

LAPORAN PRAKTIKUM 2 – SISTEM OPERASI

KH003



Disusun untuk Memenuhi Tugas Praktikum Sistem Operasi

Dosen Pengampu: Adi Widianto, S.Kom, M.Kom

Devina Dewi Faustina Basam

20220801250

PROGRAM STUDI TEKNIK INFORMATIKA

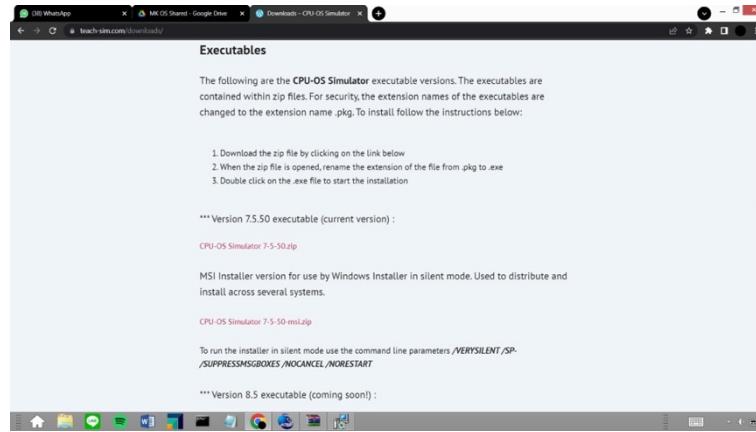
FAKULTAS ILMU KOMPUTER

UNIVERSITAS ESA UNGGUL

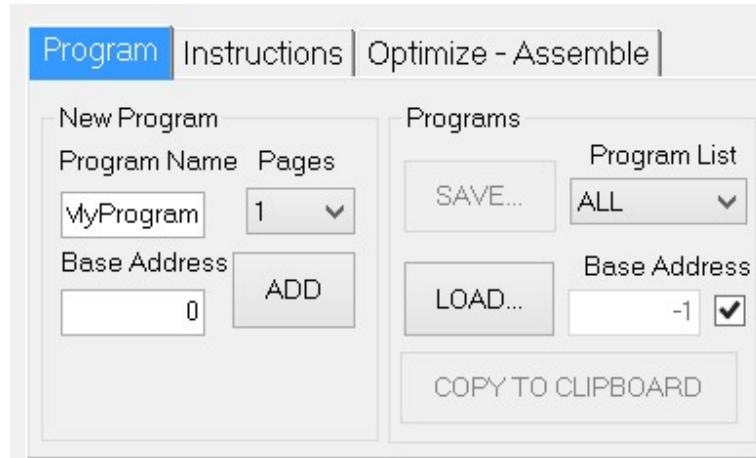
2023

PERTEMUAN 8

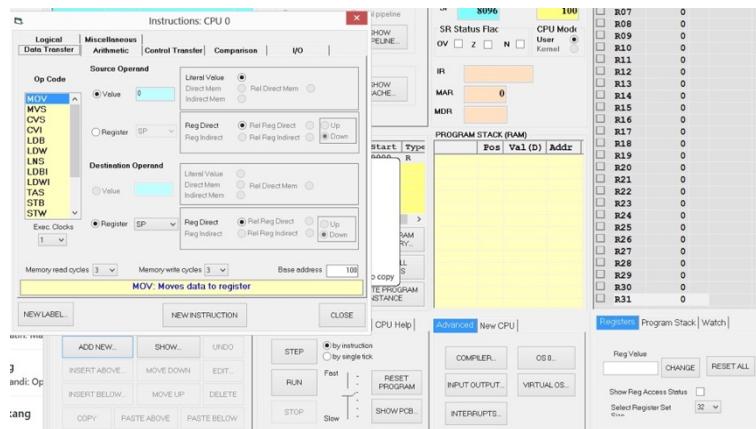
1. Download CPU OS SIMULATOR

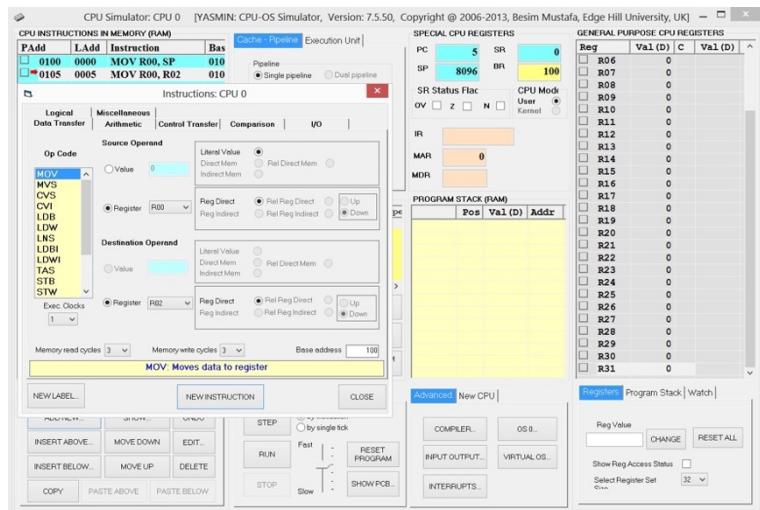


2. Membuat program baru



3. Membuat instruksi mov





PERTEMUAN 9

1. Membuat instruksi dan diakhiri HLT

The screenshot shows the 'Instructions: CPU 0' window. The 'Control Transfer' tab is selected. In the 'Op Code' dropdown, 'HLT' is selected. The 'Source Operand' section has 'Value' selected with a value of 'R00'. The 