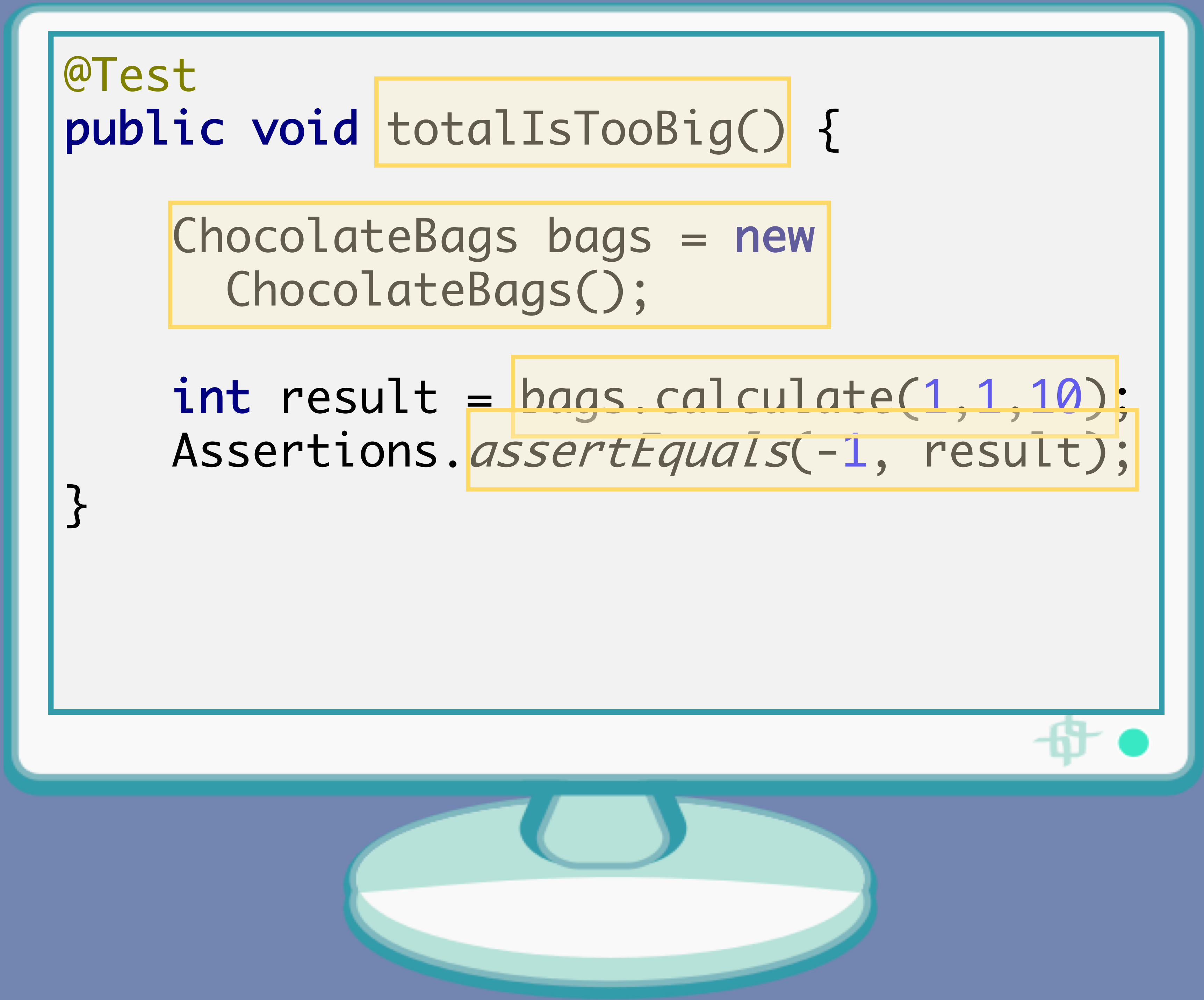
Ex: small = 5, big = 3, total = 10

☑ Need for big and small bars.

Ex: small = 5, big = 3, total = 17

☑ Only small bars.

