LAPORAN PRAKTIKUM ARSITEKTUR DAN ORGANISASI KOMPUTER

ASSEMBLY PROGRAMMING USING MARS SIMULATOR

Agus Pranata Marpaung
13323033
DIII Teknologi Komputer

INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI

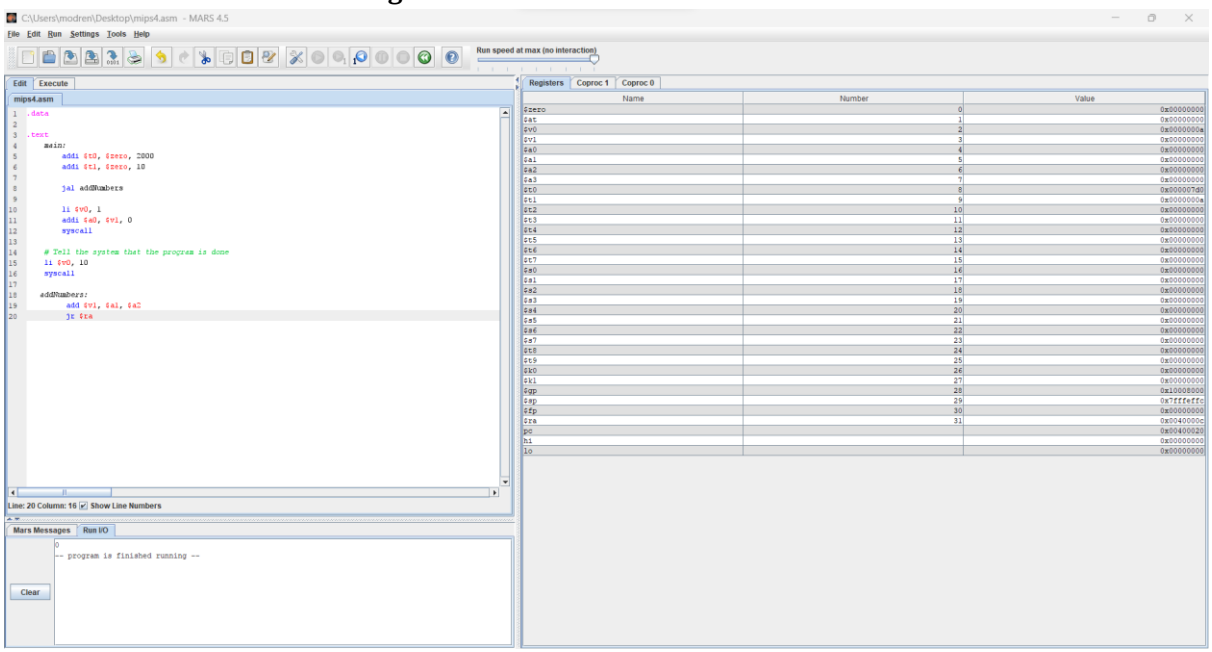
Judul Praktikum

Minggu/Sesi	:	XII/1,2
Kode Mata Kuliah	:	1031103/1041103
Nama Mata Kuliah	:	ARSITEKTUR DAN ORGANISASI KOMPUTER
Setoran	:	Laporan Materi Assembly Programming using MARS Simulator dikirimkan dalam bentuk PDF pada ecourse
Batas Waktu Setoran	:	*5 Desember 2023 jam 8:00*
Tujuan	:	1. Mahasiswa mampu mengenal Assembly Programming 2. Mahasiswa mampu menggunakan MARS Simulator 3. Mahasiswa mampu menerapkan Assembly Programming pada MARS Simulator

Petunjuk

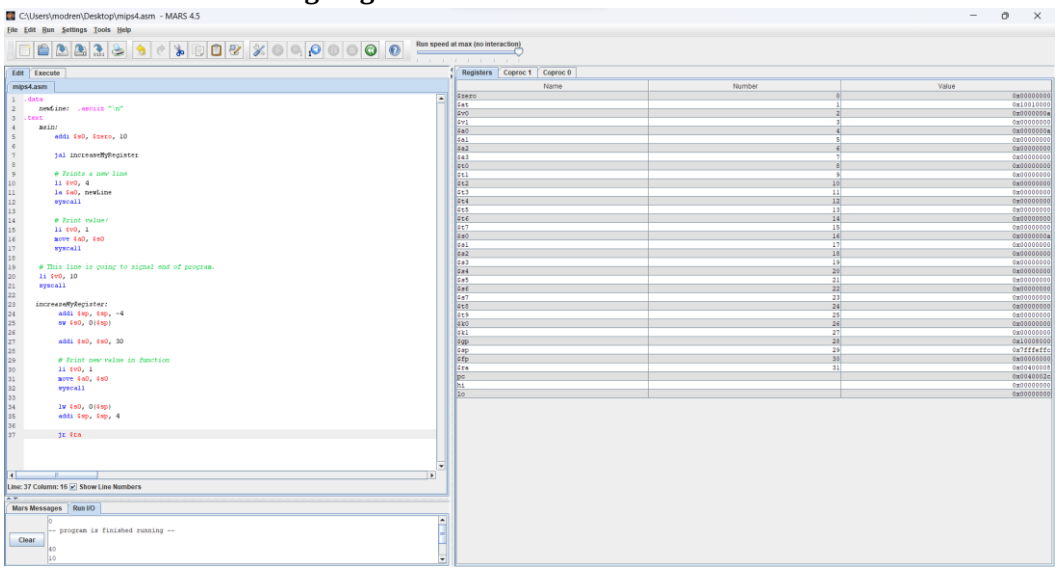
1. Laporan praktikum dikerjakan secara individu (tidak berkelompok)
2. Setiap individu diperbolehkan memberikan pertanyaan dan diskusi melalui WAG pada sesi kedua di hari praktikum
3. Tidak ada toleransi keterlambatan, jika terlambat maka akan terjadi pengurangan nilai.
4. Dalam pengerjaan laporan praktikum, dilarang keras melakukan plagiasi (mencontek).

1. MIPS Tutorial 16 Function Arguments and Return Values



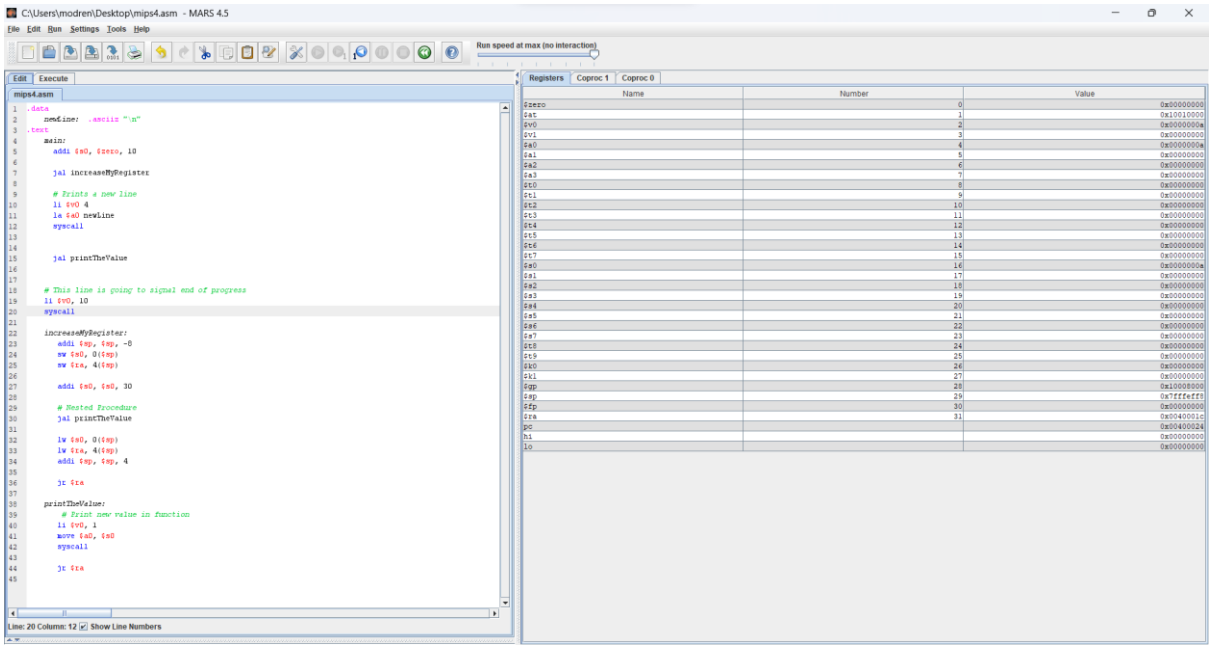
- **.data** : mendeklarasikan segmen data.
- **.text** : berisi kode program utama atau instruksi-instruksi yang akan dieksekusi oleh CPU.
- **main** : label untuk fungsi utama program.
- **addi $t0, $zero, 2000** **:** Menambahkan nilai 2000 ke register $t0.
- **addi $t1, $zero, 10** **:** Menambahkan nilai 10 ke register $t1.
- **jal addNumbers** **:** Melakukan "Jump And Link" Ke Fungsi addNumbers.
- **li $v0, 1** **:** Memuat nilai 1 ke register $v0.
- **addi $a0, $v1, 0** **:** Menyalin nilai dari register $v1 ke register $a0.
- **syscall** : mengeksekusi layanan sistem berdasarkan nilai yang ada di register $v0.
- **li $v0, 10** : Memuat nilai 10 ke register $v0.
- **syscall** : Mengakhiri program.

