pikuma.com

MIPS Instruction Set

(Warm-Up Exercises)

The following exercises are designed to introduce you to very simple MIPS instructions that we'll use in the beginning of our course.

I am aware that these exercises will sound a bit pointless, and it might seem like we are simply moving values around the machine. But don't worry; very soon we'll put all these instructions together and create something more meaningful, like sending values to the GPU to render polygons on the screen or keeping the score of a player or the number of enemies in our game.

For now, all I want is for you to get familiarized with very simple instructions of a simple MIPS CPU. These exercises will cover topics like loading values into registers, adding, subtracting, simple conditionals, and loops.

Exercise 1

Your goal here is to simply load registers $**t0**, $**t1**, and $**t2** with immediate values. Try using hexadecimal literals instead of decimals whenever possible.

```
.psx
.create "exercise1.bin", 0x80010000
.org 0x80010000
Main:
  ; TODO:
  ;   1. Load $t0 with the immediate decimal value 1
  ;   2. Load $t1 with the immediate decimal value 256
  ;   3. Load $t2 with the immediate decimal value 17
End:
.close
```

Exercise 2

Your goal here is to start $**t0** with 1 and loop 10 times, incrementing $**t0**. At the end, you should have the accumulated sum of all values (1+2+3+4+5+6+7+8+9+10) inside $**t1**.

```
.psx
.create "exercise2.bin", 0x80010000
.org 0x80010000
Main:
  ; TODO:
  ;   1. Start $t0 with the value 1 and $t1 with the value 0
  ;   2. Loop, incrementing $t0 until it reaches the value 10
  ;   3. Keep adding and accumulating all values of $t0 inside $t1
End:
.close
```

Exercise 3

This exercise requires you to translate a small C code into MIPS assembly. The goal of this short code is to perform integer division between two numbers. We declare three variables in our C code: **num** (numerator), **den** (denominator), and **res** (final result of the integer division). You should assume the values of **num**, **den**, and **res** are stored into $**t0**, $**t1**, and $**t2,** respectively.

```
/* C code: */
main() {
  int num;  // Assume num is loaded in $t0
  int den;  // Assume den is loaded in $t1
  int res;  // Assume res is loaded in $t2
  res = 0;
  while (num >= den) {
    num -= den;
    res++;
  }
}
```

```
.psx
.create "exercise3.bin", 0x80010000
.org 0x80010000
Main:
  ; TODO:
  ;   1. Initialize $t2 (res) with 0
  ;   2. While ($t0 >= $t1) {
  ;   3.   Subtract $t0-$t1 and store it back into $t0
  ;   4.   Increment $t2
  ;   5. }
End:
.close
```