CS 161
Intro to CS I

Recursion
Example: Factorial

• Definition

0! = 1;
n! = n \times (n-1) \times \ldots \times (n-(n-1)) \times 1 = n \times (n-1)! ; \ n > 0
Iterative Factorial

\[
\text{factorial}(0) = 1; \\
\text{factorial}(n) = n \times (n-1) \times (n-2) \times \ldots \times 1;
\]

```c
long factorial(int n) {
    long fact;
    if(n==0)
        fact=1;
    else
        for(fact=n; n > 1; n--)
            fact=fact*(n-1);
    return fact;
}
```
Recursive Factorial

factorial(0) = 1;
factorial(n) = n * factorial(n-1);

long factorial(int n) {
    if (n == 0) // Base case
        return 1;
    else
        return n * factorial(n - 1); // Recursive call
}
Computing Factorial Iteratively

\[
\text{factorial}(4)
\]

\[
\text{factorial}(0) = 1;
\text{factorial}(n) = n \times (n-1) \times \ldots \times 2 \times 1;
\]
Computing Factorial Iteratively

factorial(4) = 4 * 3

factorial(0) = 1;
factorial(n) = n*(n-1)*...*2*1;
Computing Factorial Iteratively

factorial(4) = 4 * 3
= 12 * 2

factorial(0) = 1;

factorial(n) = n*(n-1)*...*2*1;
Computing Factorial Iteratively

factorial(4) = 4 * 3
    = 12 * 2
    = 24 * 1

factorial(0) = 1;
factorial(n) = n*(n-1)*...*2*1;
Computing Factorial Iteratively

factorial(4) = 4 * 3
    = 12 * 2
    = 24 * 1
    = 24

factorial(0) = 1;
factorial(n) = n * (n-1) * ... * 2 * 1;
Computing Factorial Recursively

\[
\text{factorial}(4)
\]

\[
\text{factorial}(0) = 1;
\]

\[
\text{factorial}(n) = n \times \text{factorial}(n-1);
\]

---

OSU Oregon State University
Computing Factorial Recursively

factorial(0) = 1;
factorial(n) = n * factorial(n-1);

factorial(4) = 4 * factorial(3)
Computing Factorial Recursively

\[\text{factorial}(4) = 4 \times \text{factorial}(3)\]
\[= 4 \times (3 \times \text{factorial}(2))\]

\[\text{factorial}(0) = 1;\]
\[\text{factorial}(n) = n \times \text{factorial}(n-1);\]
Computing Factorial Recursively

factorial(4) = 4 * factorial(3)
    = 4 * ( 3 * factorial(2))
    = 4 * ( 3 * (2 * factorial(1)))
Computing Factorial Recursively

factorial(4) = 4 * factorial(3)
= 4 * ( 3 * factorial(2))
= 4 * ( 3 * (2 * factorial(1)))
= 4 * ( 3 * ( 2 * (1 * factorial(0)))))

factorial(0) = 1;
factorial(n) = n*factorial(n-1);
Computing Factorial Recursively

\[
\text{factorial}(4) = 4 \times \text{factorial}(3) \\
= 4 \times (3 \times \text{factorial}(2)) \\
= 4 \times (3 \times (2 \times \text{factorial}(1))) \\
= 4 \times (3 \times (2 \times (1 \times \text{factorial}(0)))) \\
= 4 \times (3 \times (2 \times (1 \times 1)))
\]

factorial(0) = 1;
factorial(n) = n \times \text{factorial}(n-1);
Computing Factorial Recursively

\[
\text{factorial}(4) = 4 \times \text{factorial}(3) \\
= 4 \times (3 \times \text{factorial}(2)) \\
= 4 \times (3 \times (2 \times \text{factorial}(1))) \\
= 4 \times (3 \times (2 \times (1 \times \text{factorial}(0)))) \\
= 4 \times (3 \times (2 \times (1 \times 1))) \\
= 4 \times (3 \times (2 \times 1))
\]
Computing Factorial Recursively

\[
\begin{align*}
\text{factorial}(4) &= 4 \times \text{factorial}(3) \\
&= 4 \times (3 \times \text{factorial}(2)) \\
&= 4 \times (3 \times (2 \times \text{factorial}(1))) \\
&= 4 \times (3 \times (2 \times (1 \times \text{factorial}(0)))) \\
&= 4 \times (3 \times (2 \times (1 \times 1))) \\
&= 4 \times (3 \times (2 \times 1)) \\
&= 4 \times (3 \times 2)
\end{align*}
\]
Computing Factorial Recursively

\[
\text{factorial}(4) = 4 \times \text{factorial}(3) \\
= 4 \times (3 \times \text{factorial}(2)) \\
= 4 \times (3 \times (2 \times \text{factorial}(1))) \\
= 4 \times (3 \times (2 \times (1 \times \text{factorial}(0)))) \\
= 4 \times (3 \times (2 \times (1 \times 1))) \\
= 4 \times (3 \times (2 \times 1)) \\
= 4 \times (3 \times 2) \\
= 4 \times 6
\]

factorial(0) = 1;
factorial(n) = n \times \text{factorial}(n-1);
Computing Factorial Recursively

\[ \text{factorial}(0) = 1; \]
\[ \text{factorial}(n) = n \times \text{factorial}(n-1); \]

\[ \text{factorial}(4) = 4 \times \text{factorial}(3) \]
\[ = 4 \times (3 \times \text{factorial}(2)) \]
\[ = 4 \times (3 \times (2 \times \text{factorial}(1))) \]
\[ = 4 \times (3 \times (2 \times (1 \times \text{factorial}(0)))) \]
\[ = 4 \times (3 \times (2 \times (1 \times 1))) \]
\[ = 4 \times (3 \times (2 \times 1)) \]
\[ = 4 \times (3 \times 2) \]
\[ = 4 \times 6 \]
\[ = 24 \]
Differences