2. MIPS Tutorial 17 Saving Registers to the Stack



- **.data** : Mendeklarasikan Segmen Data.
- **.text** : Berisi Kode Program Utama Atau Instruksi-Instruksi Yang Akan Dieksekusi Oleh CPU.
- **addi $s0, $zero, 10** : Menambahkan nilai 10 ke register $s0.
- **jal increaseMyRegister**: Melakukan "jump and link" ke fungsi increaseMyRegister.
- **li $v0, 4** : Memuat nilai 4 ke register $v0.
- **la $a0, newline** : Memuat alamat string newLine ke register $a0.
- **syscall** : Mengeksekusi Layanan Sistem Berdasarkan Nilai Yang Ada Di Register $V0.
- **li $v0, 1** : Memuat nilai 1 ke register $v0.
- **move $a0, $s0** : Memindahkan nilai dari register $s0 ke register $a0.
- **syscall** : Mengeksekusi layanan sistem berdasarkan nilai yang ada di register $v0.
- **li $v0, 10** : Memuat nilai 10 ke register $v0.
- **syscall** : Mengakhiri program.

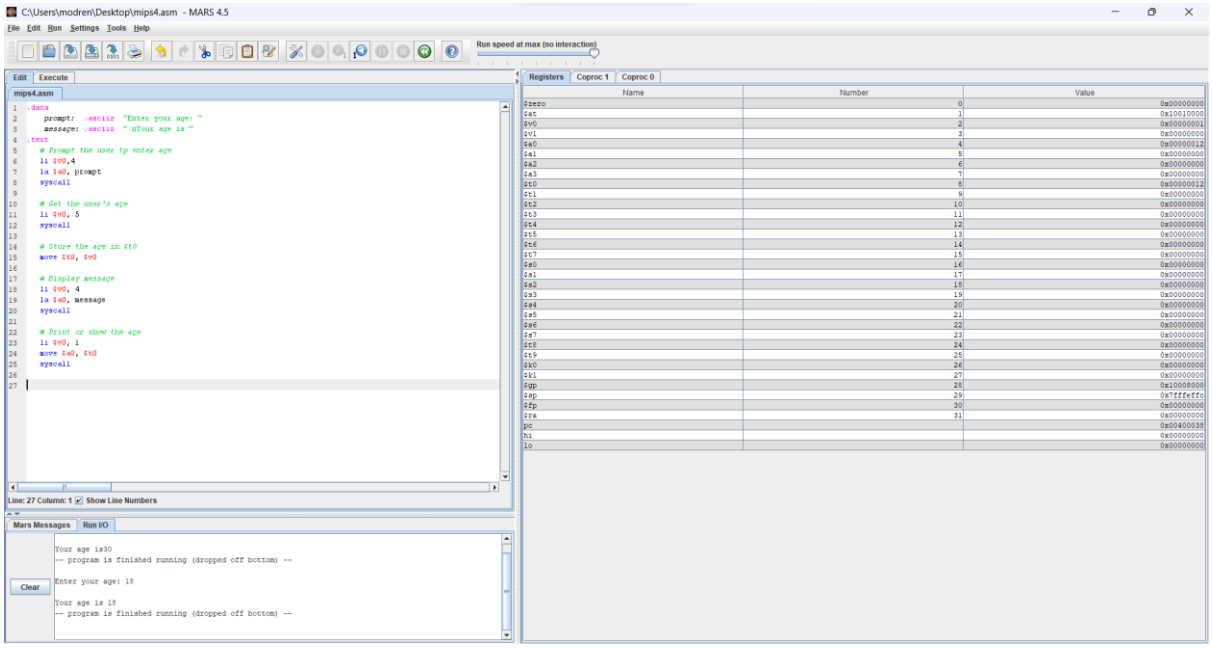
3. MIPS Tutorial 18 Nested Procedures



- **.data** : Mendeklarasikan Segmen Data.
- **.text** : Berisi Kode Program Utama Atau Instruksi-Instruksi Yang Akan Dieksekusi Oleh CPU.
- **addi $s0, $zero, 10** : Menambahkan nilai 10 ke register $s0.
- **jal increaseMyRegister**: Melakukan "jump and link" ke fungsi increaseMyRegister.
- **li $v0 4** : Memuat nilai 4 ke register $v0.
- **la $a0 newLine** : Memuat alamat string newLine ke register $a0.
- **Syscall** : Mengeksekusi Layanan Sistem Berdasarkan Nilai Yang Ada Di Register $V0.
- **jal printTheValue** : Melakukan "jump and link" ke fungsi printTheValue.
- **li $v0, 10** : Memuat nilai 10 ke register $v0.
- **Syscall** : Mengakhiri Program
- **addi $s0, $zero, 10** : Menambahkan nilai 10 ke register $s0.
- **jal increaseMyRegister**: Melakukan "jump and link" ke fungsi increaseMyRegister.
- **li $v0 4** : Memuat nilai 4 ke register $v0.
- **la $a0 newLine** : Memuat alamat string newLine ke register $a0.
- **Syscall** : Mengeksekusi Layanan Sistem Berdasarkan Nilai Yang Ada Di Register $V0.
- **jal printTheValue** : Melakukan "jump and link" ke fungsi printTheValue.
- **li $v0, 10** : Memuat nilai 10 ke register $v0.
- **Syscall** : mengakhiri program
- **addi $s0, $zero, 10** : Menambahkan nilai 10 ke register $s0.
- **jal increaseMyRegister**: Melakukan "jump and link" ke fungsi increaseMyRegister.
- **li $v0 4** : Memuat nilai 4 ke register $v0.
- **la $a0 newLine** : Memuat alamat string newLine ke register $a0.
- **Syscall** : Mengeksekusi Layanan Sistem Berdasarkan Nilai Yang Ada Di Register $V0.
- **jal printTheValue** : Melakukan "jump and link" ke fungsi printTheValue.
- **li $v0, 10** : Memuat nilai 10 ke register $v0.

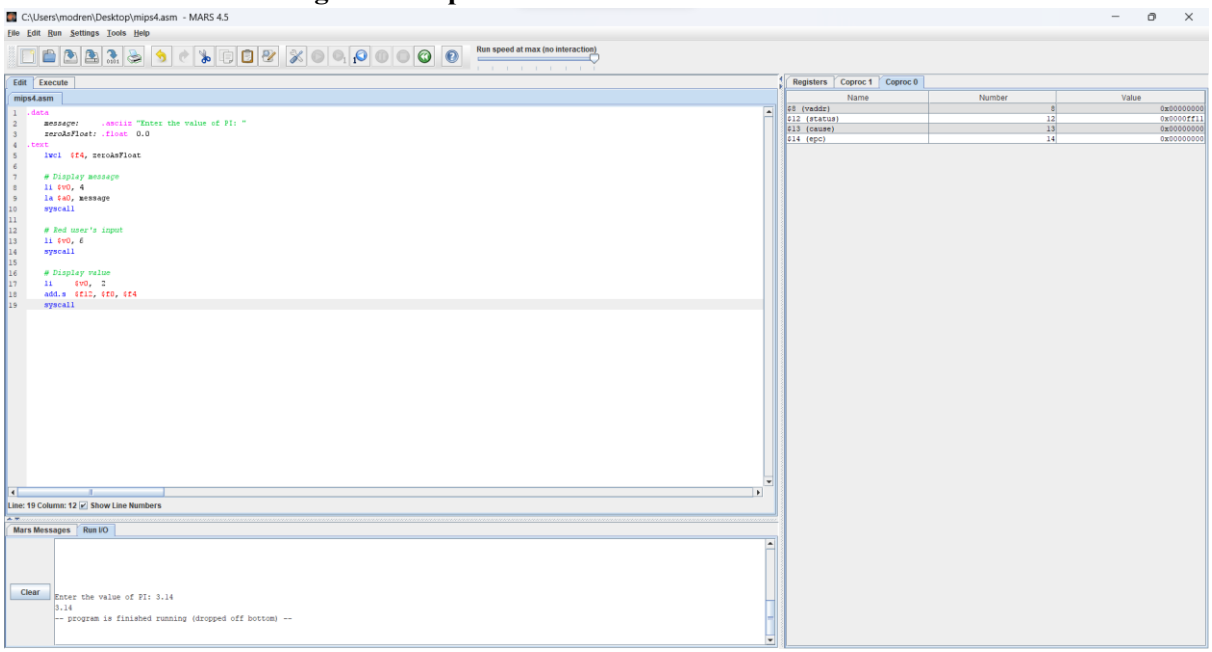
- **Syscall** : Mengakhiri Program.

