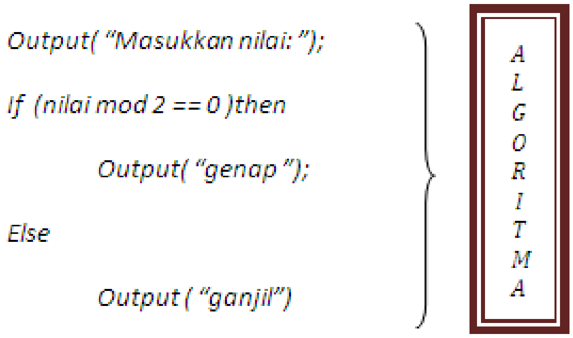
MODUL 6 : PCSPIM DAN BAHASA ASSEMBLY MIPS
(Bagian-2)

Tujuan KerjaLab:
1. Mengetahui dan memahami sintaks-sintaks SPIM 2
2. Mengimplementasikan bahasa assembly MIPS dalam studi kasus yang lebih mendalam

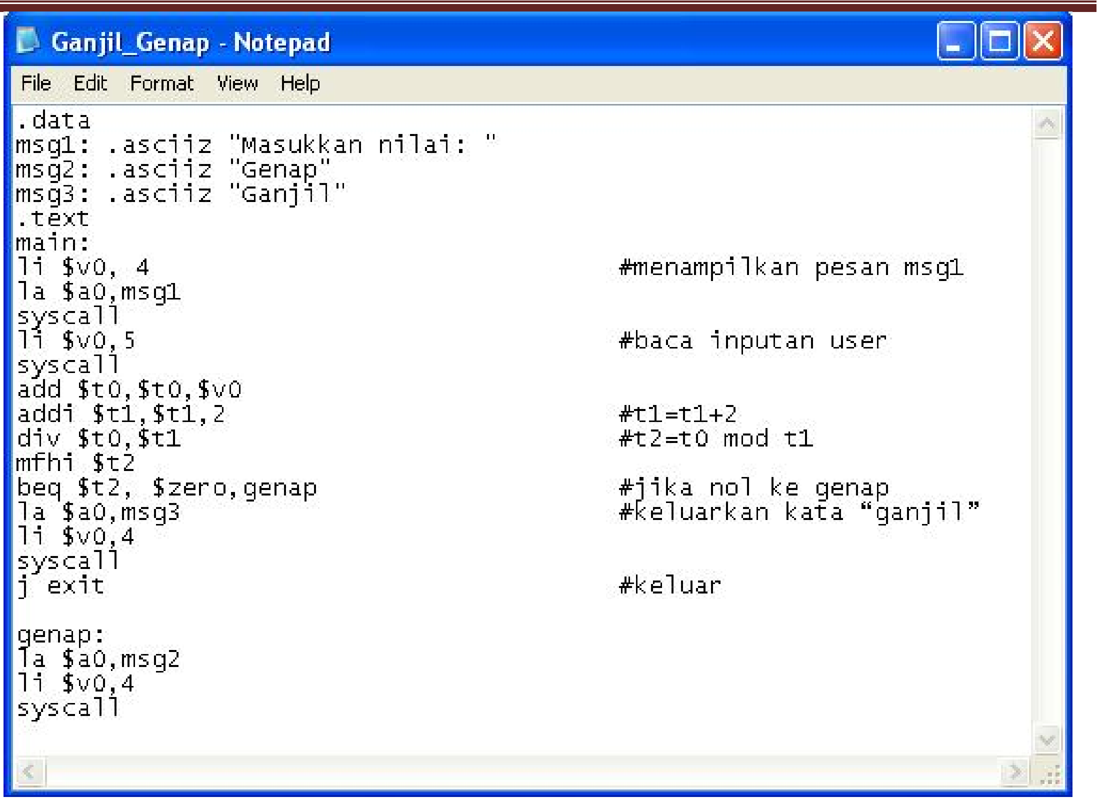
❖ Penggunaan SPIM dipakai pada saat kondisional (if …then …else…)

Kondisional kali ini akan diterapkan pada kasus pengecekan bilangan yang diinputkan user, apakah bilangan tersebut merupakan bilangan ganjil atau bilangan genap.



