for

- **Definite Loop:**
  - A loop where we know ...

  “Count from 1 to 10..”

- **Indefinite Loop:**
  - A loop where we...
    
    tell how many times we will execute the loop:
    
    “Count up from 1 to find first multiple of 3, 4 and 18”

- for loops are often useful to neatly organizing definite loops.
Simple Example

for (int i = 0; i < 5; i++) {
    cout << "i: " << i << endl;
}

Above code identical to:
{
    int i = 0;
    while (i < 5) {
        cout << "i: " << i << endl;
        i++;
    }
}
for (int i = 1; i <= 4; i++) {
    cout << i ;
}

for (int i = 100; i > 0; i--) {
    cout << i << endl
}

for (int rows = 0; rows < 10; rows++) {
    if (rows % 3 == 0) {
        rows ++;
    }
    cout << rows;
}
Notes on for

• Variables declared in the for loop's initialization...

```cpp
for (int i = 0; i < 10; i++) {
    cout << i << endl;
}
cout << i << endl;  //..
Exercise

- Convert this into a for loop:

```cpp
int i = 0;
while (i < 10) {
    cout << i << endl;
    i++;
}
```

- Convert this into a while loop:

```cpp
for (int j = 99; j > 0; j--) {
    cout << "# bottles: " << j << endl;
}
```
break and continue

• 2 special commands can be used inside a loop's body to control execution:
  – break:..
  – continue:..

  • Will re-evaluate the condition, and keep looping.

• Avoid these when possible:
  – They complicate how the loops execute.
  – Can be useful for handling error conditions.

```cpp
int sum = 0;
while (true) {
    int val = 0;
    cout << "Enter a value: ";
    cin >> val;
    if (val < 0) {
        break;
    }
    sum += val;
}
```
Summary

• Loops run code multiple times
  – while loop
    Indefinite loop: good for running while a condition is true.
  – for loop
    Definite loop: good for running a fixed number of times.

• Use either a while or for loop
  – for loops put all looping logic on one line.
  – while loops give you more customized control.