4. MIPS Tutorial 19 Getting User's Input Integers



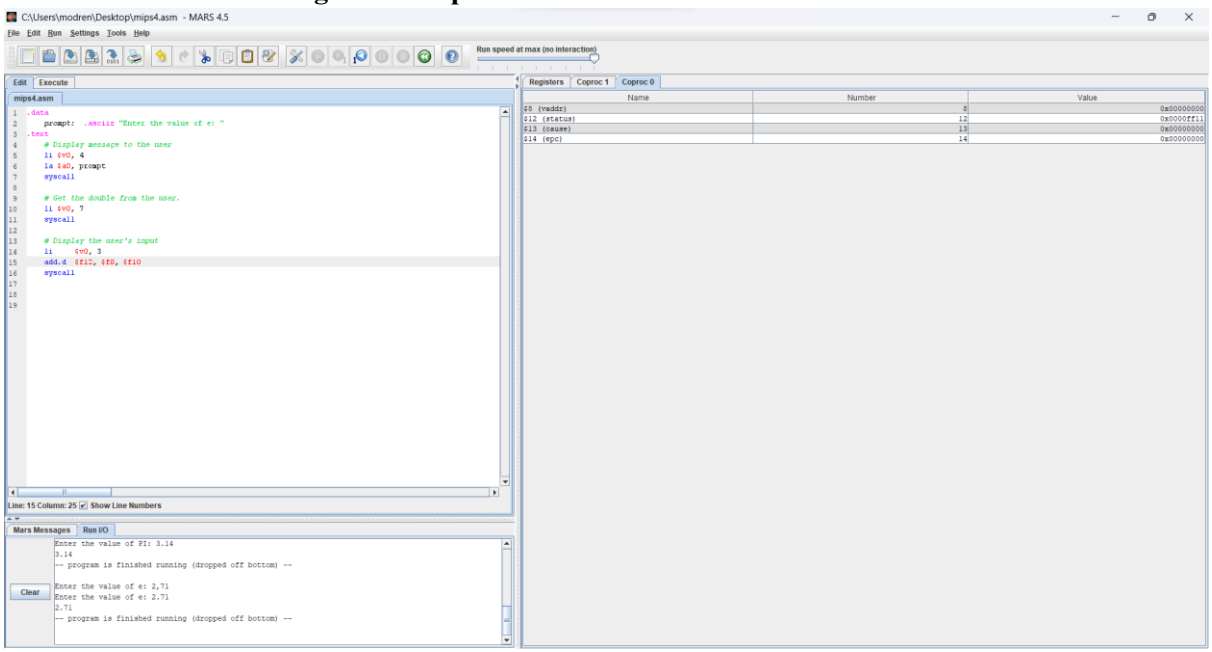
- **.data** : Mendeklarasikan Segmen Data.
- **prompt: .asciiz "Enter your age: "** : Menginisialisasi string "Enter your age: " dan memberi nama prompt.
- **message: .asciiz "\nYour age is "** : Menginisialisasi string "\nYour age is " dan memberi nama message.
- **.text** : Berisi Kode Program Utama Atau Instruksi-Instruksi Yang Akan Dieksekusi Oleh CPU.
- **li $v0, 4** : Mengatur register $v0 dengan nilai 4.
- **la $a0, prompt** : Memuat alamat prompt ke dalam register $a0.
- **syscall** : Memanggil sistem untuk mencetak string yang diidentifikasi oleh alamat yang disimpan di $a0.
- **li $v0, 5** : Mengatur register $v0 dengan nilai 5.
- **syscall** : Memanggil sistem untuk membaca integer dari input pengguna
- **move $t0, $v0** : Memindahkan nilai usia yang dibaca ke dalam register $t0.
- **li $v0, 4** : Mengatur register $v0 dengan nilai 4.
- **la $a0, message** : Memuat alamat message ke dalam register $a0.
- **syscall** : Memanggil sistem untuk mencetak string yang diidentifikasi oleh alamat yang disimpan di $a0.
- **li $v0, 1** : Mengatur register $v0 dengan nilai 1.
- **move $a0, $t0** : Memuat nilai usia dari register $t0 ke dalam register $a0.
- **syscall** : Memanggil sistem untuk mencetak nilai usia ke layar.

5. MIPS Tutorial 20 Getting User's input floats



- **.data** : Mendeklarasikan Segmen Data.
- **message: .asciiz "Enter the value of PI: "** : Mendefinisikan String "Enter The Value Of PI: " Dan Memberi Label Message Padanya.
- **zeroAsFloat: .float 0.0** : Variabel Zeroasfloat Sebagai Float Dengan Nilai Awal 0.0.
- **.text** : Berisi Kode Program Utama Atau Instruksi-Instruksi Yang Akan Dieksekusi Oleh CPU.
- **lwc1 $f4, zeroAsFloat** : Memuat Nilai 0.0 (Zeroasfloat) Ke Dalam Register Floating Point $F4.
- **li $v0, 4** : Mengatur nilai register $v0 dengan 4.
- **la $a0, message** : Memuat alamat string message ke dalam register $a0.
- **syscall** : Menjalankan sistem call untuk mencetak string ke layar.
- **li $v0, 6** : Mengatur nilai register $v0 dengan 6.
- **syscall** : Menjalankan sistem call untuk membaca nilai float dari pengguna dan menyimpannya di register floating point $f0.
- **li $v0, 2** : Mengatur nilai register $v0 dengan 2.
- **add.s $f12, $f0, $f4** : Menambahkan nilai float yang dibaca dari pengguna ($f0) dengan nilai 0.0 ($f4) dan menyimpan hasilnya di register floating point $f12.
- **syscall** : Menjalankan sistem call untuk mencetak nilai float ke layar.

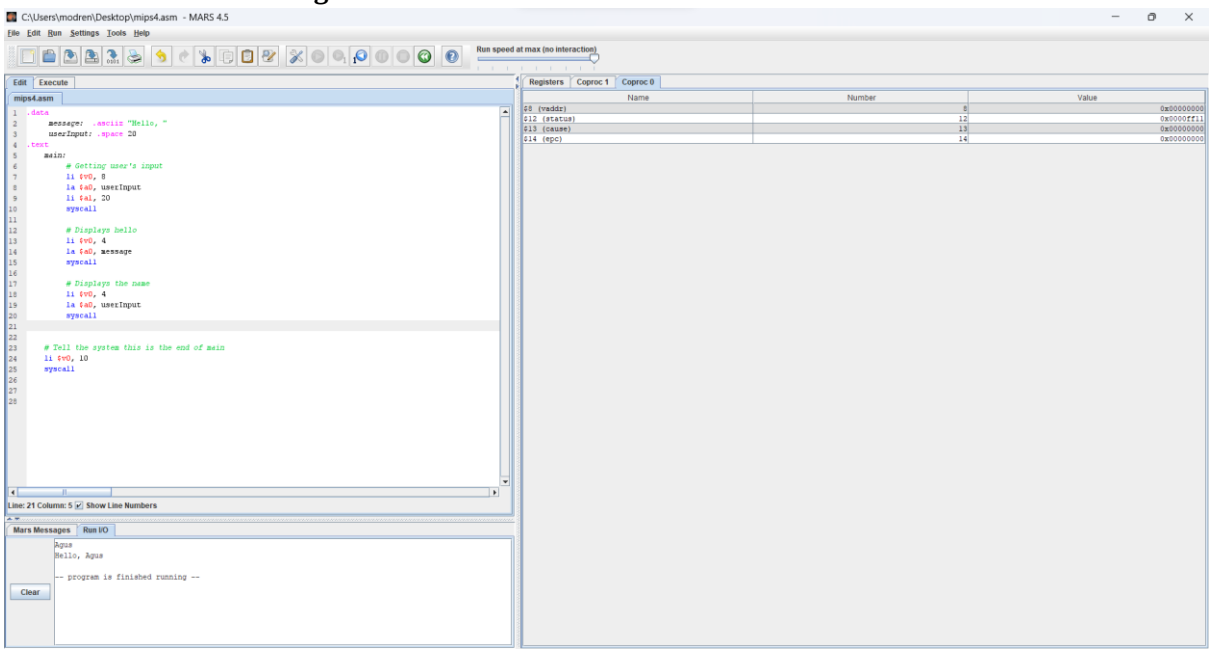
6. MIPS Tutorial 21 Getting User's Input doubles



- **.data** : Mendeklarasikan Segmen Data Dan Memiliki Satu Variabel String Yang Disebut Prompt Yang Berisi Pesan "Enter The Value Of E: ".
- **.text** : Bagian ini berisi kode program utama.
- **li $v0, 4** : Load immediate, digunakan untuk mengatur nilai register $v0 ke 4
- **la $a0, prompt** : Load address, digunakan untuk memuat alamat string prompt ke dalam register $a0
- **syscall** : Sistem call untuk mencetak string.
- **li $v0, 7** : mengambil input dari pengguna dan mengatur nilai register $v0 ke 7.
- **syscall** : Sistem call untuk membaca integer dari pengguna.
- **li $v0, 3** : Load immediate, mengatur nilai register $v0 ke 3.
- **add.d $f12, $f0, $f10** : Menambahkan nilai di register $f0 dengan nilai di register $f10 dan menyimpan hasilnya di register $f12.
- **syscall** : Sistem call untuk mencetak nilai floating-point.

