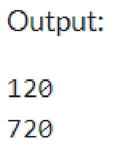
SCALA PROGRAMMING

TAIL RECURSION

Code-1:

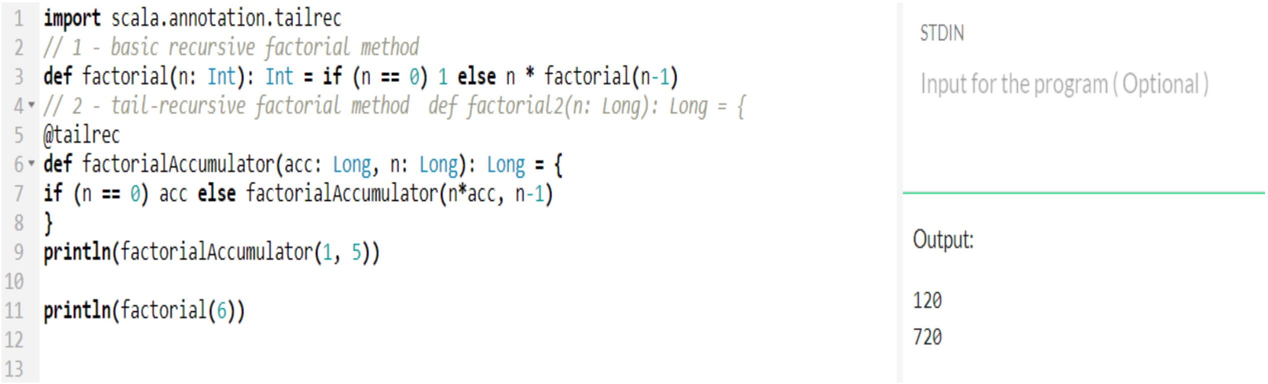
```
import scala.annotation.tailrec
// 1 - basic recursive factorial method
def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)
// 2 - tail-recursive factorial method
def factorial2(n: Long): Long = {
@tailrec
def factorialAccumulator(acc: Long, n: Long): Long = {
if (n == 0) acc else factorialAccumulator(n*acc, n-1)
}
println(factorialAccumulator(1, 5))
println(factorial(6))
```

Output-1:



```
Output:

120
720
```

Screenshot-1:



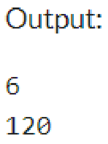
```
import scala.annotation.tailrec
// 1 - basic recursive factorial method
def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)
// 2 - tail-recursive factorial method  def factorial2(n: Long): Long = {
@tailrec
def factorialAccumulator(acc: Long, n: Long): Long = {
if (n == 0) acc else factorialAccumulator(n*acc, n-1)
}
println(factorialAccumulator(1, 5))

println(factorial(6))
```

```
Output:

120
720
```

Code-2:

```
import scala.annotation.tailrec
// 1 - basic recursive factorial method
def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)
// 2 - tail-recursive factorial method
def factorial2(n: Long): Unit = {
@tailrec
def factorialAccumulator(acc: Long, n: Long): Long = {
if (n == 0) acc else factorialAccumulator(n*acc, n-1)
}
println(factorialAccumulator(1, n))
}
factorial2(3)
println(factorial(5))
```

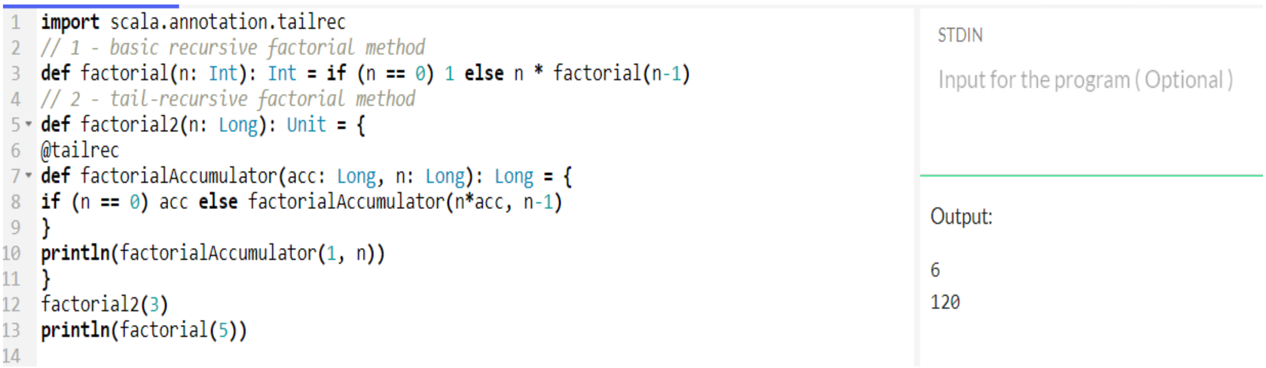
Output-2:



Output:

```
6
120
```

Screenshot-2:



```
import scala.annotation.tailrec
// 1 - basic recursive factorial method
def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)
// 2 - tail-recursive factorial method
def factorial2(n: Long): Unit = {
@tailrec
def factorialAccumulator(acc: Long, n: Long): Long = {
if (n == 0) acc else factorialAccumulator(n*acc, n-1)
}
println(factorialAccumulator(1, n))
}
factorial2(3)
println(factorial(5))
```

Output:

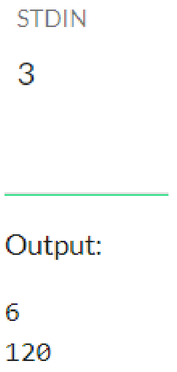
```
6
120
```

Taking input from the user:

Code-3:

```
import scala.annotation.tailrec

import scala.io.StdIn

// 1 - basic recursive factorial method

def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)

// 2 - tail-recursive factorial method

def factorial2(n: Long): Unit = {

@tailrec

def factorialAccumulator(acc: Long, n: Long): Long = {

if (n == 0) acc else factorialAccumulator(n*acc, n-1)

}

println(factorialAccumulator(1, n))

}


var x = StdIn.readInt()

factorial2(x)

println(factorial(5))
```

Output-3:



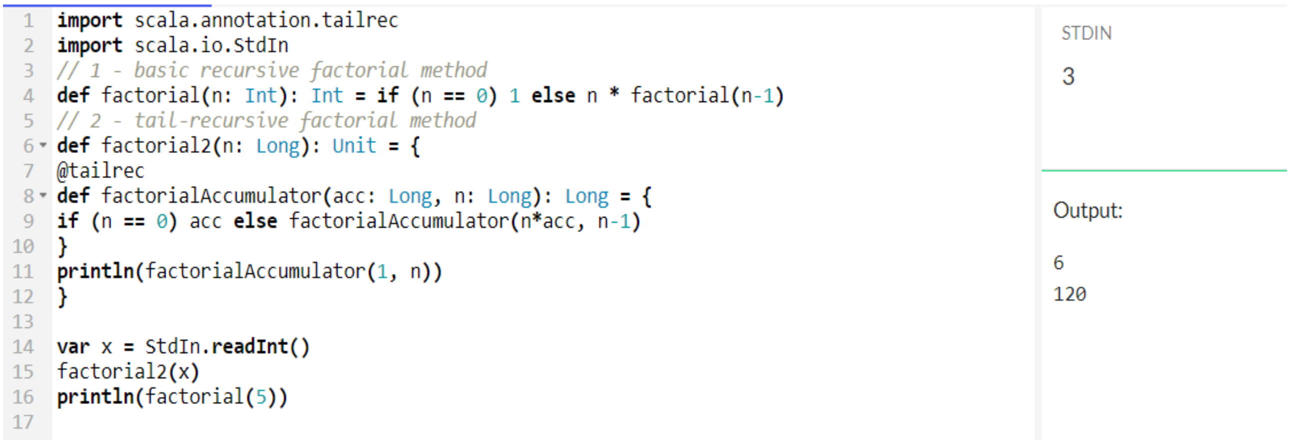
STDIN

```
3
```

Output:

```
6
120
```

Screenshot-3:



```scala
import scala.annotation.tailrec
import scala.io.StdIn
// 1 - basic recursive factorial method
def factorial(n: Int): Int = if (n == 0) 1 else n * factorial(n-1)
// 2 - tail-recursive factorial method
def factorial2(n: Long): Unit = {
@tailrec
def factorialAccumulator(acc: Long, n: Long): Long = {
if (n == 0) acc else factorialAccumulator(n*acc, n-1)
}
println(factorialAccumulator(1, n))
}

var x = StdIn.readInt()
factorial2(x)
println(factorial(5))

```

STDIN

```
3
```

Output:

```
6
120
```