• Pros
  – Readability

• Cons
  – Efficiency
  – Memory
Recursive Factorial

factorial(4)

Executes factorial(4)

Step 9: return 24
Step 0: executes factorial(4)
Step 1: executes factorial(3)
Step 2: executes factorial(2)
Step 3: executes factorial(1)
Step 5: return 1
Step 6: return 1
Step 7: return 2
Step 8: return 6
Step 4: executes factorial(0)
Recursive Factorial

factorial(4)

return 4 * factorial(3)

Executes factorial(3)

Step 0: executes factorial(4)

Step 1: executes factorial(3)

Step 2: executes factorial(2)

Step 3: executes factorial(1)

Step 4: executes factorial(0)

Step 5: return 1

Step 6: return 1

Step 7: return 2

Step 8: return 6

Step 9: return 24

Stack

Space Required for factorial(4)

Main method
Recursive Factorial

factorial(4)

Step 0: executes factorial(4)

return 4 * factorial(3)

Step 1: executes factorial(3)

return 3 * factorial(2)

Step 2: executes factorial(2)

Step 3: executes factorial(1)

Step 5: return 1

Step 6: return 1

Step 7: return 2

Step 8: return 6

Executes factorial(2)
Recursive Factorial

factorial(4)

Step 0: executes factorial(4)

return 4 * factorial(3)

Step 1: executes factorial(3)

return 3 * factorial(2)

Step 2: executes factorial(2)

return 2 * factorial(1)

Step 3: executes factorial(1)

Step 4: executes factorial(0)

Step 5: return 1

Step 6: return 1

Step 7: return 2

Step 8: return 6

Step 9: return 24

Executes factorial(1)

Space Required for factorial(4)
Space Required for factorial(3)
Space Required for factorial(2)
Space Required for factorial(4)
Main method
Recursive Factorial

factorial(4)

return 4 * factorial(3)

Step 0: executes factorial(4)

return 3 * factorial(2)

Step 1: executes factorial(3)

return 2 * factorial(1)

Step 2: executes factorial(2)

return 1 * factorial(0)

Step 3: executes factorial(1)

Step 5: return 1

Step 6: return 1

Step 7: return 2

Step 8: return 6

Executes factorial(0)

Stack

Space Required for factorial(1)
Space Required for factorial(2)
Space Required for factorial(3)
Space Required for factorial(4)
Main method
Recursive Factorial

factorial(4)

Step 0: executes factorial(4)
return 4 * factorial(3)

Step 1: executes factorial(3)
return 3 * factorial(2)

Step 2: executes factorial(2)
return 2 * factorial(1)

Step 3: executes factorial(1)
return 1 * factorial(0)

Step 4: executes factorial(0)
return 1

returns 1

Stack
- Space Required for factorial(0)
- Space Required for factorial(1)
- Space Required for factorial(2)
- Space Required for factorial(3)
- Main method

OSU Oregon State University
Recursive Factorial

\[
\text{factorial}(4) \\
\text{return } 4 \times \text{factorial}(3) \\
\text{Step 1: executes factorial}(3) \\
\text{return } 3 \times \text{factorial}(2) \\
\text{Step 2: executes factorial}(2) \\
\text{return } 2 \times \text{factorial}(1) \\
\text{Step 3: executes factorial}(1) \\
\text{return } 1 \times \text{factorial}(0) \\
\text{Step 4: executes factorial}(0) \\
\text{Stack} \\
\text{Main method} \\
\text{Space Required for factorial(4)} \\
\text{Space Required for factorial(3)} \\
\text{Space Required for factorial(2)} \\
\text{Space Required for factorial(1)} \\
\text{Space Required for factorial(0)} \\
\text{Space Required for factorial(1)} \\
\text{Space Required for factorial(2)} \\
\text{Space Required for factorial(3)} \\
\text{Space Required for factorial(4)} \\
\text{Space Required for factorial(0)} \\
\text{Main method}
\]
Recursive Factorial

Step 0: executes factorial(4)

return 4 * factorial(3)

Step 1: executes factorial(3)

return 3 * factorial(2)

Step 2: executes factorial(2)

return 2 * factorial(1)

Step 3: executes factorial(1)

return 1 * factorial(0)

Step 5: return 1

Step 6: return 1

Step 4: executes factorial(0)

returns factorial(1)
Recursive Factorial

factorial(4)  
return 4 * factorial(3)  
Step 0: executes factorial(4)  
Step 1: executes factorial(3)  
return 3 * factorial(2)  
Step 2: executes factorial(2)  
Step 3: executes factorial(1)  
return 2 * factorial(0)  
Step 4: executes factorial(0)  
Step 5: return 1  
return 1  
Step 6: return 1  
Step 7: return 2  
returns factorial(2)

Space Required for factorial(4)  
Space Required for factorial(3)  
Stack  
Main method
Recursive Factorial

factorial(4)

return 4 * factorial(3)

return 3 * factorial(2)

return 2 * factorial(1)

return 1 * factorial(0)

Step 0: executes factorial(4)

Step 1: executes factorial(3)

Step 2: executes factorial(2)

Step 3: executes factorial(1)

Step 5: return 1

Step 6: return 1

Step 7: return 2

Step 8: return 6

returns factorial(3)

OSU Oregon State University
Step 9: return 24
Step 8: return 6
Step 7: return 2
Step 6: return 1
Step 5: return 1
Step 4: executes factorial(0)
Step 3: executes factorial(1)
Step 2: executes factorial(2)
Step 1: executes factorial(3)
Step 0: executes factorial(4)

returns factorial(4)