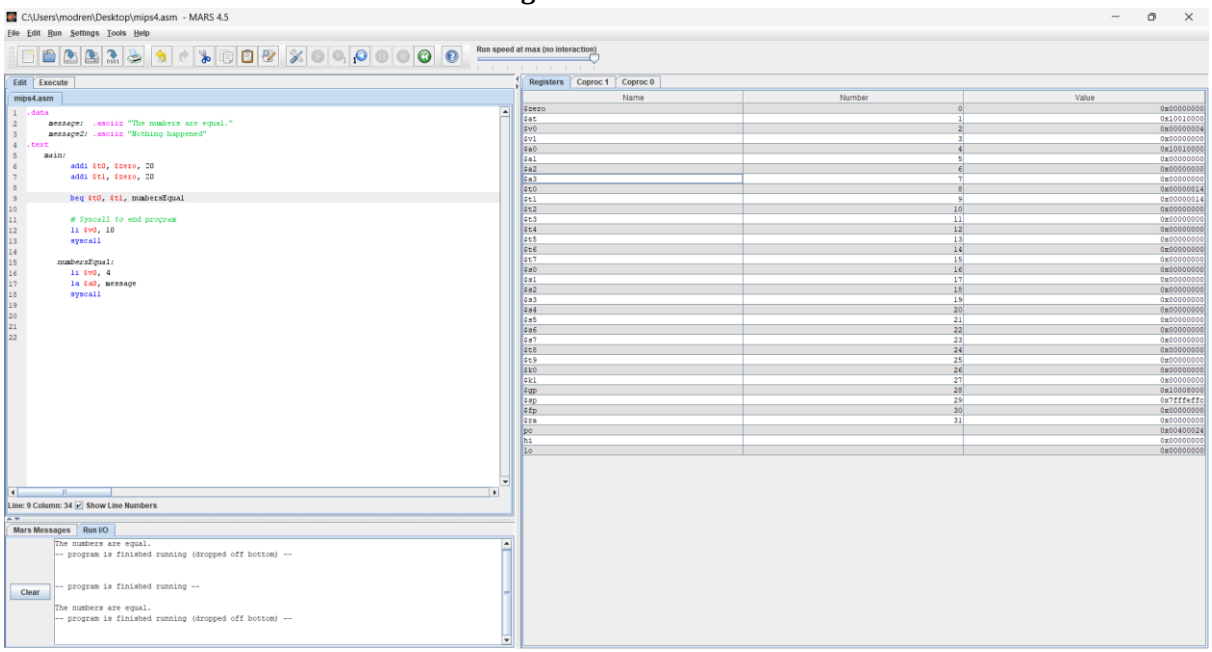
\

7. MIPS Tutorial 22 Getting text from the user

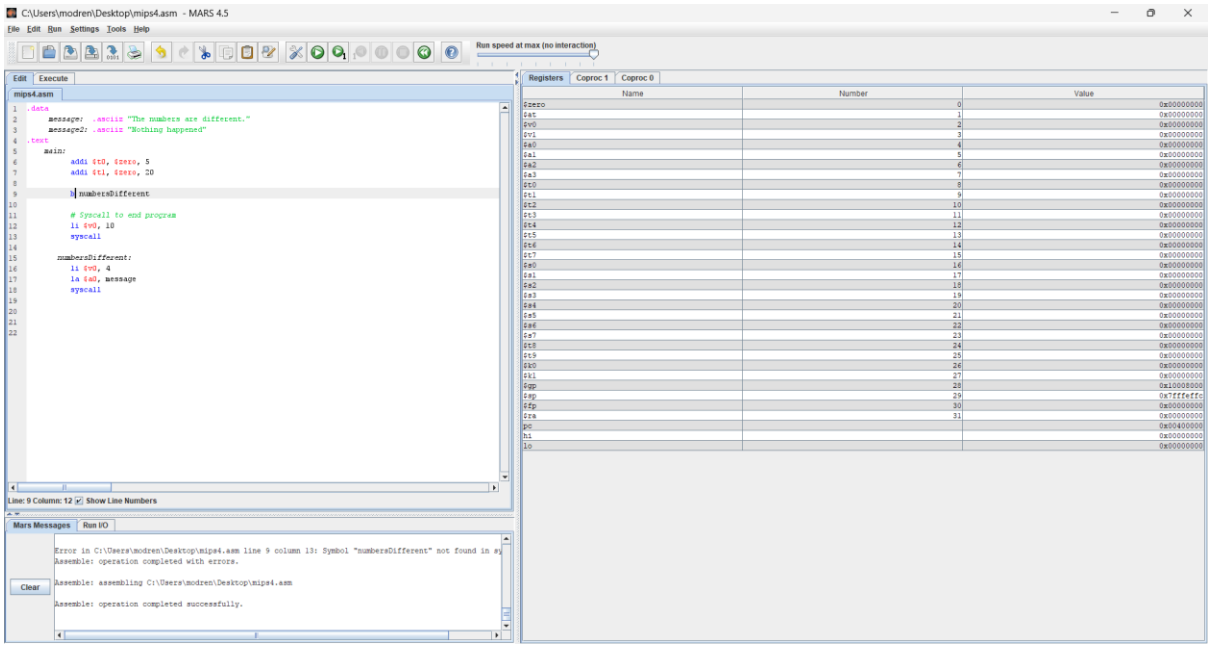


- **.data** : Menginisialisasi Data Yang Akan Digunakan Dalam Program.
- **message** : Sebuah string yang berisi "Hello, ".
- **userInput** : Sebuah buffer berukuran 20 byte untuk menyimpan input pengguna.
- **.text** : Dimulainya Instruksi Program.
- **main** : Menandai Awal Dari Program Utama.
- **li $v0, 8** : Mengatur nilai register $v0 ke 8
- **la $a0, userInput** : Memuat alamat dari variabel userInput ke dalam register $a0.
- **li $a1, 20** : Menetapkan panjang maksimal input pengguna sebesar 20 karakter.
- **syscall** : Memanggil sistem untuk membaca input dari pengguna.
- **li $v0, 4** : Mengatur nilai register $v0 ke 4
- **la $a0, message** : Memuat alamat dari variabel message ke dalam register $a0.
- **syscall** : Memanggil sistem untuk menampilkan pesan salam "Hello, "
- **li $v0, 4** : Mengatur nilai register $v0 ke 4
- **la $a0, userInput** : Memuat alamat dari variabel userInput ke dalam register $a0.
- **syscall** : Memanggil sistem untuk menampilkan input pengguna.
- **li $v0, 10** : Mengatur nilai register $v0 ke 10
- **syscall** : Memanggil sistem untuk mengakhiri program