'Destination Operand' section also has 'Value' selected with a value of 'R00'. Below these, 'Exec. Clocks' is set to 1. At the bottom, a yellow bar displays the message 'HLT: Halts the simulator'. Buttons for 'NEW LABEL...', 'NEW INSTRUCTION', and 'CLOSE' are visible.

Below this window is the 'CPU Simulator: CPU 0' window, titled 'CPU INSTRUCTIONS IN MEMORY (RAM)'. It shows two entries:

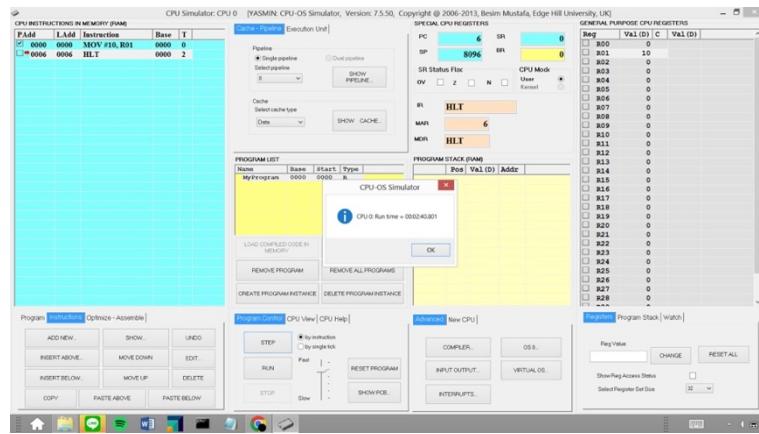
PAdd	LAdd	Instruction	Base	T
0000	0000	MOV #10, R01	0000	0
0006	0006	HLT	0000	2

2. cara menjalankan instruksi

Double klik

The screenshot shows the 'CPU Simulator: CPU 0' window. A modal dialog box in the center says 'CPU 0 Run time = 0001:54638' with an 'OK' button. The main interface shows the memory table and various control buttons like Step, Run, Stop, and Reset Program.

