L7. Iteration with While-Loops

Problem Solving With the For-Loop

```
for count = expression : expression
    The calculation to be repeated.
end
```

Question Time

How many lines of output are produced by the following script.

```
for k=100:200
    if rem(k,2)~=0
        disp('k');
    end
end
```

There is a line of output for every Odd number between 100 and 200.

Answer = 50

For-Loop Shortcoming

When you use a for-loop, you need to know the exact extent of the repetition.

```
for count = expression : expression
```

OK for Many Situations

Here is a typical for-loop Problem:

Simulate the tossing of a fair coin 100 times and print the number of "Heads"
% Running sum...
H = 0;
for tosses = 1:100
    r = rand;
    if r < .5
        % Agree that this means "heads"
        H = H + 1;
    end
end
fprintf('H = %2d
',H)

Not OK in Other Situations
Simulate the game of "Gap10":
Toss a fair coin until
| #Heads - #Tails | = 10
Score = number of required tosses
The number of required tosses
is not known in advance.

What We Need
A loop that "shuts down"
as soon as |H-T| = 10.

H = 0; T = 0; tosses = 0;
while abs(H-T)<10
    r = rand;
    tosses = tosses + 1;
    if r < .5
        H = H + 1;
    else
        T = T + 1;
    end
end
fprintf( ... )

How a While-Loop Works
Warm-up exercise:
Any for-loop can be written as a while-loop.

A Simple For-loop
s = 0;
for k=1:5
    s = s + k;
    fprintf('%2d %2d
',k,s)
end
1 1
2 3
3 6
4 10
5 15
The While-loop Equivalent

\[ k = 0; s = 0; \]
while \( k < 5 \)
\[ k = k + 1; s = s + k; \]
\[ \text{fprintf(''\%2d  \%2d\n'',k,s)} \]
end

\[
\begin{array}{c}
1 & 1 \\
2 & 3 \\
3 & 6 \\
4 & 10 \\
5 & 15 \\
\end{array}
\]

How it Works in Detail

\[ k = 0; s = 0; \]
while \( k < 5 \)
\[ k = k + 1; s = s + k; \]
\[ \text{fprintf(''\%2d  \%2d\n'',k,s)} \]
end

\[
\begin{array}{c}
0 & 0 \\
1 & 1 \\
2 & 3 \\
3 & 6 \\
\end{array}
\]

Is \( k < 5 \) true? Yes. Execute the loop body.

Is \( k < 5 \) true? Yes. Execute the loop body.

Is \( k < 5 \) true? Yes. Execute the loop body.

Is \( k < 5 \) true? Yes. Execute the loop body.
How it Works in Detail

$k = 0; s = 0$

while $k < 5$
  $k = k + 1; s = s + k$
  fprintf('%2d %2d\n',k,s)
end

<table>
<thead>
<tr>
<th>k</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

Is $k < 5$ true? Yes. Execute the loop body.

<table>
<thead>
<tr>
<th>4</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5</th>
<th>15</th>
</tr>
</thead>
</table>

Is $k < 5$ true? No, done with loop.

A Modified Problem

Print the smallest $k$ so that

$$1 + 2 + 3 + \ldots + k \geq 30$$

How it Works in Detail

$k = 0; s = 0$

while $s < 30$
  $k = k + 1; s = s + k$
end

fprintf('%2d %2d\n',k,s)

<table>
<thead>
<tr>
<th>k</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Is $s < 30$ true? Yes. Execute loop body.

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
</tr>
</thead>
</table>

Is $s < 30$ true? Yes. Execute loop body.
Defining Variables

```matlab
k = 0; s = 0;
while s < 30
    %s is the sum 1+ ... + k
    k = k + 1; s = s + k;
end
```

This “property” is true all during the loop

Spotting a While Situation

As \( n \) increases, \( \text{InnerA} \) and \( \text{OuterA} \) approach \( \pi \), the area of the unit circle.
When will \(|\text{OuterA} - \text{InnerA}| <= .000001\)?

