

# Complete Beginner While loop-2

---



Print all the odd numbers from 1 to 10

|   | <u>initialisation</u>   |     | <u>condition checks</u> | <u>loop work</u> | <u>update</u> |
|---|-------------------------|-----|-------------------------|------------------|---------------|
| <pre> int n = 1; while(n &lt;= 10) {     System.out.println(n);     n = n + 1; } </pre> | n=1<br><u>printing.</u> | 1   | ✓                       | ✓                | ✓ (2)         |
|   | 1                       | 2   | ✓                       | ✓                | ✓ (3)         |
|   | 2                       | 3   | ✓                       | ✓                | ✓ (4)         |
|   | ...                     | ... |                         |                  |               |
|   | 9                       | 10  | ✓                       | ✓                | ✓ (10)        |
|   | 10                      | 11  | ✓                       | ✓                | ✓ (11)        |
|   |                         |     | False                   |                  |               |
|   | n=1<br><u>printing.</u> | 1   | ✓                       | ✓                | ✓ (2)         |
|   | 2                       | 3   | ✓                       | ✓                | ✓ (5)         |
|   | 4                       | 5   | ✓                       | ✓                | ✓ (7)         |
|   | 6                       | 7   | ✓                       | ✓                | ✓ (9)         |
|   | 8                       | 9   | ✓                       | ✓                | ✓ (11)        |
|   |                         | 11  | 11 <= 10                |                  |               |

|   | <u>initialisation</u>   |   | <u>condition checks</u> | <u>loop work</u> | <u>update</u> |
|---|-------------------------|---|-------------------------|------------------|---------------|
| <pre> int n = 1; while(n &lt;= 10) {     System.out.println(n);     n = n * 1; } </pre> | n=1<br><u>printing.</u> | 1 | ✓                       | ✓                | ✓ (1)         |
|   | 1                       | 1 | ✓                       | ✓                | ✓ (1)         |
|   | 1                       | 1 | ✓                       | ✓                | ✓ (1)         |
|   | 1                       | 1 | ...                     | ...              | ...           |
|   | 1                       | 1 | ...                     | ...              | ...           |
| <p><u>Infinite loop</u></p> <p>Loop will never stop</p>                                 |                         |   |                         |                  |               |

|   | <u>initialisation</u>   |    | <u>condition checks</u> | <u>loop work</u> | <u>update</u> |
|---|-------------------------|----|-------------------------|------------------|---------------|
| <pre> int n = 1; while(n &lt;= 10) {     System.out.println(n);     n = n * 2; } </pre> | n=1<br><u>printing.</u> | 1  | ✓                       | ✓                | ✓ (2)         |
|   | 2                       | 4  | ✓                       | ✓                | ✓ (4)         |
|   | 4                       | 8  | ✓                       | ✓                | ✓ (8)         |
|   | 8                       | 16 | ✓                       | ✓                | ✓ (16)        |
|   |                         |    | 16 <= 10<br>False       |                  |               |

Problem: Bowl all balls of an over

initialise the count = 1. and we have to deliver the ball for an over. , no. of balls in over = 6

we have to start from count 1 & move until count is not equal to 6.

$$\text{count} = \text{count} + 1$$

| <u>count</u> | <u>condition</u> ( <u>count &lt; 6</u> ) | deliver the bowl | value of count after update. |
|--------------|--|------------------|------------------------------|
| 1            | ✓<br>↓<br>make                           | ✓ → ✓            | ✓ (2)                        |
| 2            | ✓<br>same                                | ✓ → ✓            | ✓ (2)                        |
| 3            | ✓<br>changes in condition                | ✓ → ✓            | ✓ (4)                        |
| 4            | ✓  | ✓ → ✓            | ✓ (5)                        |
| 5            | ✓  | ✓ → ✓            | ✓ (6)                        |
| 6            | [count < 6]<br>false                     | 5 deliver        |                              |

count = 1;  
while (count <= 6) {

~~//~~ deliver a bowl

count = count + 1;

}

}

count = ~~1~~ ~~2~~ ~~3~~ ~~4~~ ~~5~~ ~~6~~  
7

✓  
✓  
✓  
✓  
✓  
✓

Print all perfect square from 1 to N

Ex  $N=100$ , 1, 4, 9, 16, 25, 36, 49, 64, 81, 100

```
i = 1
while(i <= 10) {
    SOP(i * i);
    i++;
}
```

