**Quiz 12**

1. **What are the requirements and issues for code generation?**

<**Requirements**>

* Correctness
* Efficiency

<**Issues**>

* Input language: intermediate code (optimized or not)
* Target architecture: must be well understood
* Interplay between
* Instruction Selection
* Instruction Scheduling (Evaluation order)
* Register Allocation
* Memory management

1. **Write about instruction selection and instruction scheduling?**

* Choosing the order of instructions to best utilize resources
* Architecture
* RISC (pipeline)
* Vector processing
* Superscalar and VLIW
* Memory hierarchy
* Ordering to decrease memory fetching
* Latency tolerance – doing something when data does have to be fetched

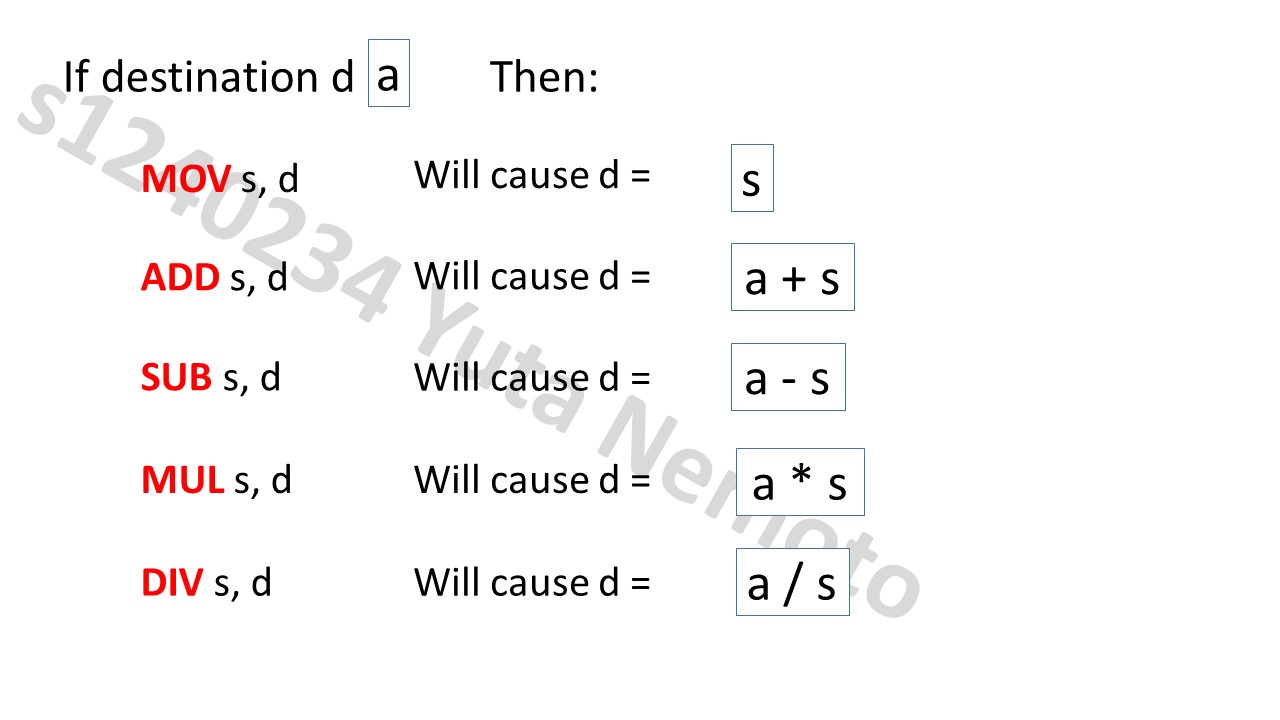
1. **Explain with example about the target machine?**

