Computer Programming

Dr. Deepak B Phatak
Dr. Supratik Chakraborty
Department of Computer Science and Engineering
IIT Bombay

Session: “for” statement in C++

Quick Recap of Relevant Topics

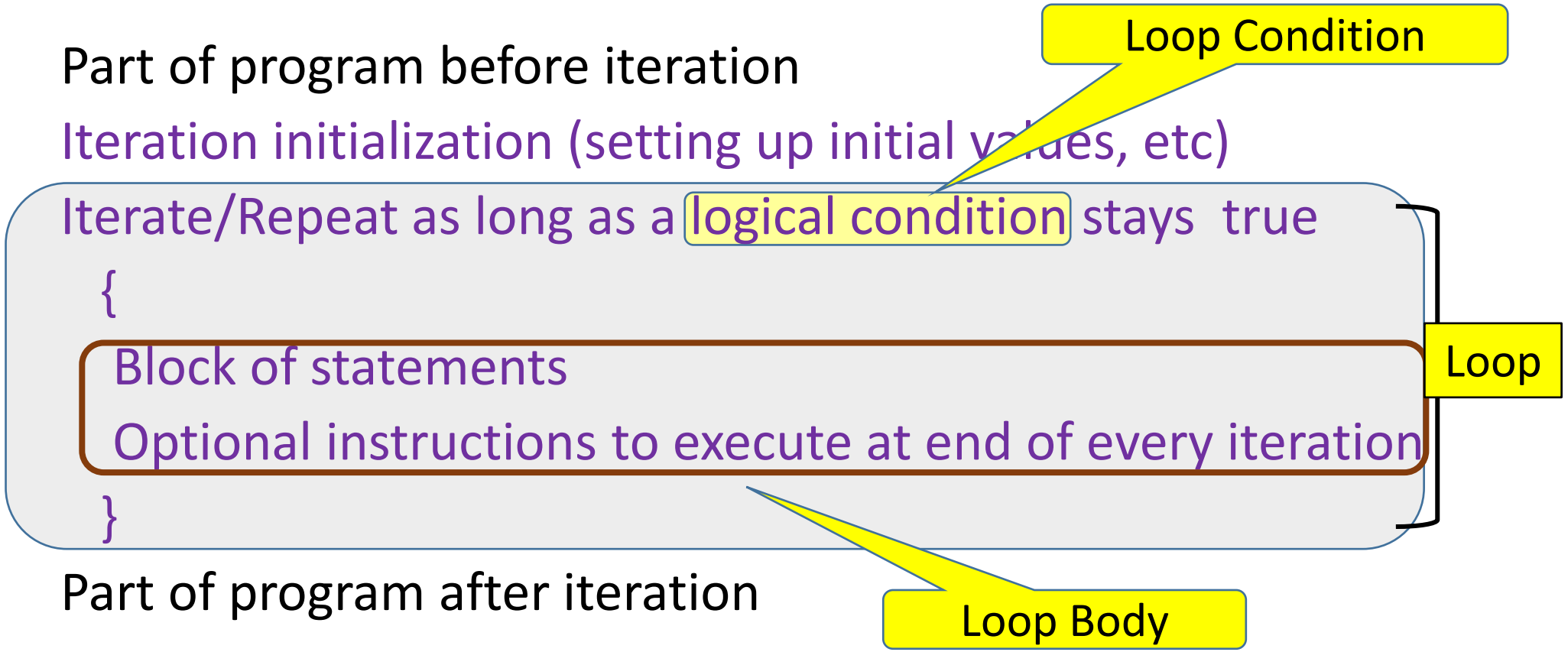
- Iteration idioms in programming
- Necessity and convenience of iteration
- “while” and “do … while …” statements in C++

Overview of This Lecture

- Iteration using “for ….” statement in C++
- “break” statement in “for” loops
- Comparison of different iteration constructs in C++

Recall Generic Iteration Construct

IIT Bombay



“for …” Statement in C++

IIT Bombay

Semi-colons not to denote end of executable statements, but to separate three parts inside “for (…..)”

