# First Game Mechanics Solutions

Ernesto Rivera-Alvarado and Saúl Guadamuz-Brenes

*Computer Science and Electronic Engineering*
*Costa Rica Institute of Technology*
ernestoriv7@yahoo.com - sguadamuz@itcr.ac.cr

## 1 Sample Questions

The following are the first game mechanics solutions that are proposed for the first iteration of the game.

### 1.1 1st Game Challenge

Given the behavior of the inputs in the game, we can propose the first game challenge to introduce the player to value loading and copying as follows: "Take and input from the *input_ buffer* and load it to the `rax`, `rbx` and `rdi` registers". While the dynamic can be regarded as easy or very easy, it introduces several key concepts: the inputs disappear from the input buffer after reading the value and loading it to a register, the values can be copied between registers, and the value does not change in the source register and the use of the `mov` instruction in two different but common contexts.

Listing 1.1: First challenge solution.

```
mov rax, [input_buffer]
mov rbx, rax
mov rdi, rax
```

### 1.2 2nd Game Challenge

Considering the proposed mechanism for generating outputs, we can introduce more complex challenges: "Take two inputs and compare them. If equal, don't perform any action; if not equal, write the value of the first input to the output". Again, even though this challenge can be regarded as simple, it incorporates several concepts that the student/player should familiarize in order to solve more complex challenges. For instance, this exercise includes the following concepts: reading inputs and writing outputs (as memory addresses), the instruction `cmp` for comparing the values of two registers, decision making based on the `cmp`

Listing 1.2: Second challenge solution.

```
mov rax, [input_buffer]
mov rbx, [input_buffer]

cmp rax, rbx
je exit

mov [input_buffer], rax
exit:
```

instruction. In this specific case, the jump, the instruction `je` used along with the `cmp` instruction for conditional jumps, label concepts and addressing using labels. In this particular case, the `exit:` label.