8. MIPS Tutorial 23 If statements Branching Instructions

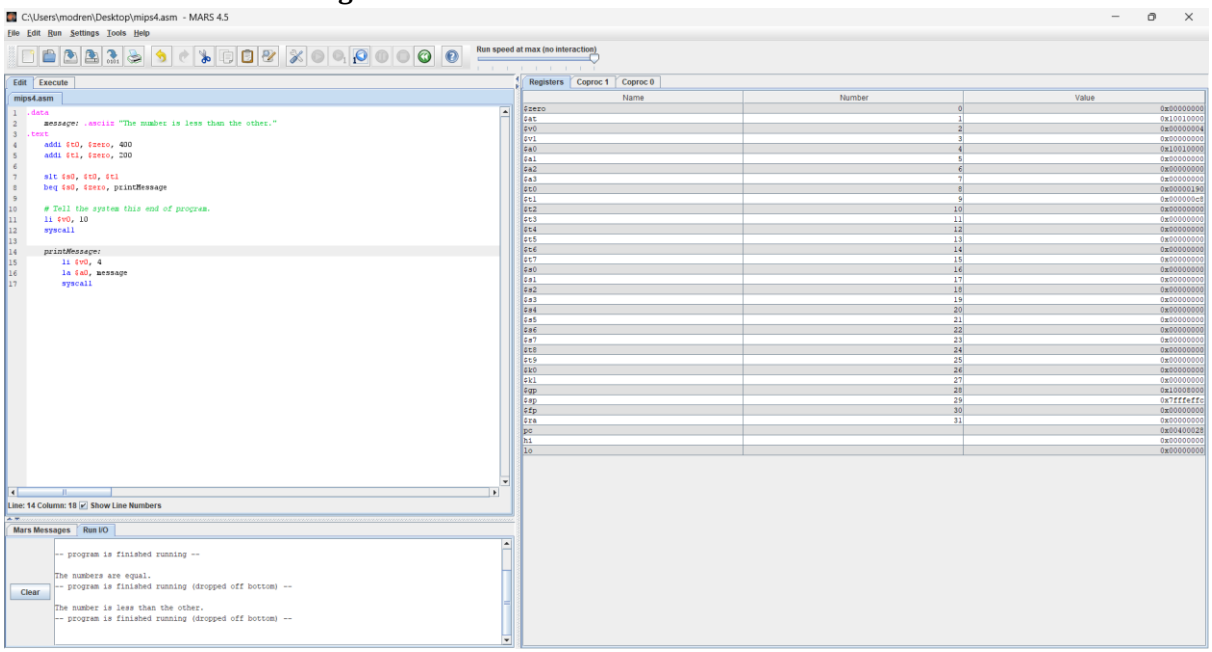


- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **message: .asciiz "The numbers are equal."**: String yang akan dicetak jika kedua angka sama.
- **message2: .asciiz "Nothing happened"** : String yang akan dicetak jika kedua angka tidak sama.
- **.text** : Dimulainya instruksi program.
- **main** : Menandai awal dari program utama.
- **addi $t0, $zero, 20** : Menginisialisasi register $t0 dengan nilai 20.
- **addi $t1, $zero, 20** : Menginisialisasi register $t1 dengan nilai 20.
- **beq $t0, $t1, numbersEqual** : Membandingkan nilai di register $t0 dan $t1.
- **li $v0, 10** : Jika nilai di $t0 dan $t1 tidak sama, program akan keluar menggunakan syscall dengan nilai sistem 10.
- **Syscall** **:** Menjalankan syscall yang sesuai dengan nilai yang ada di register $v0
- **numbersEqual** **:** nilai dalam $t0 sama dengan nilai dalam $t1.
- **li $v0, 4** **:** mengatur nilai register $v0 ke 4
- **la $a0, message** **:** mengatur alamat string "The numbers are equal." ke register $a0
- **syscall** **:** memanggil sistem operasi dan menampilkan string



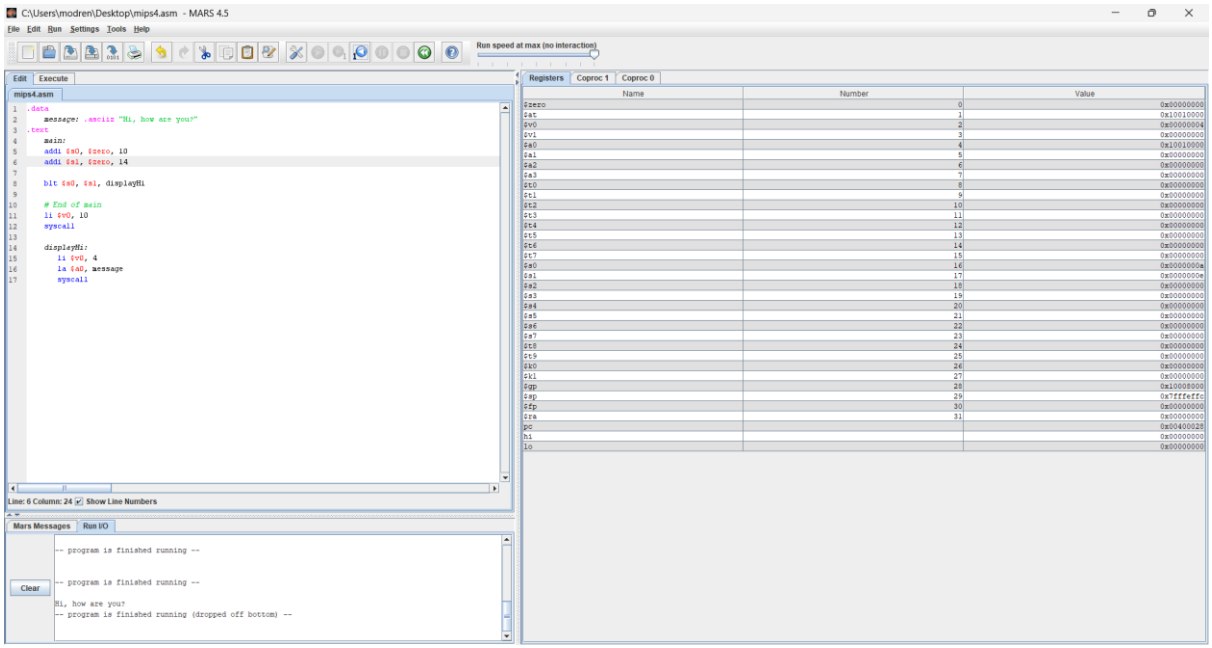
- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **message: .asciiz "The numbers are different"**: String yang akan dicetak jika kedua angka tidak sama.
- **message2: .asciiz "Nothing happened"** : String yang akan dicetak jika kedua angka sama.
- **.text** : Dimulainya instruksi program.
- **main** : Menandai awal dari program utama.
- **addi $t0, $zero, 5** : Menginisialisasi register $t0 dengan nilai 5.
- **addi $t1, $zero, 20** : Menginisialisasi register $t1 dengan nilai 20.
- **b numbersDifferent** : Memindahkan eksekusi program ke bagian yang ditunjuk oleh label
- **li $v0, 10** : Memasukkan nilai 10 ke register $v0
- **syscall** **:** Menjalankan syscall yang sesuai dengan nilai yang ada di register $v0
- **numbersDifferent** **:** Label yang dituju ke instruksi branch sebelumnya.
- **li $v0, 4** : Memasukkan nilai 4 ke register $v0.
- **la $a0, message** : Mengambil alamat dari string message dan menyimpannya di register $a0.
- **syscall** **:** Menjalankan syscall yang sesuai dengan nilai yang ada di register $v0.