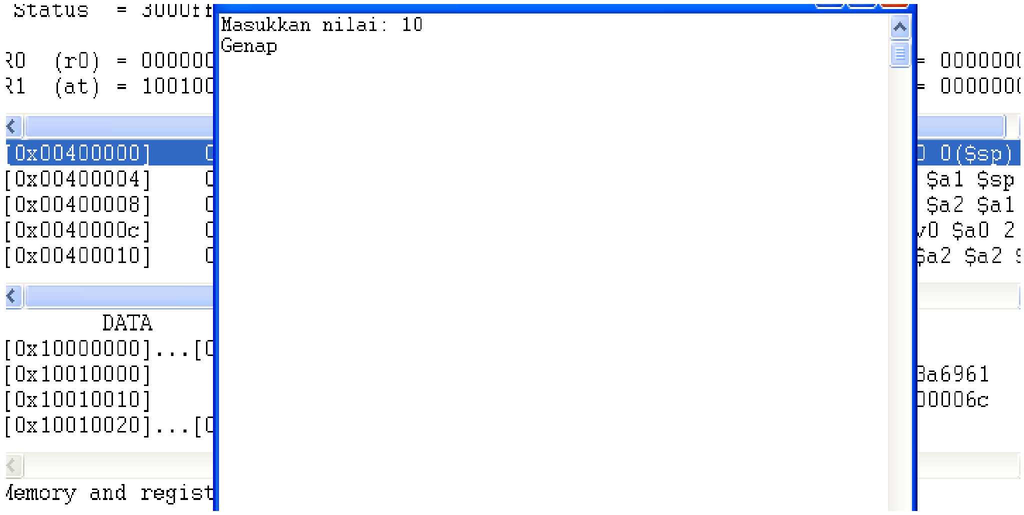
```
Output( "Masukkan nilai: ");
If (nilai mod 2 == 0 )then
        Output( "genap ");
Else
        Output ( "ganjil")
```

Bila algoritma diatas diubah kedalam bentuk SPIM maka hasilnya



```
.data
msg1: .asciiz "Masukkan nilai: "
msg2: .asciiz "Genap"
msg3: .asciiz "Ganjil"
.text
main:
li $v0, 4                              #menampilkan pesan msg1
la $a0,msg1
syscall
li $v0,5                               #baca inputan user
syscall
add $t0,$t0,$v0
addi $t1,$t1,2                         #t1=t1+2
div $t0,$t1                            #t2=t0 mod t1
mfhi $t2
beq $t2, $zero,genap                   #jika nol ke genap
la $a0,msg3                            #keluarkan kata “ganjil”
li $v0,4
syscall
j exit                                 #keluar

genap:
la $a0,msg2
li $v0,4
syscall
```

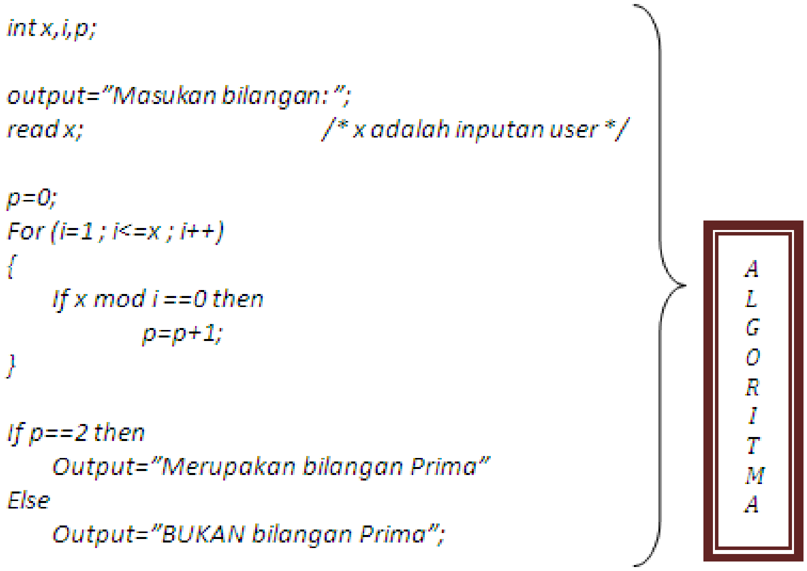
Printscreen:



```
Masukkan nilai: 10
Genap
```

- Penggunaan SPIM untuk bilangan Prima

Pada contoh kali ini kita akan mengecek apakah inputan user merupakan bilangan prima atau bukan, dan juga kita akan mengimplementasikan perulangan (for…) beserta kondisional (if…then…else).



```
int x,i,p;

output="Masukan bilangan:";
read x;                         /* x adalah inputan user */

p=0;
For (i=1 ; i<=x ; i++)
{
     If x mod i ==0 then
            p=p+1;
}

If p==2 then
     Output="Merupakan bilangan Prima"
Else
     Output="BUKAN bilangan Prima";
```