|              | <u>i = 1</u><br><u>value of i</u> | <u>condition</u><br>$i \leq 10$ | <u>print(i * i)</u> | <u>update</u><br>$i++$ |
|--------------|-----------------------------------|---------------------------------|---------------------|------------------------|
| <u>Print</u> | 1                                 | ✓                               | ✓                   | ✓ (2)                  |
| 1            | 2                                 | ✓                               | ✓                   | ✓ (3)                  |
| 4            | 3                                 | ✓                               | ✓                   | ✓ (4)                  |
| 9            | 4                                 | ✓                               | ✓                   | ✓ (5)                  |
| 16           | 5                                 | ✓                               | ✓                   | ✓ (6)                  |
| 25           | 6                                 | ✓                               | ✓                   | ✓ (7)                  |
| 36           | 7                                 | ✓                               | ✓                   | ✓ (8)                  |
| 49           | 8                                 | ✓                               | ✓                   | ✓ (9)                  |
| 64           | 9                                 | ✓                               | ✓                   | ✓ (10)                 |
| 81           | 10                                | ✓                               | ✓                   | ✓ (11)                 |
| 100          | 11                                | <u>False</u>                    |                     |                        |

```
i = 1
while(i * i <= 10) {
    SOP(i * i);
    i++;
}
```

|   | <u>initialisation</u><br><u>i = 1</u> | <u>condition</u>                | <u>loop work</u> | <u>update</u> |
|---|---------------------------------------|---------------------------------|------------------|---------------|
|   | 1                                     | ✓                               | ✓                | ✓ (2)         |
| 1 | 2                                     | ✓                               | ✓                | ✓ (3)         |
| 4 | 3                                     | ✓                               | ✓                | ✓ (4)         |
| 9 | 4                                     | $4 * 4 \leq 10$<br><u>False</u> |                  |               |

```
i = 1
while(i <= 10) {
    P1 → i = i * i;
    P2 → SOP(i);
    i++;
}
```

|              | <u>initialisation</u><br><u>i = 1</u> | <u>condition</u>             | <u>loop work</u>             | <u>update</u>                  |
|--------------|---------------------------------------|------------------------------|------------------------------|--------------------------------|
| <u>Print</u> | 1                                     | ✓                            | <u>Part 1</u><br>$i = i * i$ | <u>Part 2</u><br><u>SOP(i)</u> |
| 1            | 2                                     | ✓                            | 1                            | 2                              |
| 4            | 5                                     | ✓                            | 4                            | 5                              |
| 25           | 26                                    | $26 \leq 10$<br><u>False</u> | 25                           | 26                             |

```

i = 0
while(i <= 10) {
    SOP(i);
    i = i * i;
}

```

Syntactically

Loop will never stop.

| initialisation | condition | loop work | update. |
|----------------|-----------|-----------|---------|
| i = 0          | ✓         | ✓         | ✓ (0)   |
| Printing - { 0 | ✓         | ✓         | ✓ (0)   |
| 0              |           |           |         |
| 0              |           |           |         |
| 0              |           |           |         |
| 0              |           |           |         |
| ...            |           |           |         |


Print all perfect square from 1 to N

Ex N = 100, 1, 4, 9, 16, 25, 36, 49, 64, 81, 100

| N = 100              | initialisation | condition                    | loop work                  | update.          |
|----------------------|----------------|------------------------------|----------------------------|------------------|
| <u>i = 1</u>         | <u>i = 1</u>   | <u>i * i &lt;= (N = 100)</u> | <u>SOP(i * i)</u>          | <u>i = i + 1</u> |
| int i = 1;           |                | value of i                   |                            | ✓ (2)            |
| while (i * i <= N) { |                | 1 ✓                          | ✓                          | ✓ (2)            |
| SOP(i * i);          |                | 2 ✓                          | ✓                          | ✓ (4)            |
| i = i + 1; Printing. |                | 3 ✓                          | ✓                          | ✓ (5)            |
| 3                    | 1 -            | 4 ✓                          | ✓                          | ✓ (6)            |
| 2 * 2 = 4 -          | 2 -            | 5 ✓                          | ✓                          | ✓ (7)            |
| 3 * 3 = 9 -          | 3 -            | 6 ✓                          | ✓                          | ✓ (8)            |
| 81                   | 4 * 4 = 16 -   | 7 ✓                          | ✓                          | ✓ (9)            |
| 100 →                | 5 * 5 = 25     | 8 ✓                          | ✓                          | ✓ (10)           |
|                      | 6 * 6 = 36     | 9 ✓                          | ✓                          | ✓ (11)           |
|                      | 49             | 10 ✓                         |                            |                  |
|                      | 64             | 11                           | 11 * 11 > 100<br>121 false |                  |
|                      |                |                              | Start From                 |                  |
|                      |                |                              | 10: 40 → Break             |                  |

Problem: Given a number, Print the last digit of a number?