General Characteristics

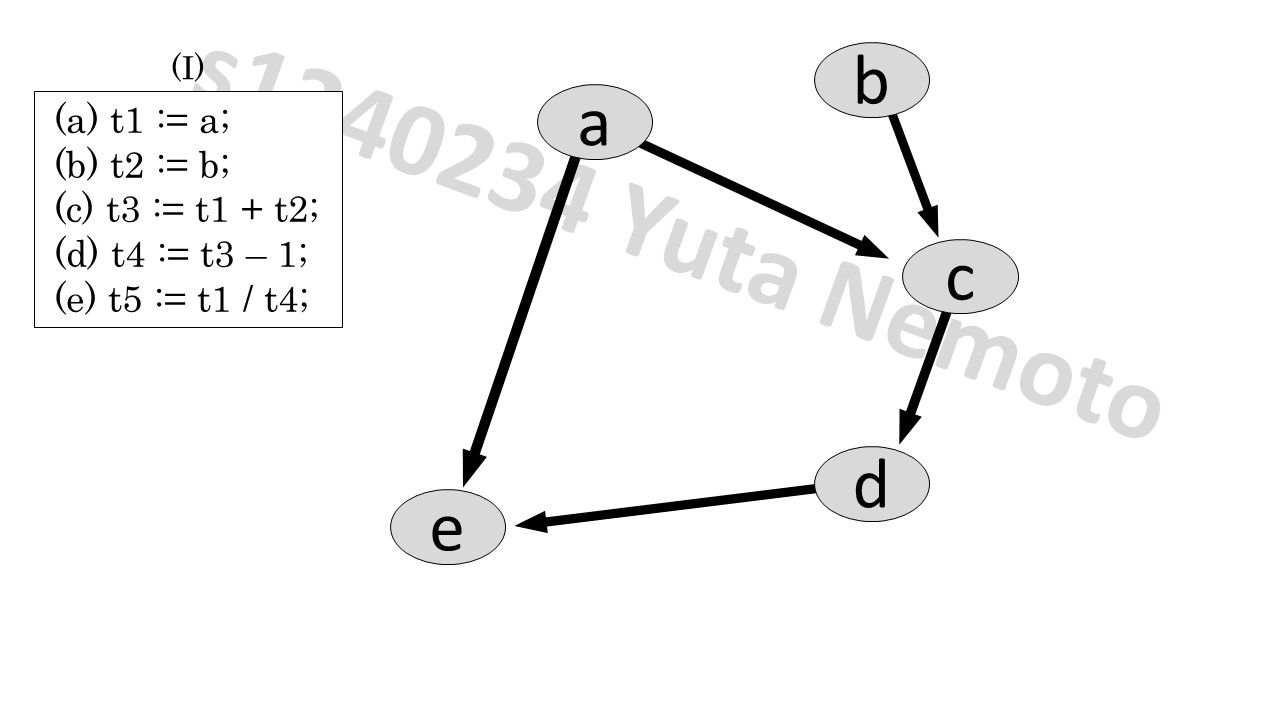
* Byte-addressable with 4-byte words
* N general-purpose registers: R0, R1, …, Rn-1.
* Two-address instructions in the form: op source, destination

Where op is an operator code, and source and destination are data fields.

Here are the examples of operator codes:



1. **Give the data dependency graph for the following set of instructions (I)?**



1. **For two register (R0 and R1) machine generate the code for the expression:**

