UCT Department of Computer Science

Computer Science 1015F

Iteration

Hussein Suleman
<hussein@cs.uct.ac.za>
March 2009

Problem

- Output the 7x table.

What is Iteration?

- Executing the same basic task or set of statements multiple times.
  - e.g., print the 7x table (from 1 to 12)

Counter-controlled Loops

- Counter-controlled loops execute for a fixed number of times.
- A special counter variable is used to control the loop and may be referred to within the loop.
- Java provides the “for” statement as a counter-controlled loop.

The “for” statement

```
for ( initialisation statement;
      condition;
      increment statement )
{
   statements …
}
```

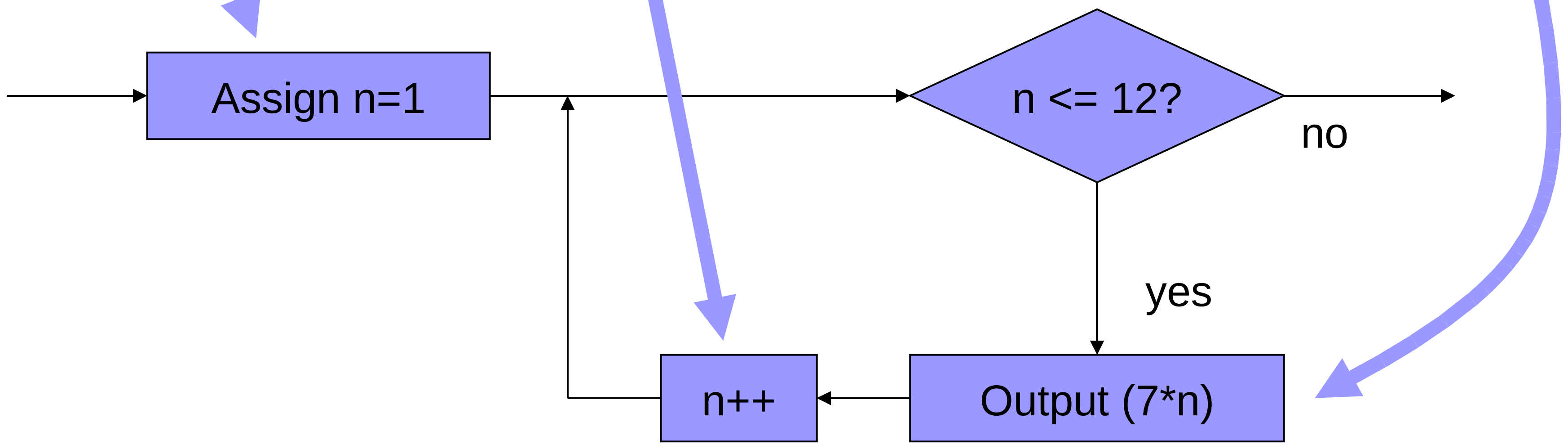
Example Usage

```
int n;

for ( n=1; n<=12; n++ )
{
   System.out.println (n + “ x 7 = “ + (n*7));
}
```

```
Output:
1 x 7 = 7
2 x 7 = 14
3 x 7 = 21
...
```

Flowchart vs Java

```
int n;

for ( n=1; n<=12; n++ )
{
   System.out.println (n + “ x 7 = “ + (n*7));
}
```



Assign n=1

n <= 12?

no

yes

Output (7*n)

n++

Additional “for” syntax

- We can define a variable in the initialisation section, which is local to the body of the loop.
  - `for ( int i=1; i<=10; i=i+1 )`
- Multiple comma-separated expressions can appear in the “increment” section, even decrements.
  - `for (int i=10; i>0; i--)`
  - `for (int i=1,j=7; i<=12; i++,j+=7)`