PseudoCode Development

Identify the repetition and a criteria that says “keep iterating”.

Pattern for doing something an Indefinite number of times

```matlab
% Initialization

while \( \text{not-stopping signal} \)
    % do something
    % update status (variables)
end
```
**Question Time**

What is the last line of output produced by this script?

```matlab
n = 5
while n>1
disp('I dunno')
  if rem(n,2)==0
    n = n/2
  else
    n = 3*n+1
  end
end
```

A. 1  B. 2  C. 4  D. 16  E. I dunno

**Two More While Examples**

Each motivated by the limitations of the for-loop

**Example 1: Up/Down Sequence**

Pick a random whole number between one and a million. Call the number `n` and repeat this process:

- If `n` is even, replace `n` by `n/2`.
- If `n` is odd, replace `n` by `3n+1`

Does it ever take more than 1000 updates to reach one?

**Aside: Random Integers**

How do we generate a random integer from an interval?

```matlab
n = ceil(1000000*rand)
```

**Need the Built-In Function ceil**

<table>
<thead>
<tr>
<th>a</th>
<th>floor(a)</th>
<th>ceil(a)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.9</td>
<td>15</td>
<td>16</td>
</tr>
<tr>
<td>12.0</td>
<td>12</td>
<td>12</td>
</tr>
</tbody>
</table>

`floor`: next smallest integer

`ceil`: next biggest integer

**Random Integers**

```matlab
n = ceil(1000000*rand)

x is random real, 0 < x < 1
x = rand

y is random real, 0 < y < 10^6
y = 100000*x

n is rand integer from 1, ..., 10^6
n = ceil(y)
```
The Central Repetition:

if rem(n,2) == 0
    n = n/2;
else
    n = 3*n + 1
end

Note cycling once n == 1:
1, 4, 2, 1, 4, 2, 1, 4, 2, 1, 4, 2, 1, ...

Shuts Down When n==1:

step = 0;
while n > 1
    if rem(n,2) == 0
        n = n/2;
    else
        n = 3*n + 1;
    end
    step = step+1;
    fprintf(' %4d %7d
', step, n)
end

Cycles after n ==1

for step = 1:1000
    if rem(n,2) == 0
        n = n/2;
    else
        n = 3*n + 1;
    end
    fprintf(' %4d %7d
', step, n)
end

Example 2: Square Roots

Pick a random number x between one and a million. Compute the sqrt(x) by

L = x; W = 1;
Repeat until relative error in L <= 10^-15:
L = (L+W)/2; W = x/L;
Print relative error in L

Shuts Down After Convergence

s = sqrt(x); L = x; W = 1; k = 0;
while k==0 || relErr > 10^-15
    k = k+1;
    L = (L+W)/2; W = x/L;
    relError = abs(L-s)/s
end

Error: relErr not initialized when the while Loop is entered.
**Shuts Down After Convergence**

\[
\begin{align*}
s &= \sqrt{x}; \quad L = x; \quad W = 1; \quad k = 0; \\
\text{while } k == 0 \text{ || relErr > } 10^{-15} \\
&\quad k = k + 1; \\
&\quad L = (L + W)/2; \quad W = x/L; \\
&\quad \text{relError} = \text{abs}(L-s)/s \\
\end{align*}
\]

During the first check of the condition, \( k == 0 \) is true. Matlab doesn't bother to check the relErr comparison since the or is true. No prob that relErr uninitialized.

**Nested Loop Problem**

On average, how many coin tosses are there in a game of Gap10?

Estimate by simulating 10,000 games.

**PseudoCode**

```matlab
sum = 0
for k=1:10000
    Simulate a game of Gap10 and assign to the variable tosses the number of required tosses.
    sum = sum + tosses;
end
p = sum/10000
```

```matlab
H = 0;  T = 0;  tosses = 0;
while abs(H-T)<10
    r = rand;
    tosses = tosses + 1;
    if r < .5
        H = H + 1;
    else
        T = T + 1;
    end
end
```