**a + b + (c \* (d – e))?**

**MOV a, R0**

**ADD b, R0**

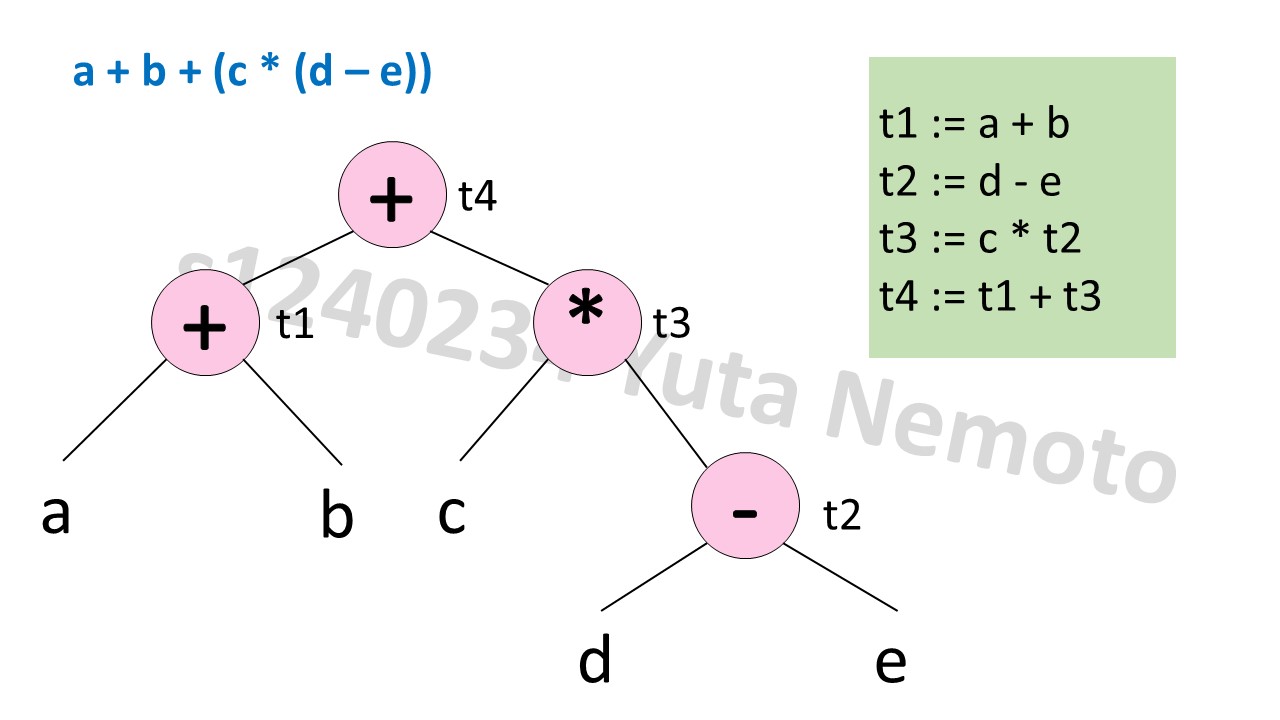
**MOV d, R1**

**SUB e, R1**

**MUL c, R1**

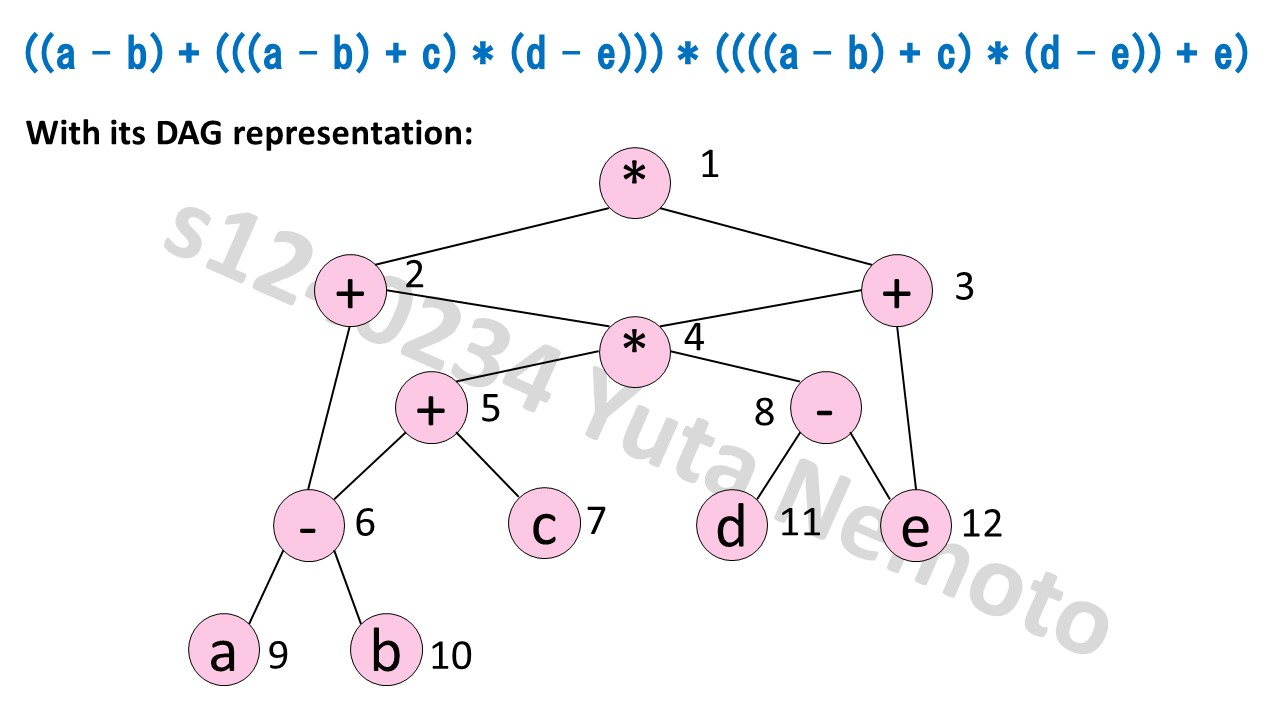
**ADD R1, R0**

**MOV R0, t4**



1. **Give the node listing for the expression:**

**((a – b) + (((a – b) + c) \* (d – e))) \* ((((a – b) + c) \* (d – e)) + e)?**



**List:**

**1: ((a – b) + (((a – b) + c) \* (d – e))) \* ((((a – b) + c) \* (d – e)) + e)**

**2: (a – b) + (((a – b) + c) \* (d – e))**

**3: (((a – b) + c) \* (d – e)) + e**

**4: ((a – b) + c) \* (d – e)**

**5: (a – b) + c**

**6: a – b**

**8: d – e**

1. **Consider the expression: (a \* b) + (e + (c – d))**
2. **Give the intermediate code and its corresponding tree**
3. **Then label the tree**
4. **Finally, generate the code for target machine with two registers R0 and R1**

<< No answer needed for here >>