Part of program before iteration

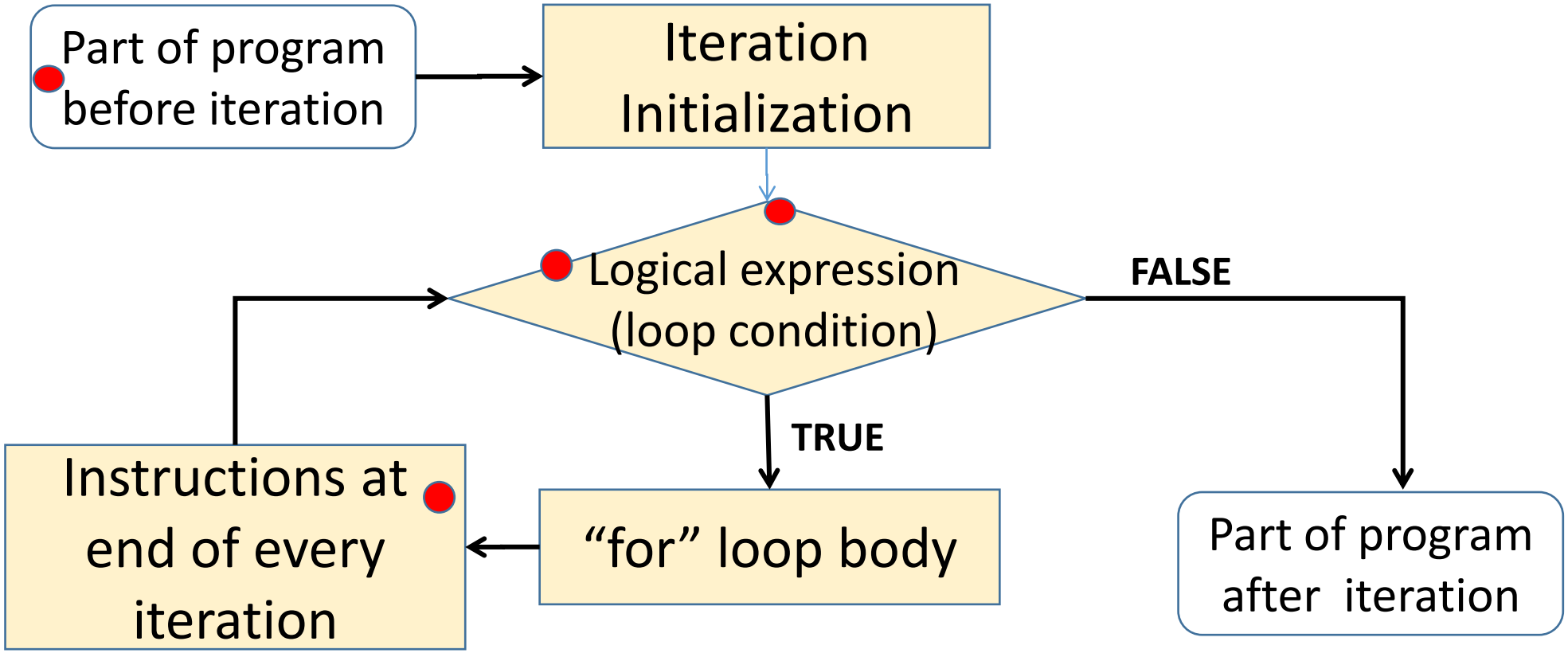
```
for (iteration initialization ; loop condition ;
        instructions to execute at end of every iteration)
{
    Block of statements  (“for” loop body )
}
```

Part of program after iteration

Note absence of semi-colon

Flowchart Representation of "for"

IIT Bombay



Points to Remember About "for"

IIT Bombay

for (initialization; loop condition; instructions after every iteration)
{ "for" loop body }

- Initialization code executed **only once** before first entry in loop
- Loop condition checked **before** executing "for" body

  Can lead to zero executions of "for" body
- Number of times loop condition is checked =

