CPSC 1045_002

Loops

Instructor: *Hazra Imran*

Loops

```
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
document.write (“All this typing gets boring <br/>”);
```

```
Do this block 12 times {
document.write (“All this typing gets boring <br/>”);
 }
```

Loops

```
Do this block 12 times {
document.write (“All this typing gets boring <br/>”);
 }
```

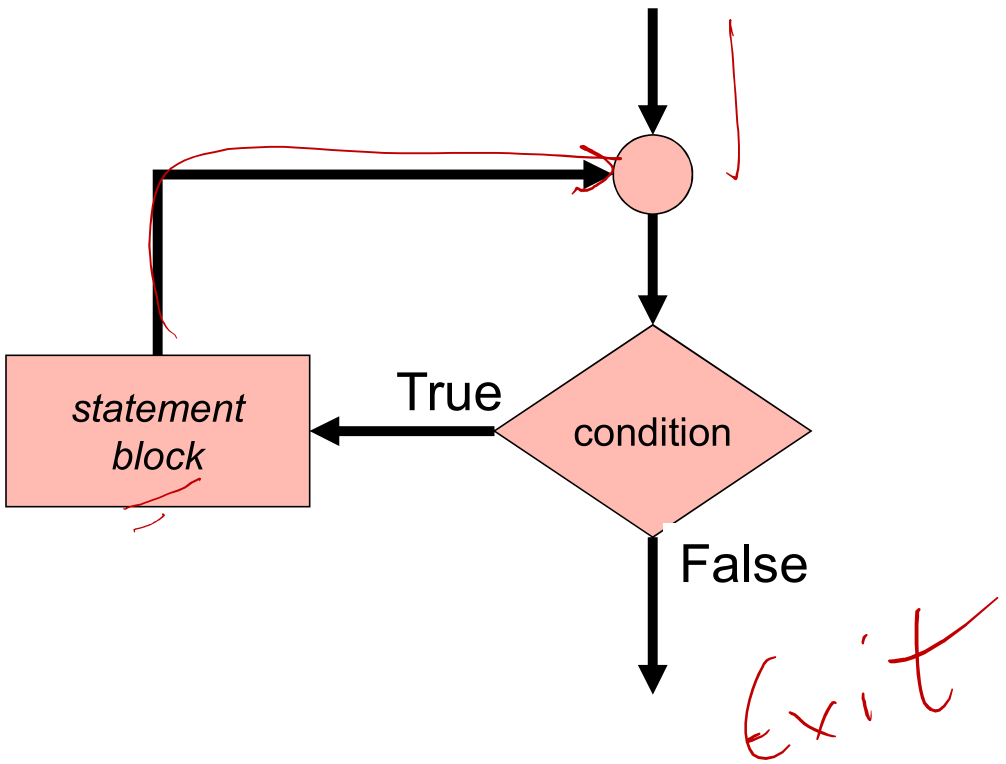
What’s a Loop?

- Functionality
  - A segment of code execute repeatedly when the loop’s Boolean expression is true
- Loops are useful for
    - Processing lists
    - Repeating steps, possibly with different parameters each time
    - Generating values, possibly with a different parameter each time.
- If statement allows us to skip code
- loops allows use to **repeat** code

- **Infinite loops** are loops that do no terminate
  - sometimes this is intentional, most of the time this is a programming error.

Loop

Loop through a set of statements as long as a condition is true



3 Kinds of Loops

- While loop
  - Used when we do NOT KNOW how many times we want the loop to run
- Do While loop
  - Used when we know we want the loop to run AT LEAST ONCE
- For loop
  - Used when we know EXACTLY how many times we want the loop to run

- NOTE: just like if statements, if you don’t use {} the loop naturally only associates with the first statement following it

{

}

While loop

- While loops are used if the number iterations that is needed to be executed is unknown

- While loops often require us to create a counter
  - This counter allows us to track how many times this loop has executed

- If we can't or don't want to use a counter, while loops are also appropriate
  - Ex. Stop when you find a value
  - Ex. Run till you reach the end of the string

While Loop Syntax

```
while(<boolean condition>)
{
    //loop body goes here
}
```

T

- Execution order

1. If the expression is false, run the statement directly after the loop body
2. Else run the loop body
3. Go back to step 1

while: Example

```
Var x = 1 ;
while ( x < 6000 ) {
        document.write ( x ) ;
        x = x + 1 ;
}
```

The condition enclosed in parentheses

Whileloop.html

while: Example

```
while ( tankIsFull == false ) {
        tank = tank + bucket ;
}

document.write ( “Tank is full now” ) ;
```