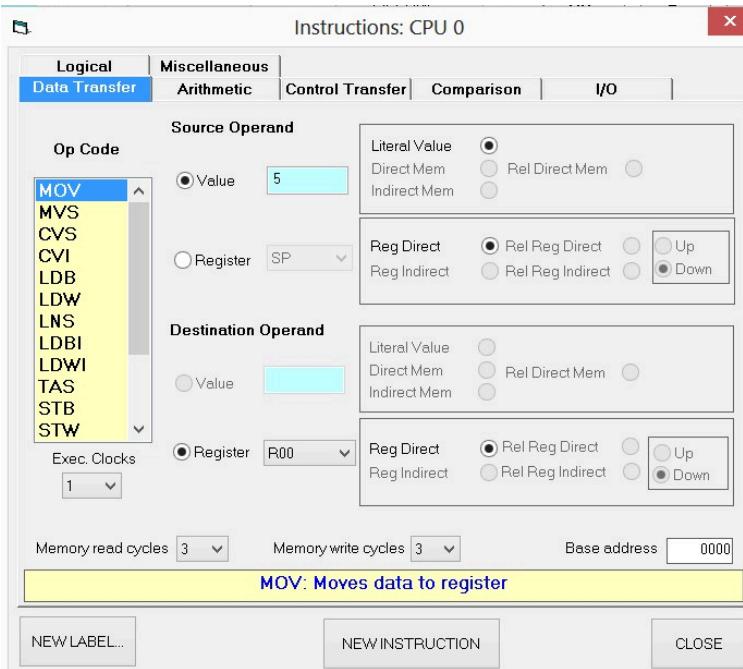
Step



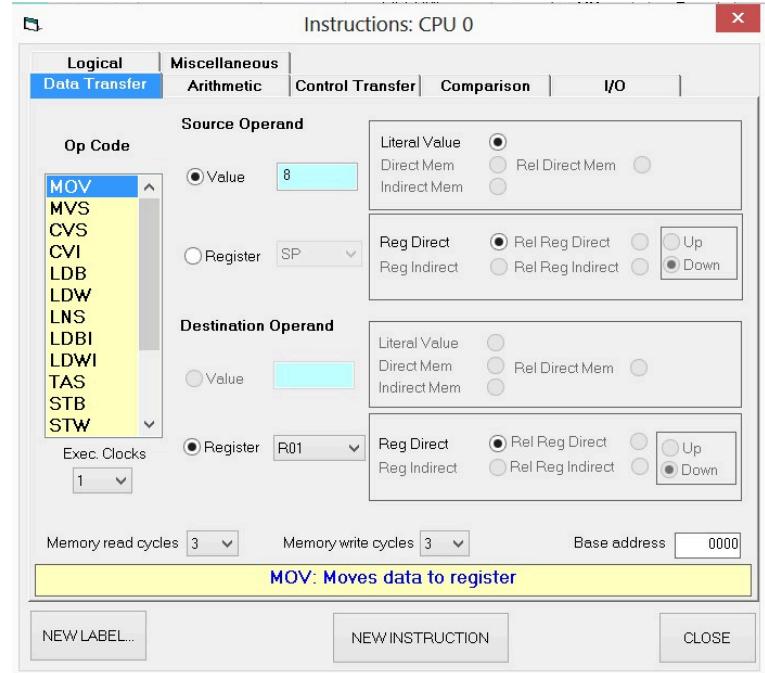
3. LATIHAN

a. Transfer data

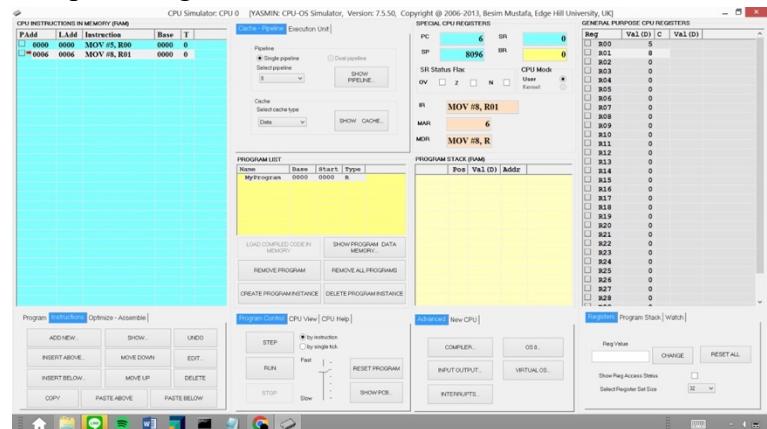
- Instruksi yang memindahkan (move) angka 5 ke register R00.