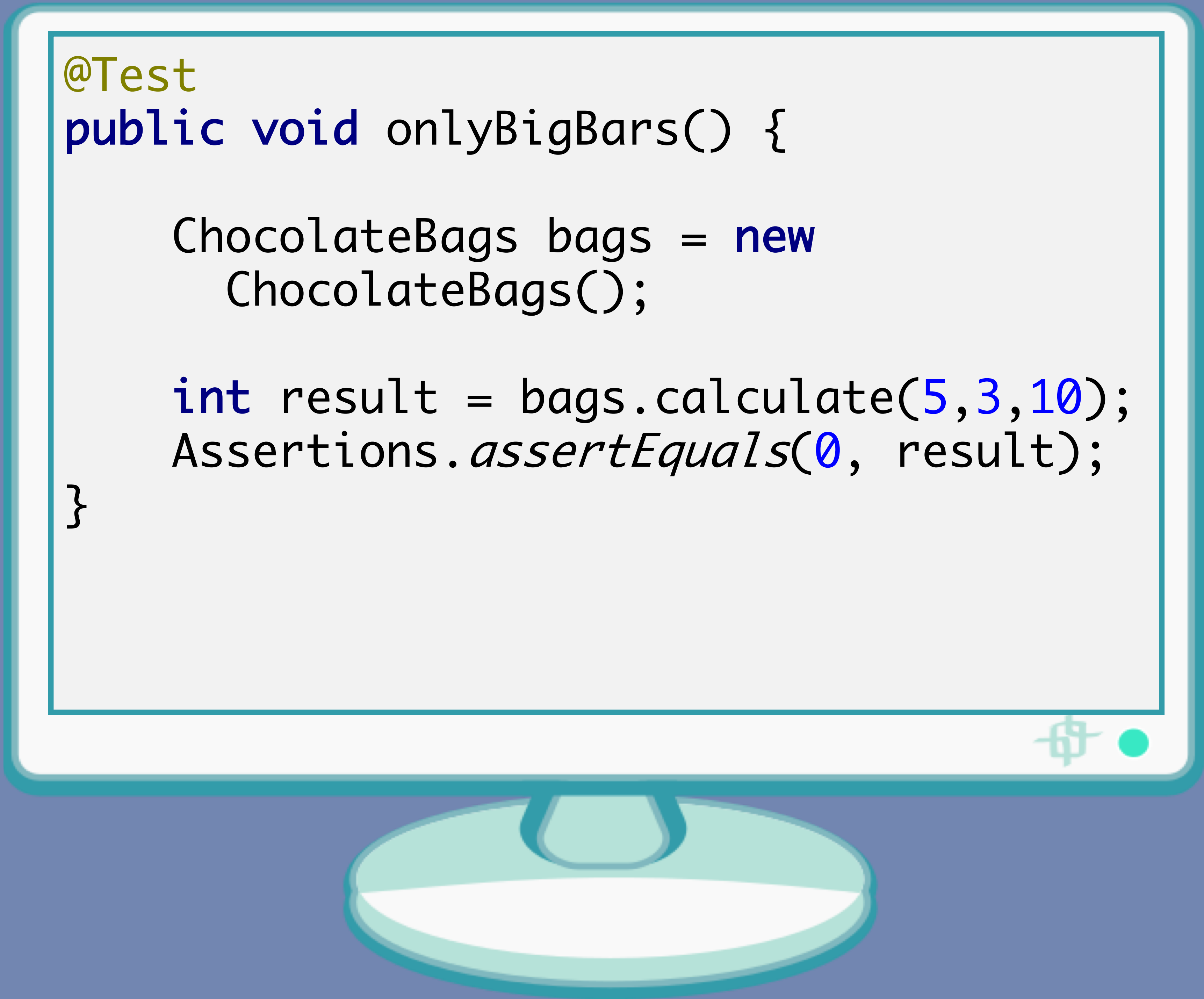
Ex: small = 4, big = 2, total = 3



```
@Test
public void totalIsTooBig() {

    ChocolateBags bags = new
      ChocolateBags();

    int result = bags.calculate(1,1,10);
    Assertions.assertEquals(-1, result);
}
```



```
@Test
public void onlyBigBars() {

    ChocolateBags bags = new
      ChocolateBags();

    int result = bags.calculate(5,3,10);
    Assertions.assertEquals(0, result);
}
```

☑ The total is higher than the amount of small and big bars.

Ex: small = 1, big = 1, total = 10

☑ Only big bars.

Ex: small = 5, big = 3, total = 10

Need for big and small bars.

Ex: small = 5, big = 3, total = 17

Only small bars.

Ex: small = 4, big = 2, total = 3