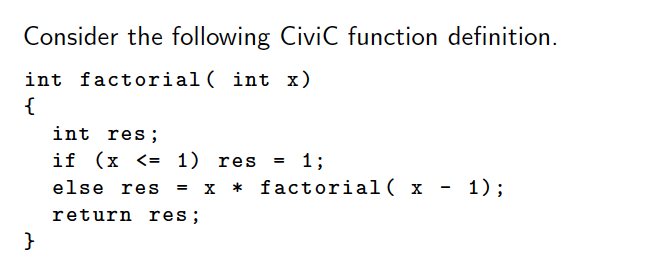
**Assignment 6**

Andrea & Aynel



**Assignment 21: Code generation**

|  |  |  |
| --- | --- | --- |
| [a] | [b] | [c] & [d] |
| **factorial**:  esr 1  istore 0  iload 1 | // go into factorial function  // init variable res­  // load variable x­ ­­­­ | bytes: 2  bytes: 2  bytes: 2 |
| iloadc\_1 | // load constant 1 | bytes: 1 |
| ile | // if x <= 1 | bytes: 1 |
| branch\_f **L1**  iloadc\_1 | // go into if ^ == true  // load constant 1 | bytes: 3 (offset: 9)  bytes: 1 |
| istore 1 | // store res = 1; | bytes: 2 |
| jump **L2** | // skip else block | bytes: 3 (offset: 17) |
| **L1:**  isrg  iload 0  iloadc\_0  isub  jsr 1  iload 0  imul  istore 1 | // go into if x <= 1 == false  // start functiecall factorial  // load variable x  // load constant 1  // sub x – 1  // factorial(x - 1)  // load variable x  // multiply x \* factorial ( x – 1)  // store result in res | bytes: 1  bytes: 2  bytes: 1  bytes: 1  bytes: 4  bytes: 2  bytes: 1  bytes: 2 |
| **L2:**  ireturn | // return res | bytes: 1 |

**Assignment 22: Compilation Schemes Revisited**

Original for-loop:

| for ( int i = lower, upper) { |

C | *Body* |

| } |

| *Rest* |

Replacement:

| int i = lower; |  
 | while (i < upper) { |

-> | C |*Body* | |

| i += 1; |

| } |

| C |*Rest* | |