9. MIPS Tutorial 24 Checking If a number is less than Another slt



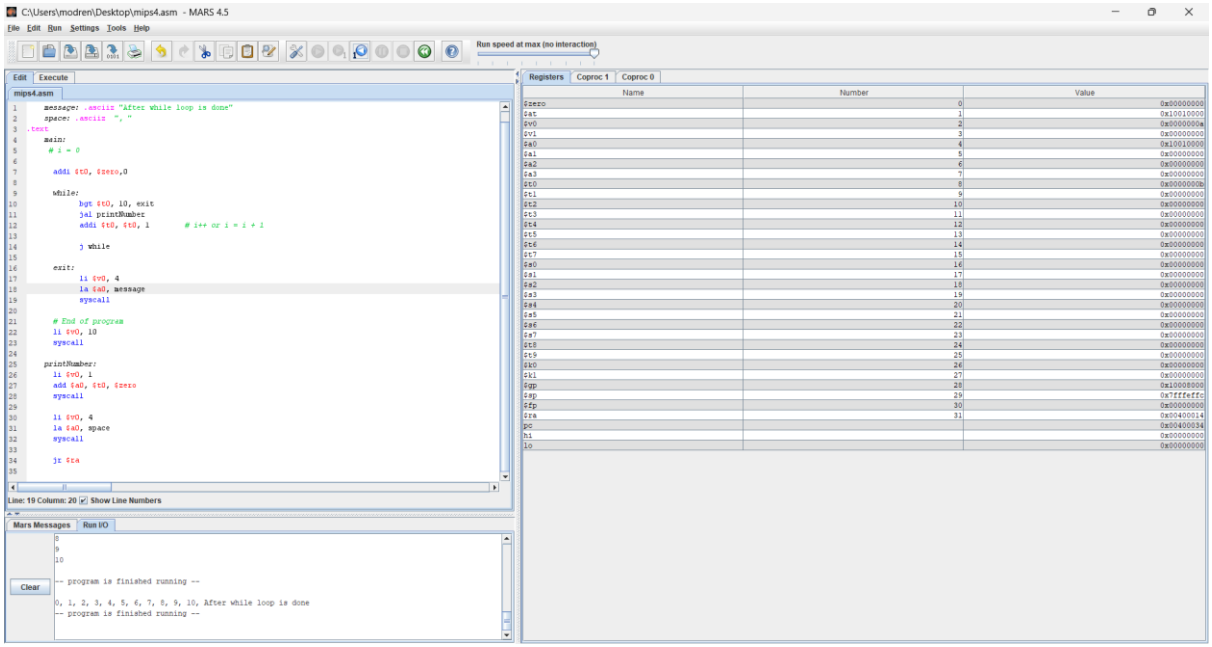
- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **message: .asciiz "The number is less than the other.":** Mendefinisikan label message dan menginisialisasi string "The number is less than the other."
- **.text** : Dimulainya instruksi program.
- **main** : Menandai awal dari program utama.
- **addi $t0, $zero, 400** **:** Menambahkan nilai 400 ke register $t0.
- **addi $t1, $zero, 200** **:** Menambahkan nilai 200 ke register $t1.
- **slt $s0, $t0, $t1** **:** Mengatur $s0 ke 1 jika nilai di $t0 kurang dari $t1, dan ke 0 jika tidak.
- **beq $s0, $zero, printMessage** **:** Branch jika nilai di $s0 sama dengan 0.
- **li $v0, 10** **:** Menempatkan nilai 10 ke register $v0.
- **syscall** **:**Memanggil sistem operasi dengan menggunakan sistem call yang sudah diatur sebelumnya.
- **printMessage:** **:** Label yang akan diakses jika kondisi branch tidak terpenuhi.
- **li $v0, 4** **:** Menempatkan nilai 4 ke register $v0. Ini adalah sistem call number untuk mencetak string.
- **la $a0, message** **:** Mengambil alamat string dari label message dan menempatkannya di register $a0.
- **syscall** **:** Memanggil sistem operasi untuk mencetak string yang alamatnya ada di register $a0.

10. MIPS Tutorial 25 Branching Pseudo Instructions



- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **message: .asciiz "The number is less than the other."** : Mendefinisikan label message dan menginisialisasi string "The number is less than the other."
- **.text** : Dimulainya instruksi program.
- **main** : Menandai awal dari program utama.
- **addi $t0, $zero, 400** : Menambahkan nilai 400 ke register $t0.
- **addi $t1, $zero, 200** : Menambahkan nilai 200 ke register $t1.
- **slt $s0, $t0, $t1** : Mengatur $s0 ke 1 jika nilai di $t0 kurang dari $t1, dan ke 0 jika tidak.
- **beq $s0, $zero, printMessage** : Branch jika nilai di $s0 sama dengan 0.
- **li $v0, 10** : Menempatkan nilai 10 ke register $v0.
- **syscall** : Memanggil sistem operasi dengan menggunakan sistem call yang sudah diatur sebelumnya.
- **printMessage:** : Label yang akan diakses jika kondisi branch tidak terpenuhi.
- **li $v0, 4** : Menempatkan nilai 4 ke register $v0.
- **la $a0, message** : Mengambil alamat string dari label message dan menempatkannya di register $a0.
- **syscall** : Memanggil sistem operasi untuk mencetak string yang alamatnya ada di register $a0.

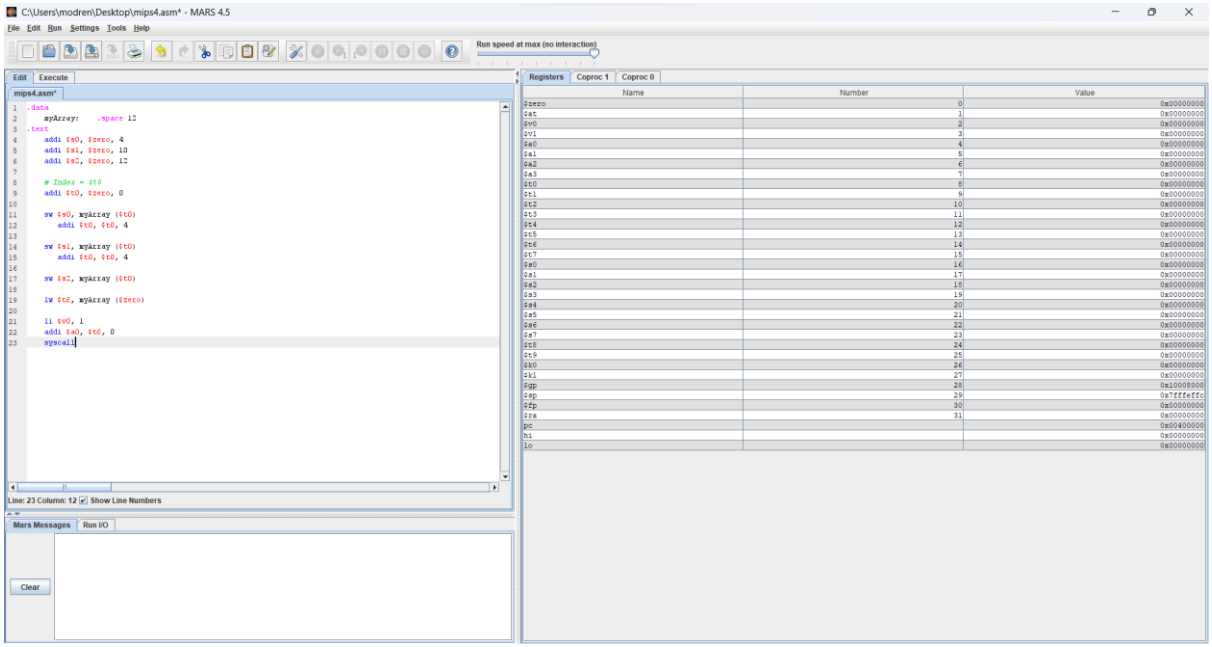
11. MIPS Tutorial 26 While Loop in MIPS



- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **message: .asciiz "After while loop is done"** : Mendefinisikan string dengan pesan "After while loop is done".
- **space: .asciiz ", "** : Mendefinisikan string dengan koma dan spasi.
- **main** : Menandai awal dari program utama.
- **addi $t0, $zero, 0** : Menginisialisasi register $t0 dengan nilai 0.
- **while:** : Label yang menandakan awal dari loop.
- **bgt $t0, 10, exit** : Instruksi branch jika nilai di $t0 lebih besar dari 10.
- **jal printNumber** : Melakukan jump-and-link ke fungsi printNumber.
- **addi $t0, $t0, 1** : Menambahkan nilai register $t0 dengan 1. Ini adalah langkah increment untuk loop.
- **j while** : Instruksi untuk melompat kembali ke label while, sehingga membuat loop.
- **exit:** : Label yang menandakan akhir dari loop.
- **li $v0, 4** : Menyiapkan system call untuk mencetak string.
- **la $a0, message** : Memuat alamat string message ke register $a0.

- **syscall** : Melakukan system call untuk mencetak string yang diidentifikasi oleh $a0.
- **li $v0, 10** : Menyiapkan system call untuk mengakhiri program.
- **syscall** : Melakukan system call untuk mengakhiri program.
- **printNumber:** : Label untuk fungsi printNumber.
- **li $v0, 1** : Menyiapkan system call untuk mencetak integer.
- **add $a0, $t0, $zero** : Memuat nilai register $t0 ke register $a0.
- **syscall** : Melakukan system call untuk mencetak nilai integer yang diidentifikasi oleh $a0.
- **li $v0, 4** : Menyiapkan system call untuk mencetak string.
- **la $a0, space** : Memuat alamat string space ke register $a0.
- **syscall** : Melakukan system call untuk mencetak string yang diidentifikasi oleh $a0.
- **jr $ra** : Melompat kembali ke alamat yang disimpan di register $ra.