Bila diconvert menjadi SPIM syntax maka perlu kita ketahui bahwa,

t1 = x (inputan user);

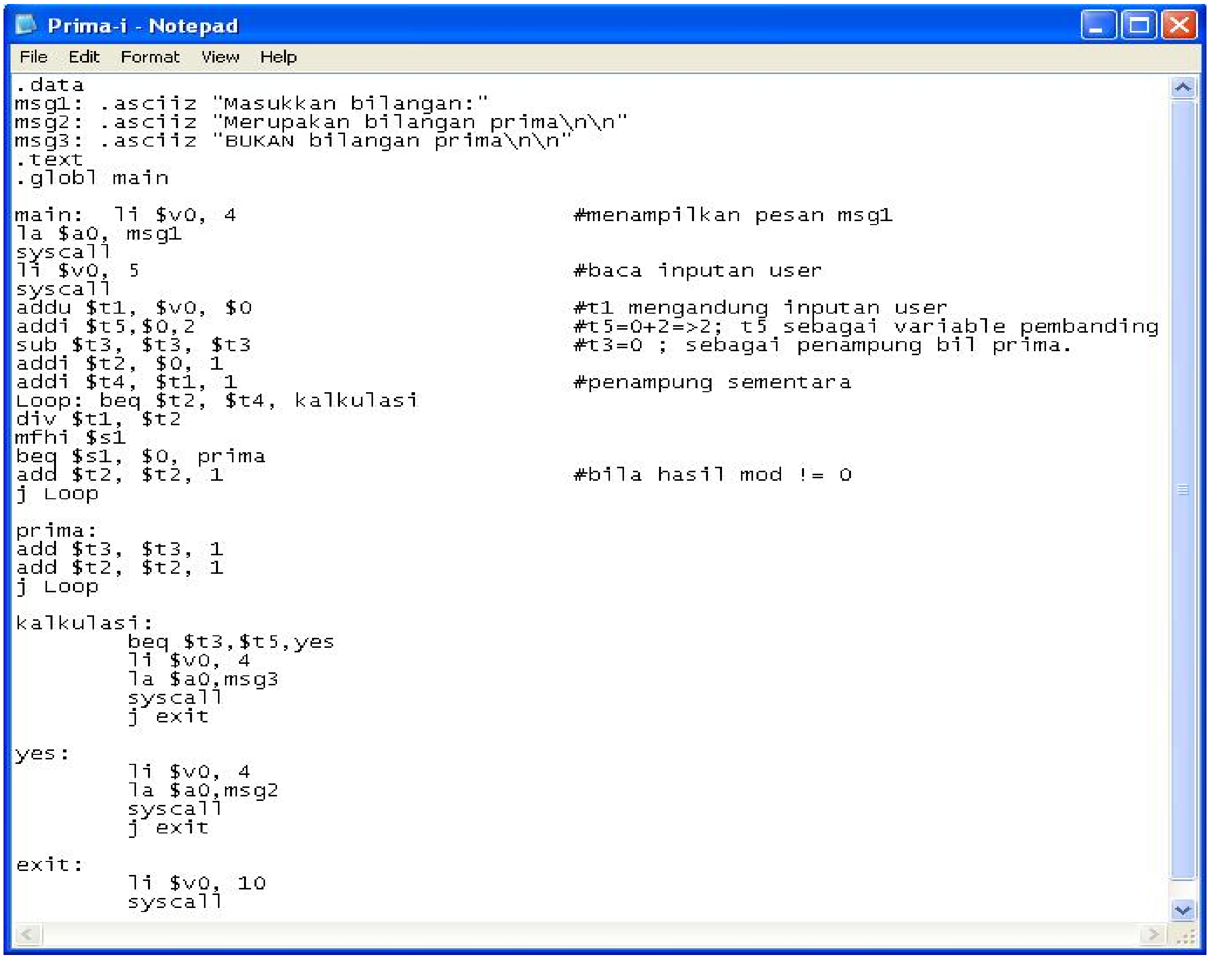
t2 = i;

t3 = p;

t4 = penampung nilai (x+1) ;