Pestering Users

- *While-loops* can be used to repeatedly ask user for input until the user gives a valid input
- For example

var usinput = ' ';

1580

```
~~var~~ userInput = +prompt("Enter a year greater than 1582");
while (userInput < 1582){
     userInput = +prompt("Are you senseless? I said less 
than 1582. Try again!");
}
```

Userinput = usinput + '1580

us = 1580

1580 Any

Solving Problems without knowing number of iterations

- While-loops can also be used to solve problems where
  - We know the stop condition
  - But don’t know how many iterations to run the loop

- Example problems:
  - Finding the first prime number after N
  - Checking if a word is contained inside a string

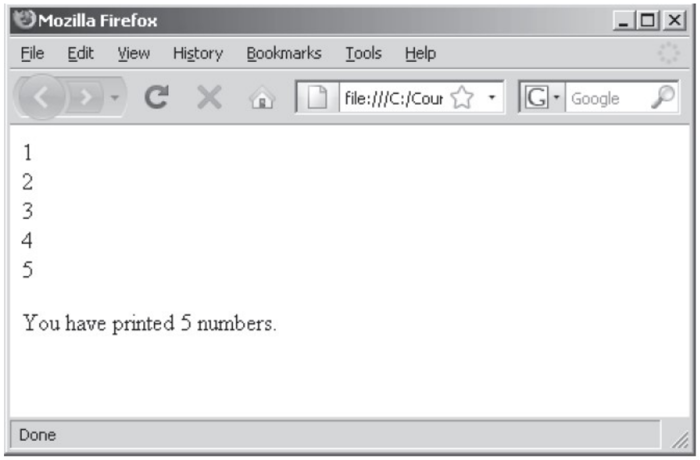
while Statements

- Counter
  - Variable incrementing or decrementing with each loop statement iteration

- Examples:
  - while statement using an increment operator
  - while statement using a decrement operator
  - while statement using the *= assignment operator

while Statements

```
var count = 1;
while (count <= 5) {
     document.write(count + "<br />");
     count++;
}
document.write("<p>You have printed 5 numbers.</p>");
```



```
1
2
3
4
5

You have printed 5 numbers.
```

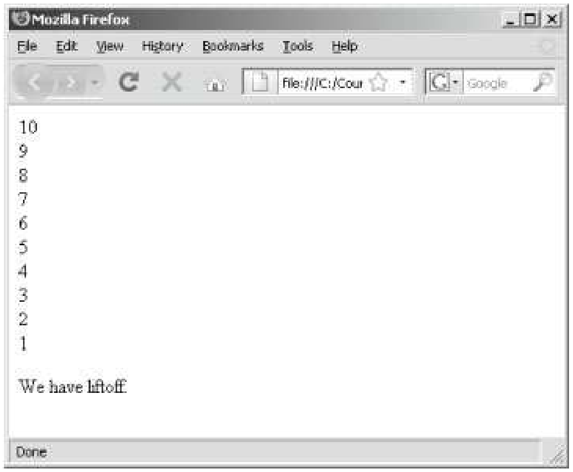
Output of a while statement using an increment operator

while Statements

9

```
var count = 10;
while (count > 0) {✓
      ~~document.~~write(count + "<br />");
      count--;
}
document.write("<p>We have liftoff.</p>");
```

10
9



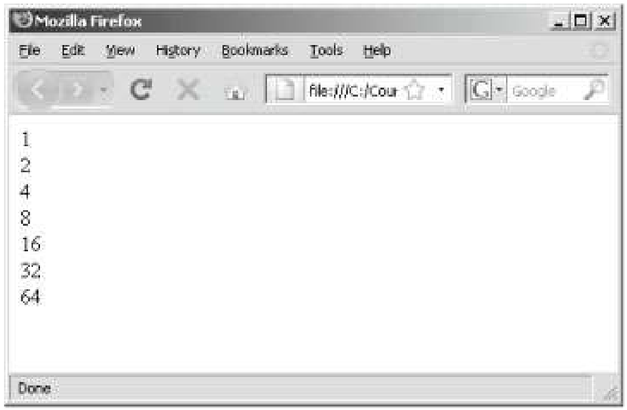
```
10
9
8
7
6
5
4
3
2
1

We have liftoff.
```

Output of a while statement using a decrement operator

while Statements

```
var count = 1;
while (count <= 100) {
     document.write(count + "<br />");
     count *= 2;
}
```