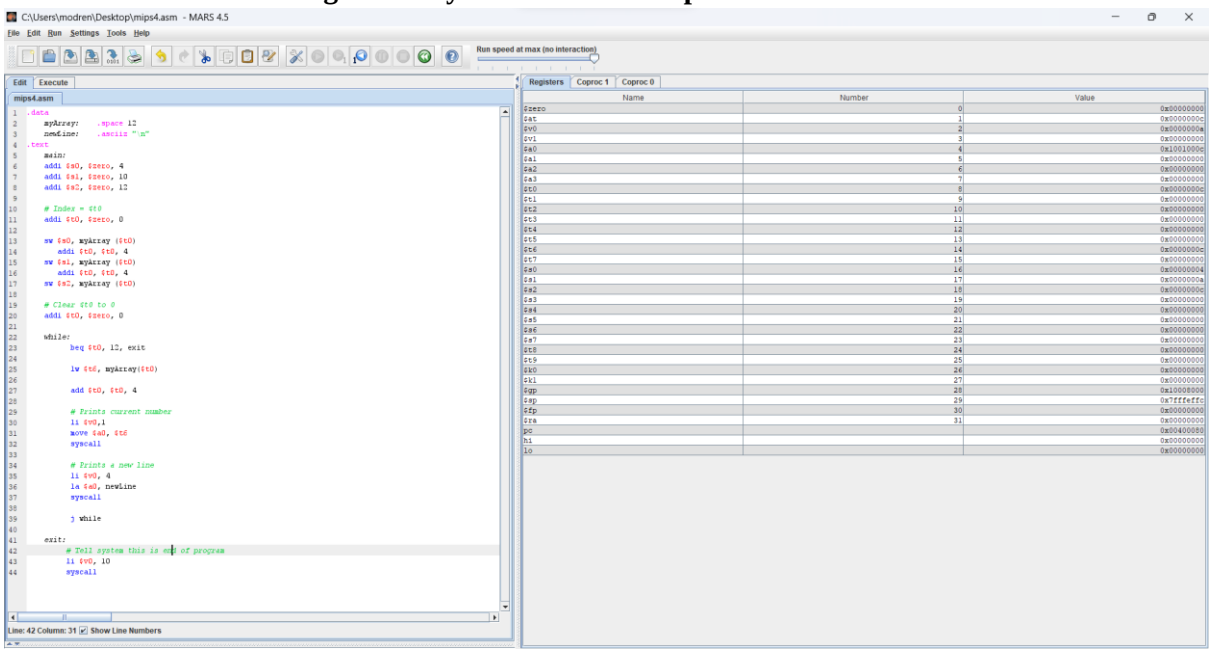
12.MIPS Tutorial 27 Arrays



- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **myArray: .space 12** : Mengalokasikan ruang 12 byte untuk larik bernama myArray.
- **.text** : Dimulainya instruksi program.
- **addi $s0, $zero, 4** : Menetapkan nilai 4 ke register $s0.
- **addi $s1, $zero, 10** : Menetapkan nilai 10 ke register $s1.

- **addi $s2, $zero, 12** : Menetapkan nilai 12 ke register $s2.
- **addi $t0, $zero, 0** : Menetapkan nilai 0 ke register $t0.
- **sw $s0, myArray ($t0)** : Menyimpan nilai dari register $s0 ke lokasi memori yang ditunjuk oleh myArray dan offset $t0.
- **addi $t0, $t0, 4** : Menambahkan 4 ke nilai di register $t0.
- **sw $s1, myArray ($t0)** : Menyimpan nilai dari register $s1 ke lokasi memori yang ditunjuk oleh myArray dan offset $t0.
- **addi $t0, $t0, 4** : Menambahkan 4 ke nilai di register $t0.
- **sw $s2, myArray ($t0)** : Menyimpan nilai dari register $s2 ke lokasi memori yang ditunjuk oleh myArray dan offset $t0.
- **lw $t6, myArray ($zero)** : Memuat nilai dari lokasi memori yang ditunjuk oleh myArray ke register $t6.
- **li $v0, 1** : Menetapkan nilai 1 ke register $v0.
- **addi $a0, $t6, 0** : Menetapkan nilai dari register $t6 ke register $a0.
- **syscall** : Memanggil sistem untuk mencetak nilai yang ada di register $a0.

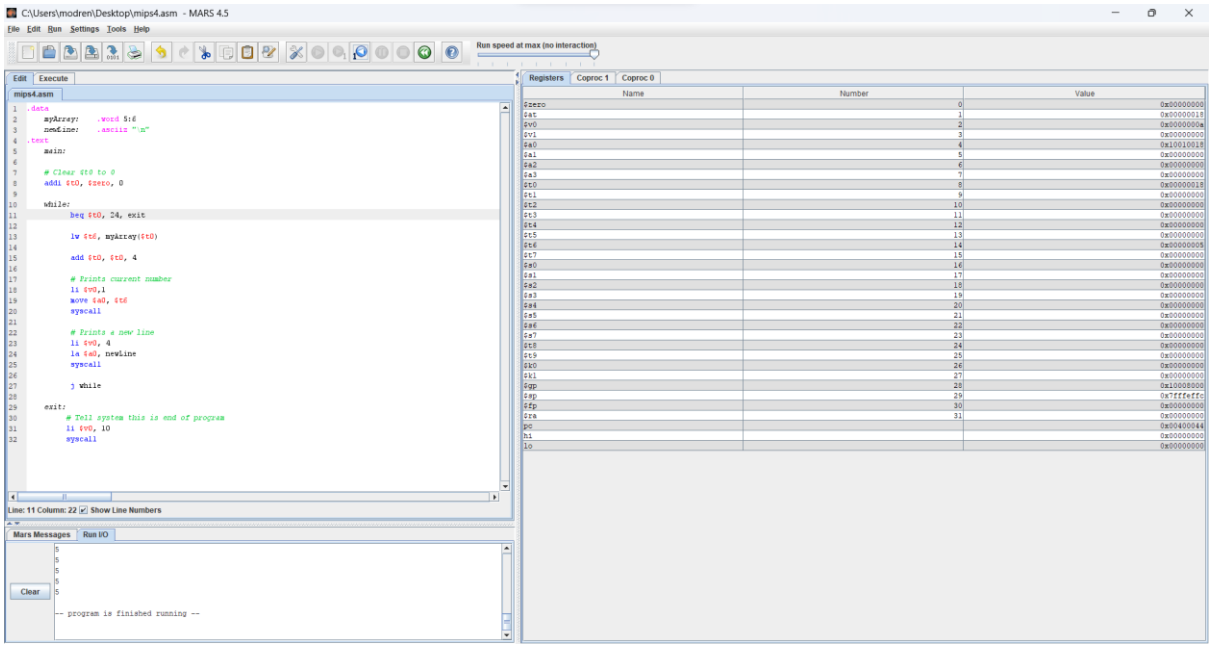
13. MIPS Tutorial 28 Printing an Array with a While Loop



- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **myArray: .space 12** : Mengalokasikan 12 byte untuk array bernama myArray.
- **newLine: .asciiz "\n"** : Mendefinisikan string newline ("\n").
- **.text** : Dimulainya instruksi program.
- **addi $s0, $zero, 4** : Menginisialisasi register $s0 dengan nilai 4.
- **addi $s1, $zero, 10** : Menginisialisasi register $s1 dengan nilai 10.
- **addi $s2, $zero, 12** : Menginisialisasi register $s2 dengan nilai 12.
- **addi $t0, $zero, 0** : Menginisialisasi register $t0 (index) dengan nilai 0.
- **sw $s0, myArray ($t0)** : Menyimpan nilai dari register $s0 ke dalam elemen pertama array.
- **addi $t0, $t0, 4** : Menambahkan index sebesar 4 untuk pindah ke elemen berikutnya.

- **sw $s1, myArray ($t0)** : Menyimpan nilai dari register $s1 ke dalam elemen kedua array.
- **addi $t0, $t0, 4** : Menambahkan index sebesar 4 untuk pindah ke elemen berikutnya.
- **sw $s2, myArray ($t0)** : Menyimpan nilai dari register $s2 ke dalam elemen ketiga array.
- **addi $t0, $zero, 0** : Menginisialisasi register $t0 (index) dengan nilai 0.
- **while:** : Label untuk loop while.
- **beq $t0, 12, exit** : Branch ke label exit jika nilai pada register $t0 sama dengan 12.
- **lw $t6, myArray($t0)** : Memuat nilai dari array pada indeks $t0 ke dalam register $t6.
- **add $t0, $t0, 4** : Menambahkan index sebesar 4.
- **li $v0, 1** : Mengatur sistem call untuk mencetak integer.
- **move $a0, $t6** : Memindahkan nilai yang akan dicetak ke dalam register $a0.
- **syscall** : Memanggil sistem call untuk mencetak nilai integer.
- **li $v0, 4** : Mengatur sistem call untuk mencetak string.
- **la $a0, newline** : Memuat alamat string newline ke dalam register $a0.
- **syscall** : Memanggil sistem call untuk mencetak newline.
- **j while** : Branch ke label while untuk mengulangi loop.

