Fullname: Nguyen Huu Hoang Hai Anh

Student ID: 20226010

**Lab 06. CODE GENERATION**

*Requirement: A report shows the results of implementation the examples with explanation about the instructions of the target program is generated from which statement of the source program.*

1. **Example 1**

***Result***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Explanation***

|  |  |  |
| --- | --- | --- |
| **Source Statement** | **Bytecode Address** | **Generated Bytecode Instruction** |
| Program Example1; | 0 | **J 1** → Jump over stack‐frame setup |
| *implicit stack‐frame setup* | 1 | **INT 4** → Reserve 4 words for activation record |
| *(empty Begin…End. block)* | — | *(no body statements → no instructions)* |
| End of program (End.) | 2 | **HL** → Halt program execution |

1. **Example 2**

***Result***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Explanation***

|  |  |  |
| --- | --- | --- |
| **Source Statement** | **Bytecode Address** | **Generated Bytecode Instructions** |
| Program Example2; | 0 | **J 1** → Jump over stack frame setup to main code |
| *Implicit stack frame setup for the program* | 1 | **INT 5** → Reserve 5 memory slots (activation record) |
| n := READI; | 2-4 | **LA 0,4** → Load Address of variable n (scope 0, offset 4) **RI** → Read Integer from input **ST** → Store value into n |
| If (n - (n/2) \* 2) = 0 | 5-14 | **LV 0,4** → Load Value of n **LV 0,4** → Load Value of n again (for division) **LC 2** → Load Constant 2 **DV** → Divide (n/2) **LC 2** → Load Constant 2 **ML** → Multiply ((n/2)\*2) **SB** → Subtract (n - result) **LC 0** → Load Constant 0 (for comparison) **EQ** → Check Equality (result == 0?) **FJ 18** → False Jump to address 18 if condition fails |
| Call WRITEC('E') (Then branch) | 15-16 | **LC 69** → Load Constant 69 (‘E’) WRC → Write Character ('E') |
| *jump over Else branch after printing 'E'* | 17 | **J 20** → Jump to address 20 (skip Else branch) |
| Call WRITEC('O') (Else branch) | 18 – 19 | **LC 79** → Load Constant 79 ('O') **WRC** →Write Character ('O') |
| Call WRITELN; | 20 | **WLN** → Write Line (newline) |
| End of program (End.) | 21 | **HL** → Halt program execution |

1. **Example 3**

***Result***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Explanation***

|  |  |  |
| --- | --- | --- |
| **Source Statement** | **Bytecode Address** | **Generated Bytecode Instructions** |
| Program Example3; | 0 | **J 1** → Jump over stack frame setup to main code |
| |  | | --- | | *implicit stack‐frame setup for the program* |  |  | | --- | |  | | 1 | **INT 7** → Reserve 7 memory slots (activation record) |
| n := READI; | 2-4 | **LA 0,4** → Load Address of n (scope 0, offset 4) **RI** → Read Integer from input **ST** → Store value into n |
| S := 0; | 5-7 | **LA 0,6** → Load Address of S (scope 0, offset 6) **LC 0** → Load Constant 0 **ST** → Store 0 into S |
| i := 1; | 8-10 | **LA 0,5** → Load Address of i (scope 0, offset 5) **LC 1** → Load Constant 1 **ST** → Store 1 into i |
| While i <= n Do | 11 – 14 | *(loop header: condition check and branch)*  **LV 0,5** → Load Value of i **LV 0,4** → Load Value of n **LE** → Less or Equal comparison (i <= n) **FJ 26** →False Jump to 26 if condition fails (exit loop) |
| Begin … End; (loop body) |  | *(body of the while loop)* |
| S := S + i; | 15-19 | **LA 0,6** → Load Address of S (for result storage) **LV 0,6** → Load Value of S **LV 0,5** → Load Value of i **AD** → Add S + i **ST** → Store result into S |
| i := i + 1; | 20-24 | **LA 0,5** → Load Address of i (for result storage) **LV 0,5** → Load Value of i **LC 1** → Load Constant 1 **AD** → Add i + 1 **ST**  → Store result into i |
| *Jump back to loop hader* | 25 | **J 11** → Jump back to loop condition check |
| Call WRITEI(S); | 26-27 | **LV 0,6** → Load Value of S **WRI** → Write Integer (S) |
| Call WRITELN; | 28 | **WLN** → Write Line (newline) |
| End of program (End.) | 29 | **HL** → Halt program execution |

1. **Example 4**

***Result***

**A screenshot of a computer

AI-generated content may be incorrect.**

***Explanation***

|  |  |  |
| --- | --- | --- |
| **Source Statement** | **Bytecode Address** | **Generated Bytecode Instructions** |
| Program Example4; | 0 | **J 1** → Jump over stack‐frame setup |
| *(implicit stack‐frame setup for the program)* | 1 | **INT 7** → Reserve 7 memory slots (activation record) |
| n := READI; | 2 – 4 | **LA 0,4** →Load Address of n (scope 0, offset 4)  **RI** →Read Integer from input  **ST** → Store value into n |
| S := 0; | 5 – 7 | **LA 0,6** →Load Address of S (scope 0, offset 6)  **LC 0** →Load Constant 0  **ST** → Store 0 into S |
| i := 1; | 8 – 10 | **LA 0,5** → Load Address of i (scope 0, offset 5)  **LC 1** →Load Constant 1  **ST** → Store 1 into i |
| For i := 1 To n Do S := S + i; |  | *(for‐loop header, body, and increment)* |
| Loop condition i ≤ n | 11 – 14 | **LV 0,5** → Load Value of i  **LV 0,4** → Load Value of n  **LE** → Less or Equal comparison (i <= n)  **FJ 26** → False Jump to 26 if condition fails (exit loop) |
| Loop body S := S + i; | 15 – 19 | **LA 0,6** → Load Address of S (for result storage)  **LV 0,6** →Load Value of S  **LV 0,5** → Load Value of i  **AD** →Add S + i  **ST** → Store result into S |
| Loop increment i := i + 1 | 20 – 24 | **LA 0,5** → Load Address of i (for result storage)  **LV 0,5** → Load Value of i  **LC 1** → Load Constant 1  **AD** →Add i + 1  **ST** → Store result into i |
| *Jump back to loop header* | 25 | **J 11** → Jump back to loop condition check |
| Call WRITEI(S); | 26 – 27 | **LV 0,6** →Load Value of S  **WRI** → Write Integer (S) |
| Call WRITELN; | 28 | **WLN** → Write Line (newline) |
| End of program (End.) | 29 | **HL** → Halt program execution |