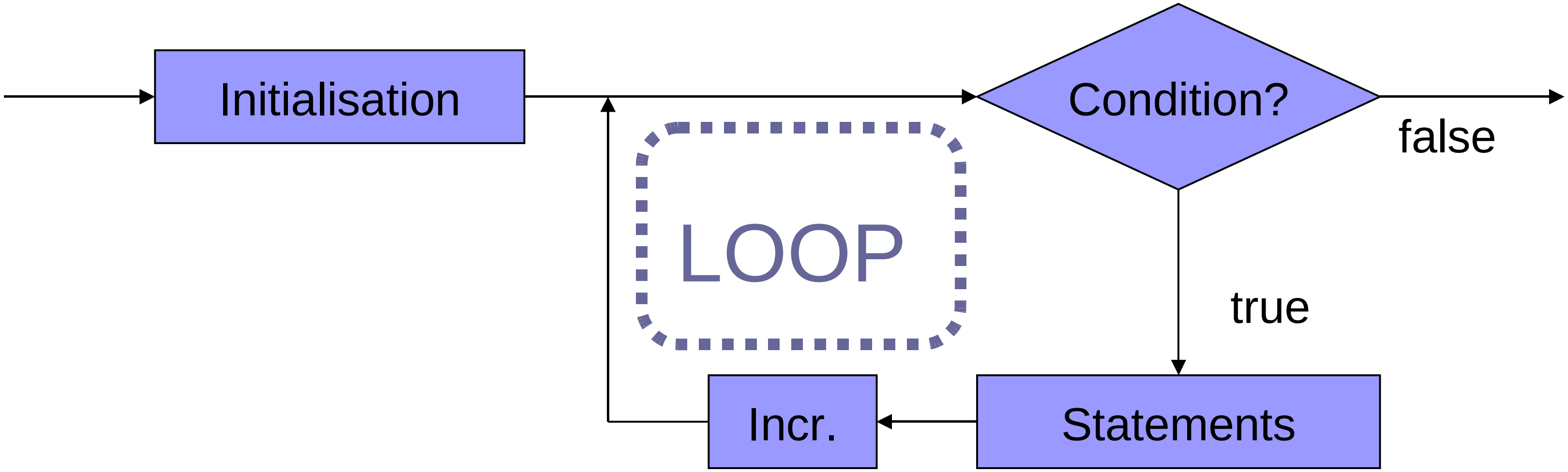
Problem revisited

- Output the `n x table` for any integer value of n.

Solution?

```
for ( int i=1; i<=12; i++ )
   System.out.println (i+" x "+n+" = "+(n*i));
```

General Semantics of “for”



Problem

- Find the product of the integers from 1..n, corresponding to n!.

Problem

- Calculate a^b using a **for** loop, assuming that a is a float and b is an integer.

Nesting of statements

- **for** and **if** are both statements, therefore they can each appear within the statement body.

```
for ( int i=1; i<=10; i++ )
{ if (a<b) max=b; }


if (a<b)
{ for (int i=1; i<=10; i++ ) ... }


for ( int i=1; i<=10; i++ )
   for ( int j=1; j<=10; j++ )
```

Nested loops

- Where a task is carried out multiple times and a subtask within that is carried out multiple times.
  - Example:

    Draw a triangle of arbitrary height on the screen, such as:

```
*
* *
* * *
* * * *
```

Problem

- Write programs to generate (on the screen) the following triangles of user-specified height:

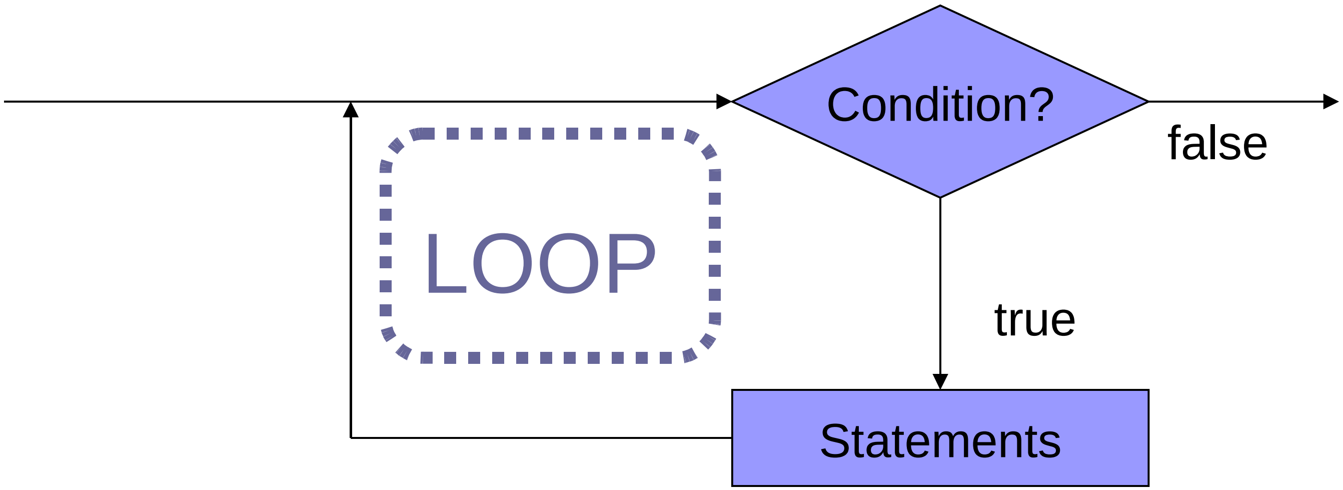
```
      *                 *                 * * * *
    * *               * * *                 * * *
  * * *             * * * * *                 * *
* * * *           * * * * * * *                 *
```

Condition-controlled Loops

- If we do not know the number of iterations a priori (in advance), we can use a condition-controlled (or event-controlled) loop - where the loop executes while a condition is true
- Two statements:

```
while (<condition>)  { <statements> }
do { <statements> } while (<condition>)
```

“while” Example



```
int sum = 0;
int num = input.nextInt ();
while (num != 0)
{
   sum = sum + num;
   num = input.nextInt ();
}
```

Problem

- Approximate the logarithm (with a base of 10) of an integer using repeated division.