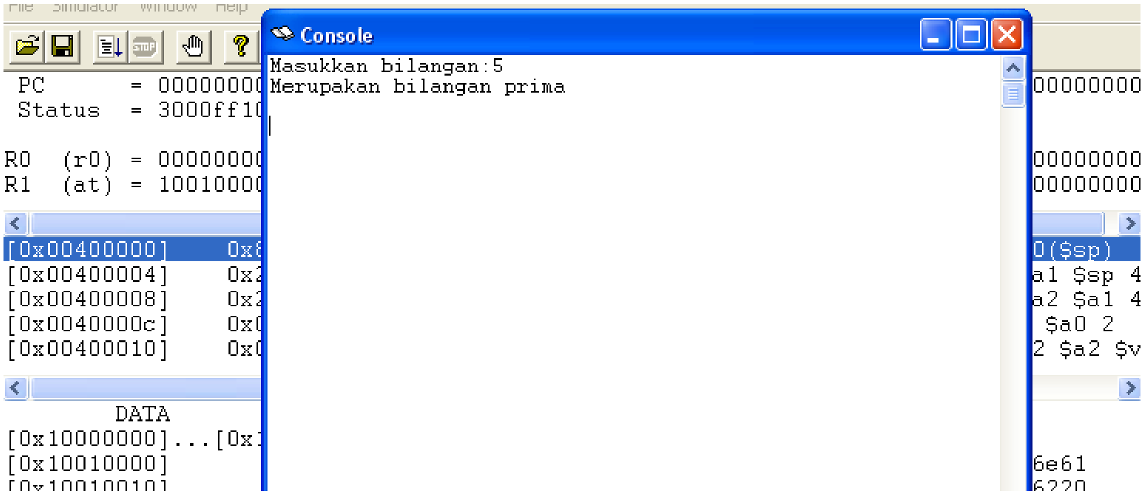
t5 = penampung bilangan 2;

s1= hasil dari perhitungan (x mod i)



```
.data
msg1: .asciiz "Masukkan bilangan:"
msg2: .asciiz "Merupakan bilangan prima\n\n"
msg3: .asciiz "BUKAN bilangan prima\n\n"
.text
.globl main

main:  li $v0, 4                        #menampilkan pesan msg1
la $a0, msg1
syscall
li $v0, 5                               #baca inputan user
syscall
addu $t1, $v0, $0                       #t1 mengandung inputan user
addi $t5,$0,2                           #t5=0+2=>2; t5 sebagai variable pembanding
sub $t3, $t3, $t3                       #t3=0 ; sebagai penampung bil prima.
addi $t2, $0, 1
addi $t4, $t1, 1                        #penampung sementara
Loop: beq $t2, $t4, kalkulasi
div $t1, $t2
mfhi $s1
beq $s1, $0, prima
add $t2, $t2, 1                         #bila hasil mod != 0
j Loop

prima:
add $t3, $t3, 1
add $t2, $t2, 1
j Loop

kalkulasi:
        beq $t3,$t5,yes
        li $v0, 4
        la $a0,msg3
        syscall
        j exit

yes:
        li $v0, 4
        la $a0,msg2
        syscall
        j exit

exit:
        li $v0, 10
        syscall
```

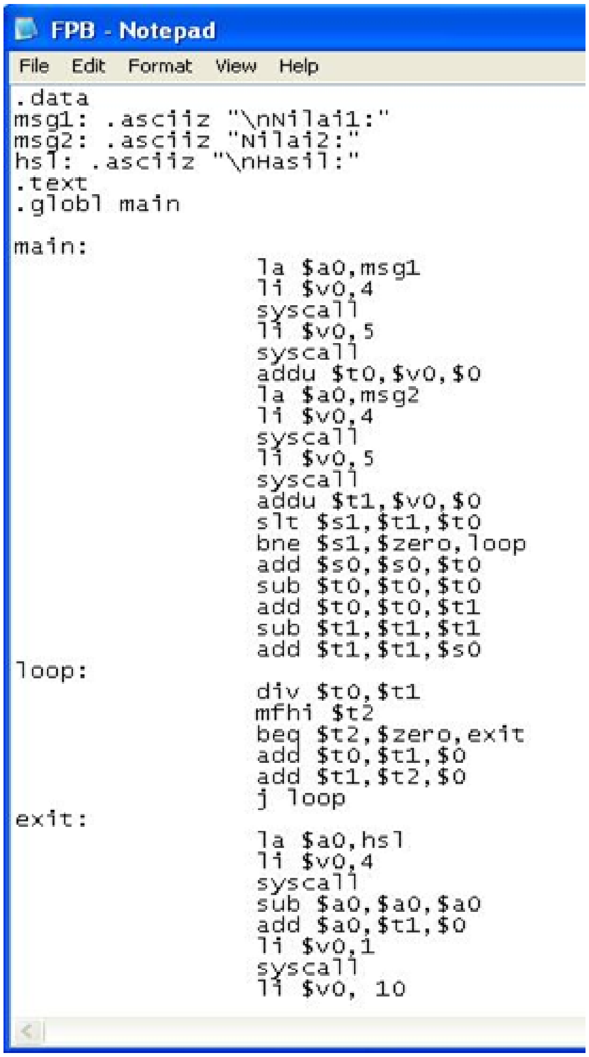
Printscreen:



```
Masukkan bilangan:5
Merupakan bilangan prima
```