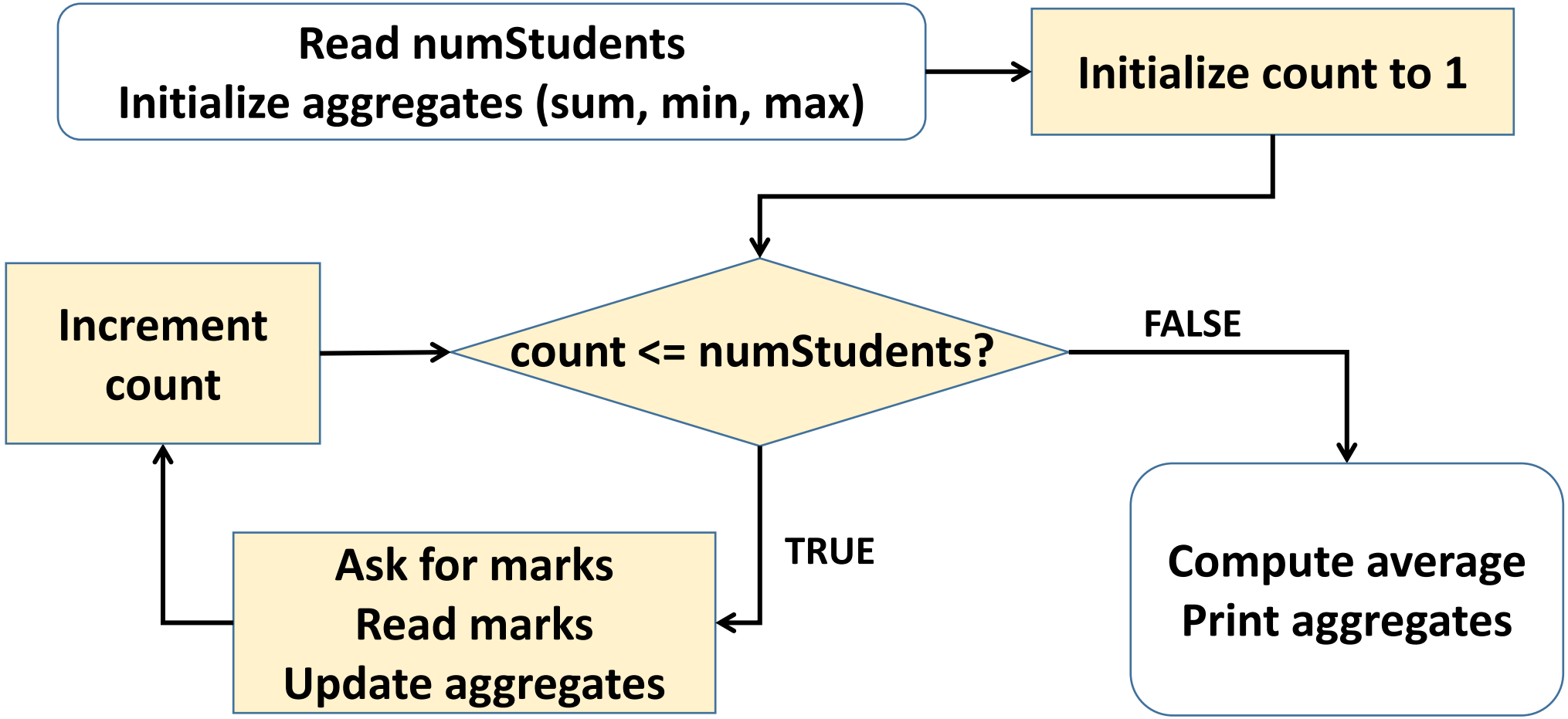
  Number of times "for" body executed + 1, if loop terminates
- Loop condition can be changed in "for" body or in "instructions after every iteration"

Revisiting The Quiz Marks Problem

Read number of students in CS101, read quiz 1 marks of all CS101 students and print their sum, average, maximum and minimum

Flowchart Representation

IIT Bombay



C++ Program with "for"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max, numStudents;
   float average, count; // Variable declarations
   cout << "Give number of students: ";    cin >> numStudents;
   for (count = 0.0; count <= numStudents; count = count + 1)
    {  cout << "Give marks of student " << count << ": ";
       cin >> marks;
      // Update sum, max, min
    }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

C++ Program with "for"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max, numStudents;
   float average, count; // Variable declarations
   cout << "Give number of students: ";    cin >> numStudents;
   for (count = 0.0; count <= numStudents; count = count + 1)
    {  cout << "Give marks of student " << count << ": ";
       cin >> marks;
      // Update sum, max, min
    }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

C++ Program with "for"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max, numStudents;
   float average, count; // Variable declarations
   cout << "Give number of students: ";    cin >> numStudents;
   for (count = 1.0; count <= numStudents; count = count + 1)
    {  cout << "Give marks of student " << count << ": ";
       cin >> marks;
      // Update sum, max, min
    }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

C++ Program with "for"

IIT Bombay

```
int main() {
   int m                    ax,
   float ave        count; // variable      arations
   cout << "G     e number of students: ";    cin >> numSt
   for (count = 1.0; count <= numStudents; count = count + 1)
   {  cout << "Give marks of student " << count << ": ";
      cin >> marks;
     // Update sum, max, min
   }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

Initialization code

Loop condition

Instruction to execute after every iteration

"for" loop body

C++ Program with "for"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max, n
   float average, count; // Variabl
   cout << "Give number of stude
   for (count = 1.0; count <= numS
    {  cout << "Give marks of student
       cin >> marks;
      // Update sum, max, min
    }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