Problem

- Approximate the logarithm (with a base of 10) of an integer using repeated division.
- Design a user interface where the user can continue to ask for logarithms until a value of 0 is supplied.

Menus

- A menu is a list of choices presented to the user, with the means to select one.
  - Example:

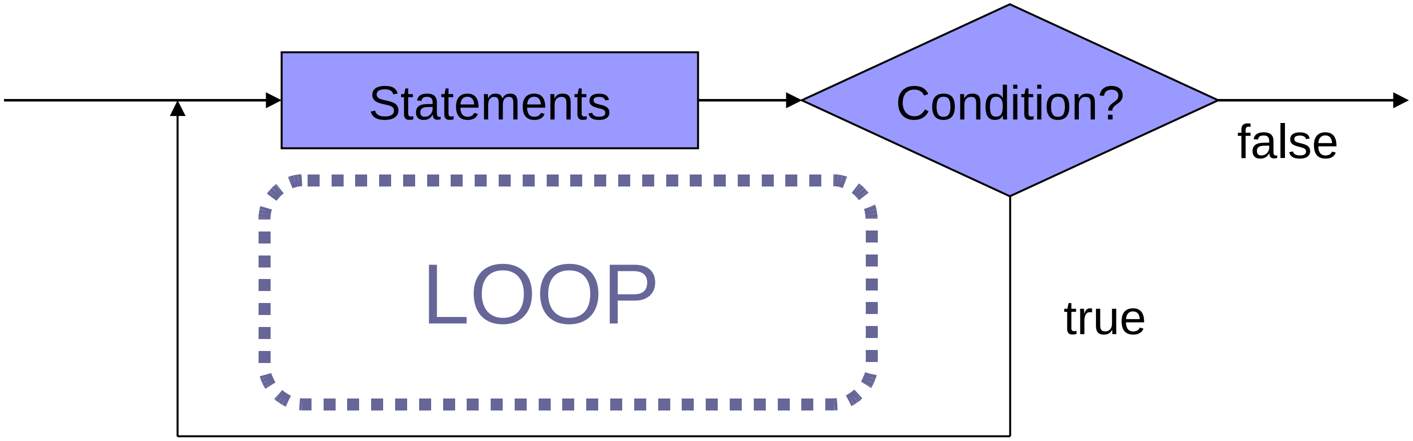
```
Souper Sandwich Menu
1. Chicken, cheese and chilli sauce
2. Chicken and cheese
3. Cheese
4. Exit Program
Enter the sandwich number:
```

Menu Example

```
Menu souper = new Menu ();
souper.print ();    // output options
int choice = input.nextInt (); // get selection
while (choice != 4) // continue until exitted
{
   System.out.println (); // leave a line
   switch (choice)  // output ingredients
   {
      case 1 : System.out.println ("Add chilli");
      case 2 : System.out.println ("Add chicken");
      case 3 : System.out.println ("Add cheese");
   }
   souper.print (); // output options
   choice = input.nextInt (); // get selection
}
```

“do..while” statement

- When the “loop body” is going to be executed at least once, we can check the condition after the loop (instead of before).



“do..while” Example

```
Menu souper = new Menu ();
int choice;
do {
   souper.print ();       // output options
   choice = input.nextInt (); // get selection
   System.out.println (); // leave a line
   switch (choice)        // output ingredients
   {
      case 1 : System.out.println ("Add chilli");
      case 2 : System.out.println ("Add chicken");
      case 3 : System.out.println ("Add cheese");
   }
} while (choice != 4)     // continue until exitted
```

Problem

- Find the reverse of an integer.
- For example, the reverse of the integer 12345 is 54321 and the reverse of 98 is 89. Use only integer manipulations - do not convert the number to a String.

Infinite Loops

- Loops where the condition is always true
  - Example:

```
while (true)
{
   System.out.println (“Wheeee!”);
}


do { … } while (true);
for ( int i=1; i<10; ) { … }
```

break

- exits immediately from a loop
  - Example:

```
int i = 0;
while (true)
{
   i++;
   System.out.println (i);
   if (i == 10) break;
}
```

continue

- immediately starts next iteration
  - Example:

```
for ( int i=0; i<=10; i++ )
{
   if (i % 3 == 0)
      continue;
   System.out.println (i);
}
```

Selecting Loops

- General Rules:
  - When you know the number of iterations, use a “for”
  - When the iterations depend on a condition,
    - use a “do..while” if the loop must execute at least once
    - otherwise, use a “while”

Converting Loops

- How do we write the equivalent of
  - “while” using “for”
  - “do..while” using “for”
  - “for” using “while”
  - “do..while” using “while”
  - “for” using “do..while”
  - “while” using “do..while”