1
2
4
8



```
1
2
4
8
16
32
64
```

Output of a while statement using the *= assignment operator

while Statements

- Infinite loop
  - Loop statement that never ends
    - Conditional expression: never false
  - Example:

```
var count = 1;
while (count <= 10) {
     window.alert("The number is " + count + ".");
}
```

The Do While Loop

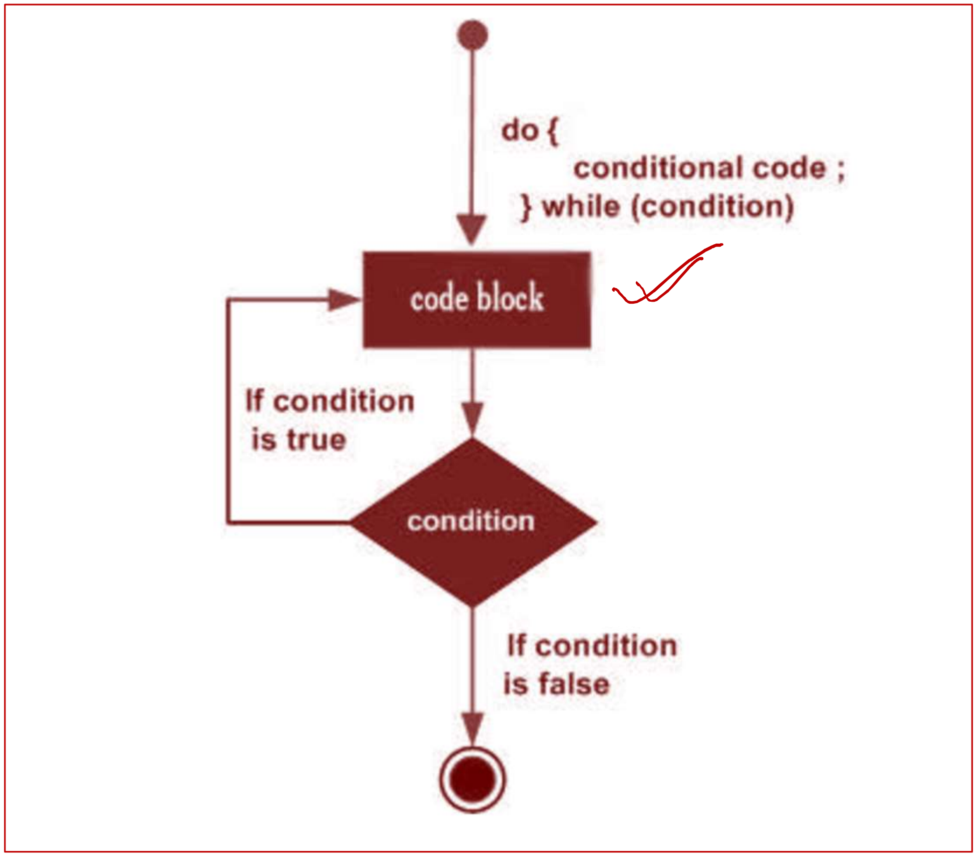
- The do while loop is usually used if you know you want the loop to execute AT LEAST ONCE
- Very similar to while loops, except while loops may execute zero times
- Do while loops execute **one** or **more** times

```
do{
   /// Body goes here
}while (<Boolean condition>);
```

Do while

```
do{
   /// code body goes here
}while (<Boolean condition>);
```

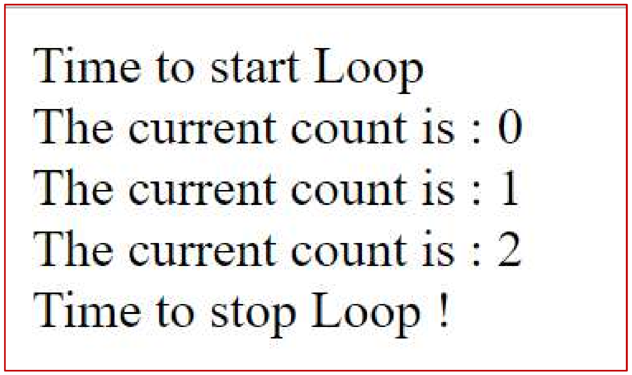
1. Execute the code body
2. If condition is not true, terminate the loop and execute the statement directly after the loop body
3. Else return to step 1



Do while

```
var count = 0;
document.write("Time to start Loop " + "<br />");
do{
document.write(" The current count is : " + count + "<br />");
count++;
}
while (count < 3);
document.write (" Time to stop Loop !");
```

3<3



```
Time to start Loop
The current count is : 0
The current count is : 1
The current count is : 2
Time to stop Loop !
```

break and continue

x = 9

```
var x=0;
while (x < 10) {
   x = x + 1;
   if (x == 3)
     continue;

   document.write("x = "+x);
   if (x == 5)
     break;
}
document.write("Loop done");
```

```
x = 1 ✓
x = 2 ✓
x = 4 ✓
x = 5 ✓
Loop done
```

Practice Lab (Use only while or do while loops)

Download Practice_Lab4_requiredfile.html. Write Javascript code for the following

1. Display Even Number : Use while loop to display the even numbers from 2 to 50. Each number should be displayed in a new line .