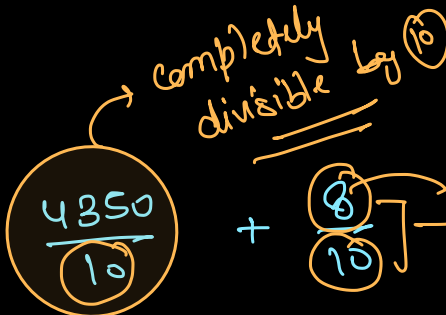
$$N = 4358$$

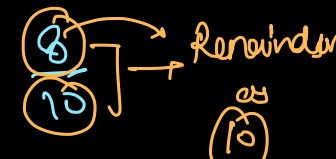


$\%$   $\rightarrow$  modulo operator  $\rightarrow$  gives remainder.

$$4358 = 4350 + 8$$

$$\frac{4358}{10} = \frac{4350 + 8}{10} = \frac{4350}{10} + \frac{8}{10}$$






last digit =  $N \% 10$

$$l.d = 4358 \% 10 = 8$$

Repeat.

$l.d = N \% 10$   
 last digit


- ① How is it working?   
 ② why only 10?

$$N = 4358$$

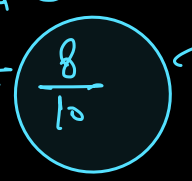
$$l.d = (4358) \% 10 = (\underline{4350 + 8}) \% 10$$

$$= (\underline{4350 \% 10}) + (\underline{8 \% 10})$$


$$l.d = 0 + 8 = 8$$



$$\frac{4358}{10} = \frac{4350}{10} + \frac{8}{10}$$



$$\begin{array}{r}
 435 \\
 10 \overline{) 4358} \\
 \underline{40} \phantom{0} \phantom{0} \phantom{0} \\
 35 \phantom{0} \phantom{0} \phantom{0} \\
 \underline{30} \phantom{0} \phantom{0} \phantom{0} \\
 58 \phantom{0} \\
 \underline{50} \phantom{0} \\
 8
 \end{array}$$



$last\ digit = N \% 10$

Problem: Given a Number, print all the digits of a number.

N = 6 3 4 1 → print → 1  
 ← 4  
 starting direction  
 from least significant digit to most significant digit

N = 6 7 4 3  
 3  
 4  
 7  
 6

Sol: SOP (n % 10) → 3  
 SOP (n % 100) → 43  
 SOP (n % 1000) → 743  
 SOP (n % 10000) → 6743

Some where.

N = 0  
 '0' is also a digit

if (N == 0) {  
 SOP(0);  
 } else {  
 approach of print all digit  
 }

Ques: no. of times we have generate

number = 7436

l.d = number % 10 ⇒ 6

number = number / 10

10 ) 7436  
 70  
 43  
 40  
 36  
 30  
 6 ← Remainder

| number | l.d. | num = num / 10 |
|--------|------|----------------|
| 7436   | 6    | 743            |
| 743    | 3    | 74             |
| 74     | 4    | 7              |
| 7      | 7    | 0              |
| 0      | 0    |                |

0

Break point

No. of digits →

$N = 34562$

Count = 0

```

N = 34562
count = 0;
while (N >= 0) {
    N = N / 10;
    count++;
}
SOP(count);

```

loop will never end

| Count | N     | condition | work | Update |
|-------|-------|-----------|------|--------|
| 0     | 34562 | ✓         | 3456 | 1      |
| 1     | 3456  | ✓         | 345  | 2      |
| 2     | 345   | ✓         | 34   | 3      |
| 3     | 34    | ✓         | 3    | 4      |
| 4     | 3     | ✓         | 0    | 5      |
| 5     | 0     | ✓         | 0    | 6      |
| 6     | 0     | ✓         | 0    | 7      |
| 7     | 0     | ...       | ...  | ...    |

NOTE: If we want to iterate on a number, we have made a condition  $n > 0$  ] → moving condition

\*

$4 \wedge 3 = 4 * 4 * 4 =$   
 1 ← After 1st  
 4 ← " 2nd "  
 16 ← " 3rd "  
 64

prod = 1 → 6 time  
 iterate 2 time  
 $prod = prod * 4$   
 $= 4 * 4 * 4$



Print table -

Print the table for N.

N=4.

```

int i=1;
while(i<=10) {
    SOP(N+"*"+i+"="+"+(N*i));
    i++;
}
    
```

boolean expression

```

if (1) {
    // if statement
} else {
}
    
```

1 2 4 8

16 → n After value.

4 \* 1 = 4  
 4 \* 2 = 8  
 4 \* 3 = 12  
 4 \* 4 = 16  
 4 \* 5 = 20  
 4 \* 6 = 24  
 4 \* 7 =  
 4 \* 8 =  
 4 \* 9 =  
 4 \* 10 = 40

4 \* 1 = 4  
 4 \* 2 = 8  
 4 \* 3 = 12  
 4 \* 4 = 16  
 ,  
 !

4 \* 10 = 40.

initialisation

n=1

condition

n<=10

work

SOP(n)

output

n = <sup>1\*2</sup><sub>2\*2</sub> n\*2

✓ (2)

✓ (4)

✓ (8)

✓ (16)

n  
1

2

4

8

16

✓

✓

✓

✓

False.

compu