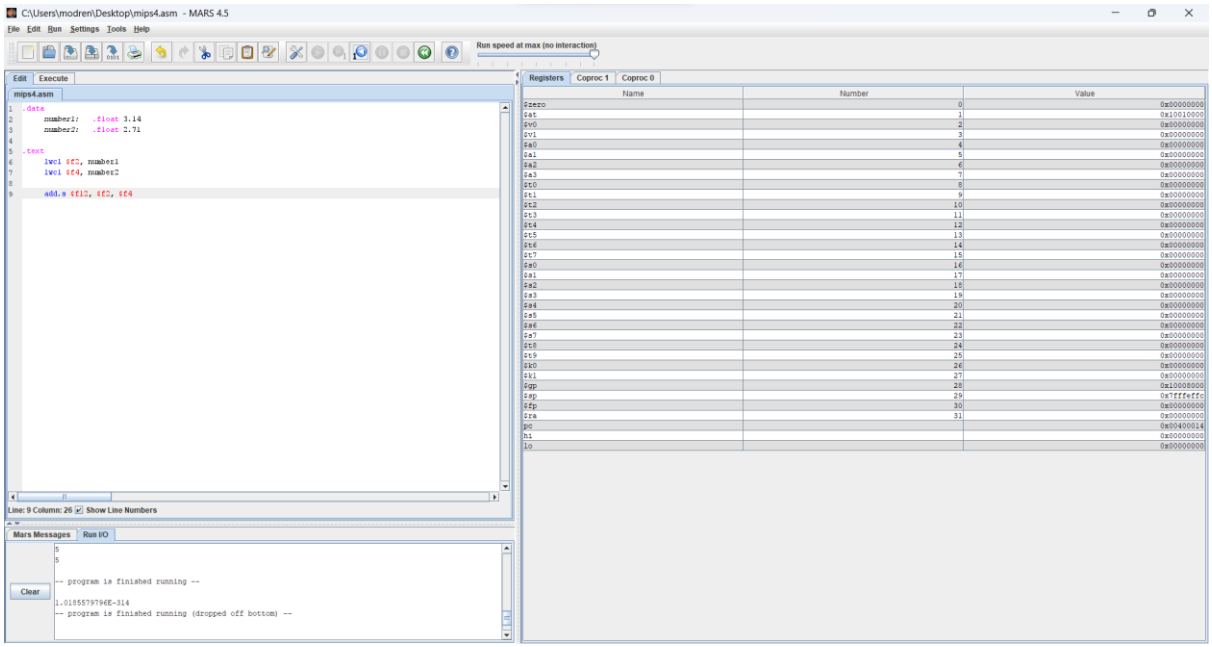
14.MIPS Tutorial 29 Array Initializer



- **.data** : menginisialisasi data yang akan digunakan dalam program.
- **myArray: .word 5:6** : Membuat array bernama myArray yang berisi 5 elemen, setiap elemen diinisialisasi dengan nilai 6.
- **newLine: .asciiz "\n"** : Membuat string bernama newLine yang berisi karakter newline ("\n").
- **.text** : Dimulainya instruksi program
- **main:** : Memulai bagian program utama.

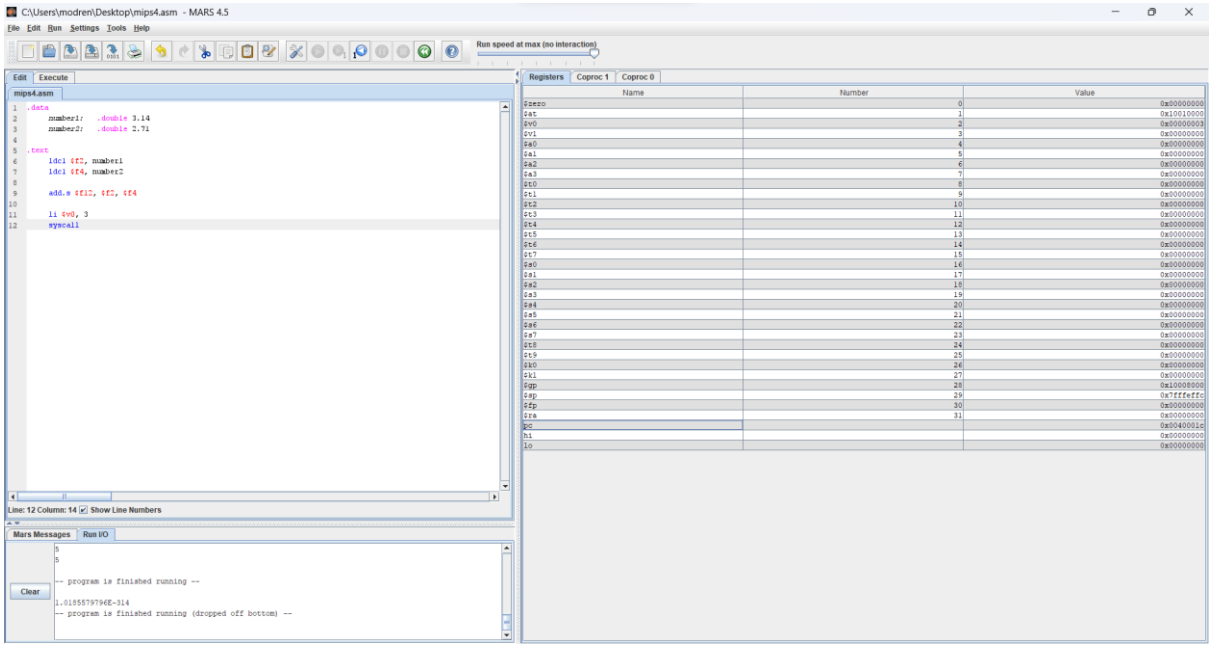
- **addi $t0, $zero, 0** : Menginisialisasi register $t0 (penghitung loop) dengan nilai 0.
- **while:** : Label awal dari loop.
- **beq $t0, 24, exit** : Melompat ke label exit jika nilai dalam register $t0 sama dengan 24.
- **lw $t6, myArray($t0)** : Memuat nilai dari array myArray pada indeks yang ditunjukkan oleh $t0 ke dalam register $t6.
- **add $t0, $t0, 4** : Menambahkan 4 ke nilai dalam register $t0 (menggeser ke indeks berikutnya dalam array).
- **li $v0, 1** : Mengatur sistem call untuk mencetak integer.
- **move $a0, $t6** : Menempatkan nilai yang akan dicetak (dalam register $t6) ke dalam register $a0.
- **syscall** : Memanggil sistem untuk mencetak nilai integer.
- **li $v0, 4** : Mengatur sistem call untuk mencetak string.
- **la $a0, newline** : Mengatur register $a0 dengan alamat string newLine.
- **syscall** : Memanggil sistem untuk mencetak karakter newline.
- **j while** : Melompat kembali ke label while untuk mengulangi loop.
- **exit:** : Label yang dituju jika loop selesai.
- **li $v0, 10** : Mengatur sistem call untuk mengakhiri program.
- **syscall** : Untuk mengakhiri program.

15. MIPS Tutorial 30 Floating Point Arithmetic



- **.data** : Menginisialisasi data yang akan digunakan dalam program.
- **number1: .float 3.14** **:** Mendeklarasikan variabel number1 sebagai float dan menginisialisasinya dengan nilai 3.14.
- **number2: .float 2.71** **:** Mendeklarasikan variabel number2 sebagai float dan menginisialisasinya dengan nilai 2.71.
- **.text** : Dimulainya instruksi program
- **lwc1 $f2, number1** : Memuat nilai dari alamat memori yang ditunjuk oleh number1 ke dalam register floating point $f2.

- **lwc1 $f4, number2** : Memuat nilai dari alamat memori yang ditunjuk oleh number2 ke dalam register floating point $f4.
- **add.s $f12, $f2, $f4** : Menambahkan nilai dalam register floating point $f2 dan $f4, dan menyimpan hasilnya dalam register floating point $f12.



- **.data** : Menginisialisasi data yang akan digunakan dalam program.
- **number1: .double 3.14** : Mendefinisikan variabel number1 dengan tipe data double dan memberinya nilai awal 3.14.
- **number2: .double 2.71** : Mendefinisikan variabel number2 dengan tipe data double dan memberinya nilai awal 2.71.
- **.text** : Dimulainya instruksi program
- **ldc1 $f2, number1** : Meng-Load double-precision floating-point value dari alamat memori yang diwakili oleh label number1 ke dalam register floating-point $f2.
- **ldc1 $f4, number2** : Meng-Load double-precision floating-point value dari alamat memori yang diwakili oleh label number2 ke dalam register floating-point $f4.
- **add.s $f12, $f2, $f4** : Menjumlahkan nilai yang terdapat di dalam register floating-point $f2 dan $f4, dan hasilnya disimpan dalam register floating-point $f12.
- **li $v0, 3** : Menginisialisasi sistem call untuk mencetak nilai floating-point ke layar.
- **syscall** : Memanggil sistem call yang sesuai dengan nilai yang sudah di-set sebelumnya di register $v0