2. Display the sum of all entered numbers

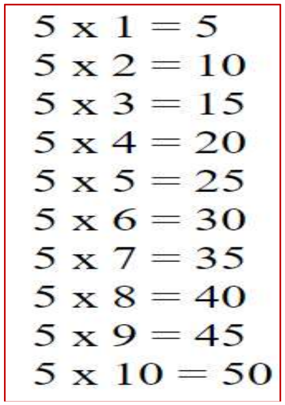
- Show first message (“Enter numbers, 0 to stop”). After each entered number, it should show second message (“Enter another, 0 to stop:”). After 0 , it should show the sum on the HTML page.

3. Display SumAndAverage

- Display the sum of 1, 2, 3, ..., to 30. Also compute and display the average. The output shall look like: The sum is 465 and the average is 15.5

4**. Display Table**

- Show the following message to the user (“What table you want:”). By default it should show 0. Then the table of entered number should be printed



5 x 1 = 5
5 x 2 = 10
5 x 3 = 15
5 x 4 = 20
5 x 5 = 25
5 x 6 = 30
5 x 7 = 35
5 x 8 = 40
5 x 9 = 45
5 x 10 = 50

for: Example

Initialization

condition

update

```
for ( x = 1 ; x < 6000 ; x = x + 1 ) {
    document.write ( x ) ;
}
```

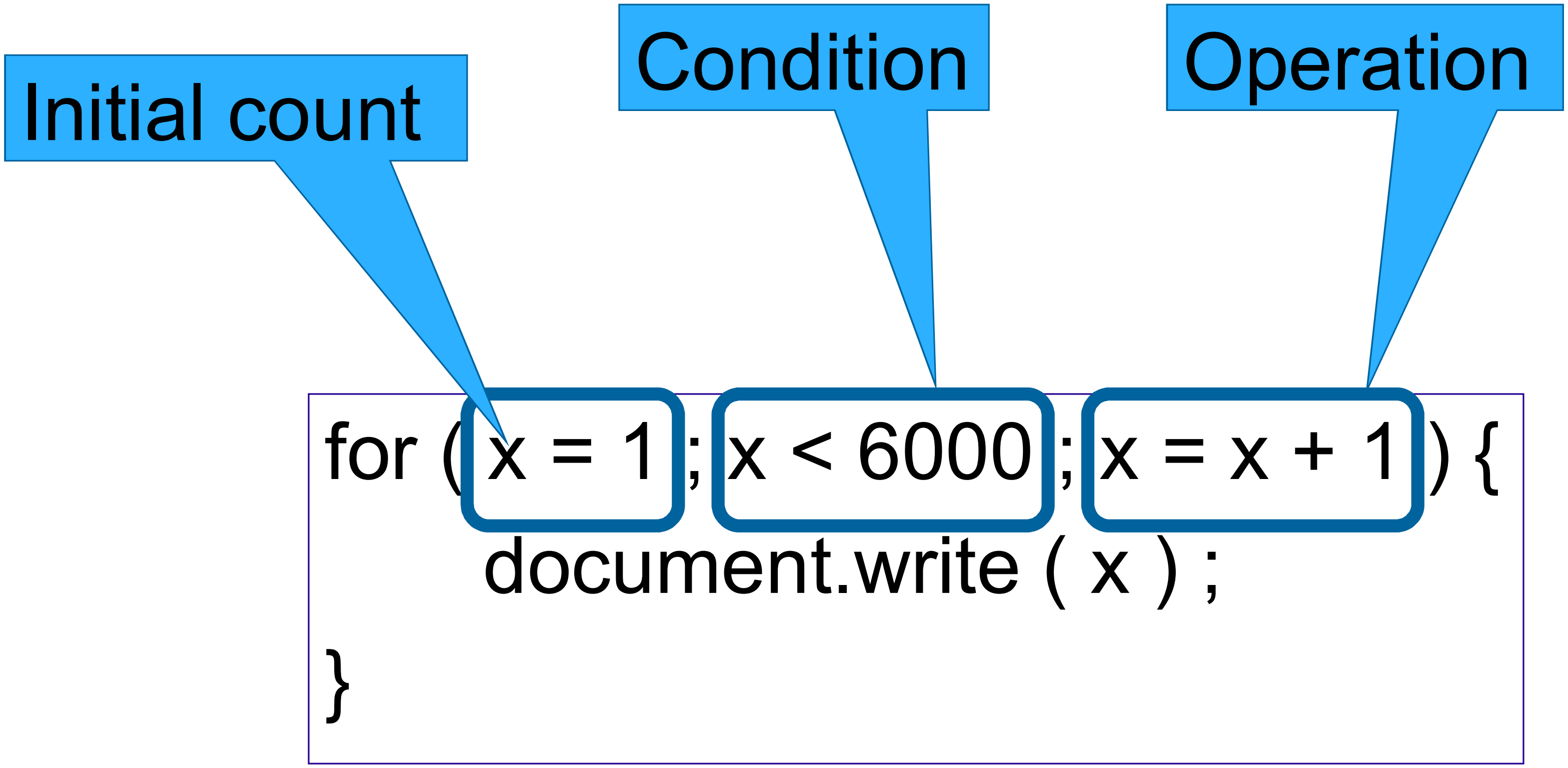
For loop (1)