```
sum = sum + marks;
if (count == 1) { min = marks; max = marks; }
else {
   min = (min > marks) ? marks: min;
   max = (max < marks) ? marks: max;
}
```

C++ Program with "for"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max, numStudents;
   float average, count; // Variable declarations
   cout << "Give number of students: ";    cin >> numStudents;
   for (count = 1.0; count <= numStudents; count = count + 1)
    {  cout << "Give marks of student " << count << ": ";
       cin >> marks;
      // Update sum, max, min
    }
   average = sum/count;
   // Print  average, sum, min, max
   return 0;
}
```

“break” Statement in “for” Loops

IIT Bombay

- Behaviour same as in “while” and “do … while …” loops

  Jump out of the loop immediately on executing “break”

Recall our variant of the quiz 1 marks problem

Read quiz 1 marks of CS101 students one at a time

Stop reading if -1000 is entered as marks

Print number of marks entered, sum, average, maximum and minimum

Our C++ Program with "break"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max;
   float average, count;  // Variable declarations
   for (count = 1.0;  true ; count = count + 1)
   { cout << "Give marks of student " << count << ": ";   cin >> marks;
     if (marks == -1000) { break; }
     else { … Update sum, min, max … }
   }
   average = sum/(count – 1);
   // Print  count – 1, average, sum, min, max
   return 0;
}
```

Infinite loop !!!

Jump out of "for" loop on executing "break"

Our C++ Program with "break"

IIT Bombay

```
int main() {
   int marks, sum = 0, min, max;
   float average, count;  // Variable declarations
   for (count = 1.0;  true ; count = count + 1)
   { cout << "Give marks of student " << count << ": ";   cin >> marks;
     if (marks == -1000) { break; }
     else { … Update sum, min, max … }
   }
   average = sum/(count – 1);
   // Print  count – 1, average, sum, min, max
   return 0;
}
```

These instructions skipped after executing "break"

Jump out of "for" loop on executing "break"

Next statement executed after "break"

"for" Loops with Empty Parts

IIT Bombay

```
int main() {
   int marks, sum = 0, min, ma
   float average, count:  // Variable declarations
   for (count = 1.0;          ; count = count + 1)
   { cout << "Give marks of student " << count << ": ";   cin >> marks;
     if (marks == -1000) { break; }
     else { … Update sum, min, max … }
   }
   average = sum/(count – 1);
   // Print  count – 1, average, sum, min, max
   return 0;
}
```

Skipping loop condition in "for" equivalent to "true" loop condition

“for” Loops with Empty Parts

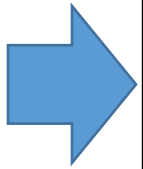
IIT Bombay

```
                    ;
for (  count = 1.0   ; true    ;  count = count + 1 )
 { cout << “Give marks of student “ << count << “: “;
   cin >> marks;
   if (marks == -1000) { break; }
   else { … Update sum, min, max … }
                                ;
 }
```

From “for …” to “while …”

IIT Bombay

```
for (initialization;
      loop condition;
      instrAfterEveryIteration)
 {
  “for” Loop Body
 }
```

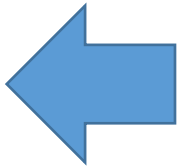


```
Initialization ;
while (loop condition)
{
   “for” Loop Body;
   instrAfterEveryIteration;
 }
```

From “while …” to “for …”

IIT Bombay

```
for (   ; loop condition;  )
 {
    “while”Loop Body
 }
```



```
while (loop condition) {
   “while” Loop Body
}
```

“for … ” to “do … while …”, and “do … while …” to “for …” can be done by recalling the transformation of “while …” to and from “do … while …”

“for … “ vs “while …”

IIT Bombay

```
for (count = 1.0;
     count <= numStudents;
     count = count + 1)
{
  // Process marks
}
```

```
count = 1.0;
while (count <= numStudents)
{
  // Process marks
  count = count + 1;
}
```

“Book-keeping” cleanly isolated

Real computation we want to do

“for … “ vs “while …”

IIT Bombay

```
for (count = 1.0;
      count <= numStudents;
      count = count + 1)
{
  // Process marks
}
```

```
count = 1.0;
while (count <= numStudents)
 {
   // Process marks
   count = count + 1;
 }
```

Real computation we want to do

“Book-keeping” mixed up

“for …” vs “while …”

IIT Bombay

- Often a programmer’s choice dependent on the context
- In general, a good idea to separate book-keeping from real computation
  - Prefer “for …” loops
- However, loop condition may not simply be based on book-keeping
  - Real computation in loop body may determine loop condition
  - Prefer “while …” loops

Summary

- "for …" statement in C++
  - Variants of "for …" statement
- Use of "break" statement in "for …" statements
- Comparison with "while …" and "do … while …" statements