

for Loops

Lecture 13

Sections 5.1 - 5.6

Robb T. Koether

Hampden-Sydney College

Wed, Sep 25, 2019

1 Loops Controlled by a Counter

2 `for` Loops

3 Assignment

Outline

1 Loops Controlled by a Counter

2 `for` Loops

3 Assignment

Loops Controlled by a Counter

- If we know in advance the number of times a loop should be executed, then we can count the iterations and quit at the proper time.
- Establish a counter and do the following.
 - **Initialize** the counter to 0.
 - **Test** the counter on each iteration.
 - **Increment** the counter.

Loops Controlled by a Counter

- If the counter controls the loop, then the testing does not involve the input.
- Therefore, the pattern prompt-read-test-action is no longer in effect.
- Indeed, there may not be any input.
- If there is input, then it is typically part of the action.

Unrolling the Loop

- The sequence should be
 - 1 **get limit** – Get the desired number of iterations.
 - 2 **Initialize** – Set the counter to 0.
 - 3 **test** – Compare the counter to the limit.
 - 4 **action** – Execute the body of the loop.
 - 5 **increment** – Increment the counter.
 - 6 **test** – Compare the counter to the limit.
 - 7 **action** – Execute the body of the loop.
 - 8 **increment** – Increment the counter.
 - 9 **test** – Compare the counter to the limit.
 - 10 \vdots


Loops Controlled by a Counter

```
get limit  
initialize ctr  
test  
action  
increment  
test  
action  
increment  
:
```

The “unrolled” loop

Loops Controlled by a Counter

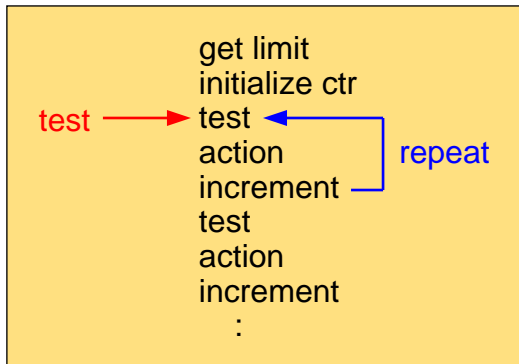
```
get limit
initialize ctr
test ←
action
increment
test
action
increment
:
```



The diagram illustrates a loop structure. A blue arrow originates from the 'increment' step of the first iteration and points back to the 'test' step, indicating a loop. The word 'repeat' is written in blue next to the arrow.

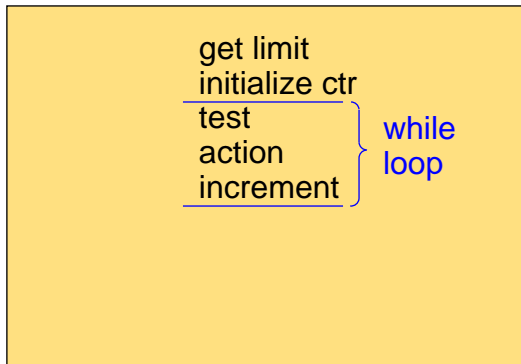
Repeat after incrementing

Loops Controlled by a Counter



The *test* is here

Loops Controlled by a Counter



This must be the **while** loop

Loops Controlled by a Counter

Loops Controlled by a Counter

```
int limit;  
cin >> limit;           // Get the limit  
int counter = 0;        // Initialize the ctr  
while (counter < limit) // Test  
{  
    action              // Action  
    counter++;           // Increment  
}
```

Example of a Counter

- Example
 - `CounterSum.cpp`

Outline

1 Loops Controlled by a Counter

2 `for` Loops

3 Assignment

The **for** Statement

The **for** Statement

```
for (init-ctr; test-ctr; incr-ctr)  
{  
    action  
}
```

- The form of the **for** statement.

The **for** Statement

Example

```
int limit;
cin >> limit;
int sum = 0;
for (int i = 0; i < limit; i++)
{
    cout << "Enter a number: ";
    cin >> number;
    sum += number;
}
```

- Add up 10 numbers.

Examples of a `for` Loop

- Examples

- `ForSum.cpp`
- `CountLetters.cpp`

Outline

1 Loops Controlled by a Counter

2 `for` Loops

3 Assignment

Assignment

Assignment

- Read Sections 5.1 - 5.6.