# Lab 8 Solutions

# Lab 8 - Disassembly Challenge

The following is a disassembled output of a simple C code snippet. Can you figure out what this code snippet does?. If possible translate it back to a pseudocode (high-level language equivalent). Use all of the concepts that you learned so far to solve the challenge.

```
mov dword ptr [ebp-4],1
mov eax,dword ptr [ebp-4]
mov dword ptr [ebp-8],eax
```

# Solution

- At ❶, the program copies a dword value 1 into a memory address (specified by ***ebp-4***).

- At ❷, the same value is copied into the eax register, which is then copied to a different memory address, ***ebp-8***, at ❸.
- We know that in a high-level language like C, a variable that you define (for example, ***int val;***) is just a symbolic name for a memory address

- Let's identify the memory address references and give them a symbolic name. In the disassembled program, we have two addresses (within square brackets): ***ebp-4*** and ***ebp-8***.

- Let's label them and give them symbolic names; let's say, ***ebp-4 = a*** and ***ebp-8 = b***

```
mov dword ptr [ebp-4],1 ❶
mov eax,dword ptr [ebp-4] ❷
mov dword ptr [ebp-8],eax ❸
```

```
mov dword ptr [a],1      ; treat it as mov [a],1
mov eax,dword ptr [a]    ; treat it as mov eax,[a]
mov dword ptr [b],eax    ; treat it as mov [b],eax
```

In a high-level language, when you assign a value to a variable, let's say ***val = 1***, the value ***1*** is moved into the address represented by the ***val*** variable. In assembly, this can be represented as ***mov [val], 1***. This means that ***val = 1***, in a high-level language, is the same as ***mov [val],1*** in assembly. Using this logic, the preceding program can be written into a high-level language equivalent as:

```
a = 1
eax = a
b = eax ❹
```

Tthe registers are used by the CPU for temporary storage. So, let's replace all the register names with their values on the right side of the = sign (for example, replace eax with its value, a, at ❹). The resulting code is shown here:

```
a = 1
eax = a
b = a
```

- In the program, the **eax** register is used to temporarily hold the value of **a**, so we can remove the entry at ❺. We are now left with the simplified code shown here:

```
a = 1
eax = a ❺
b = a
```

```
a = 1
b = a
```