1. The ‘for’ loop starts by initializing the *counter variable* (which in this case is x)

2. The initial value in this case is ‘1’, but can be any other positive or negative number as well

3. Next the ‘for’ loop checks the condition. If the condition evaluates to a ‘true’ value, the ‘for’ loop goes through the loop once

for: Example



Initial count

Condition

Operation

```
for ( x = 1 ; x < 6000 ; x = x + 1 ) {
        document.write ( x ) ;
}
```

For loop (2)

4. After reaching the end of that iteration, the ‘for’ loop goes to the top once again, performs the operation, checks the condition

5. If the condition evaluates to a ‘false’ value, the ‘for’ loop finishes looping

6. Otherwise, the ‘for’ loop goes through the loop once again

7. Repeat from step 4

The For Loop - Summarized

```
for (start; condition; update) {
  JavaScript Commands
}
```

- ***start*** is the starting value of the counter
- ***condition*** is a Boolean expression that must be true for the loop to continue
- ***update*** specifies how the counter changes in value each time the command block is executed

Demo

for: Example

```
for ( x = 10 ; x > 0 ; x = x - 1 ) {
   document.write ( x ) ;
}
```

How many iterations would this 'for' loop run for?

Forloop1.html

10?

for: Example

10 < 0

```
for ( x = 10 ; x < 0 ; x = x - 1 ) {
   document.write ( x ) ;
}
```

How many iterations would this ‘for’ loop run for?

None?

You Try

0 - 20

- Write a for loop that will iterate from 0 to 20. For each iteration, it will check if the current number is even or odd, and report that to the console (e.g. "2 is even")

```
var j;
for(j = 0; j <= 20; j++){
     if(j % 2 === 0)
          console.log(j + " is even");
     else console.log(j + " is odd");
}
```

for --?-- while

- When the exact number of iterations is known, use the ‘for’ loop
- When the number of iterations depend upon a condition being met, use the ‘while’ loop
- ‘for’ loops become especially useful when used in conjunction with arrays. We’ll find out about arrays next time, and we’ll probe their usefulness as part of ‘for’ loop structures

Reversing a String

- Write some JavaScript that reads a string from the user, then alerts the reverse of that string back to the user . Ex. If the user enters “watch me whip” you will alert “pihw em hctaw”

John = 4

```
var input = prompt("enter a string for me to reverse");
var current;
var reverse = '';
for(current = input.length - 1; current >= 0; current--)
     reverse += input.charAt(current);
alert(reverse);
```

Math.round

Nested For Loops

- You can use loops inside of loops
- When you use nested for loops think of it like a clock’s minute and hour hand

Nested For Loops

- Example: Use nested for loops to print a right triangle of asterisks to the console like this:

```
*    ✓
**     ✓
***   ✓
****     ✓
*****     ✓
```

- Where the height of the triangle is specified by the user (so in this case the user entered 5)

Solution

\n

```
var row, star;
var toPrint = '';
const MAX_ROWS = prompt('Enter the number of rows of stars to
print');

for (row = 1; row <= MAX_ROWS; row++)
{
    for (star = 1; star <= row; star++)
        toPrint += "*";
toPrint += '\n';
}
console.log(toPrint);
```

2

3 ≤ 2

3 < 2

row 1 *

— * *

r

toPrint = toPrint + '*'