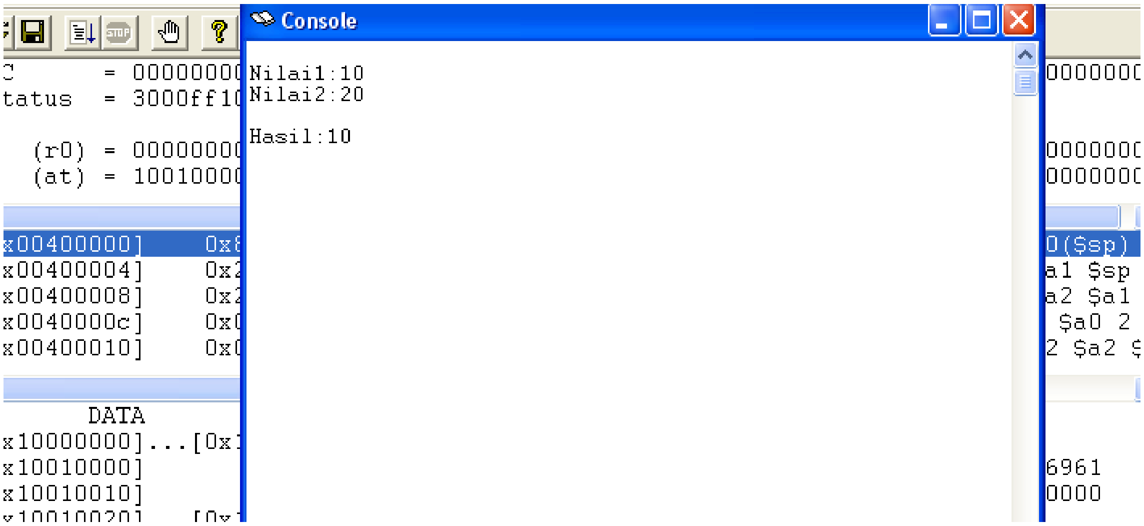
❖ Penggunaan SPIM untuk menghitung FPB

Pada contoh kali ini kita akan mengecek apakah inputan user merupakan bilangan FPB atau bukan.



```
.data
msg1: .asciiz "\nNilai1:"
msg2: .asciiz "Nilai2:"
hsl: .asciiz "\nHasil:"
.text
.globl main

main:
                la $a0,msg1
                li $v0,4
                syscall
                li $v0,5
                syscall
                addu $t0,$v0,$0
                la $a0,msg2
                li $v0,4
                syscall
                li $v0,5
                syscall
                addu $t1,$v0,$0
                slt $s1,$t1,$t0
                bne $s1,$zero,loop
                add $s0,$s0,$t0
                sub $t0,$t0,$t0
                add $t0,$t0,$t1
                sub $t1,$t1,$t1
                add $t1,$t1,$s0
loop:
                div $t0,$t1
                mfhi $t2
                beq $t2,$zero,exit
                add $t0,$t1,$0
                add $t1,$t2,$0
                j loop
exit:
                la $a0,hsl
                li $v0,4
                syscall
                sub $a0,$a0,$a0
                add $a0,$t1,$0
                li $v0,1
                syscall
                li $v0, 10
```

Printscreen:



```
Nilai1:10
Nilai2:20

Hasil:10
```

```
C         = 0000000C
tatus     = 3000ff10

  (r0)  = 0000000C
  (at)  = 1001000C

x00400000]     0x8
x00400004]     0x2
x00400008]     0x2
x0040000c]     0x0
x00400010]     0x0

      DATA
x10000000]...[0x1
x10010000]
x10010010]
```

```
0000000C

0000000C
0000000C

0($sp)
a1 $sp
a2 $a1
 $a0 2
2 $a2 $

6961
0000
```