- Instruksi yang memindahkan (move) angka 8 ke register R01.

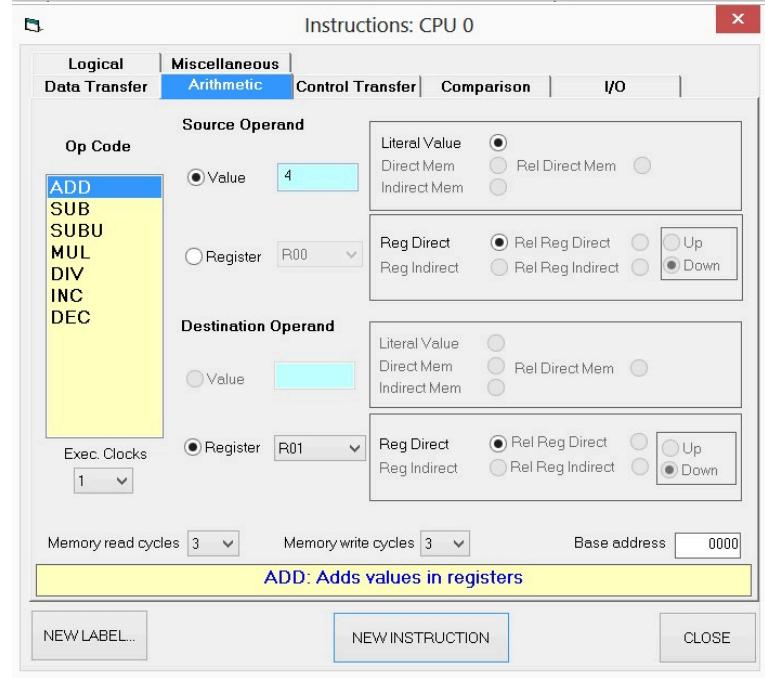
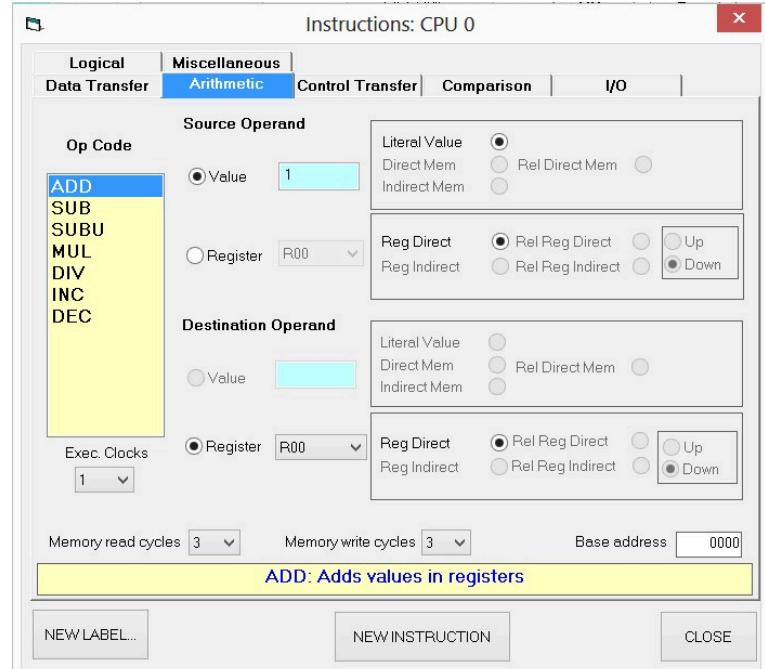


- Tampilan Register Set.

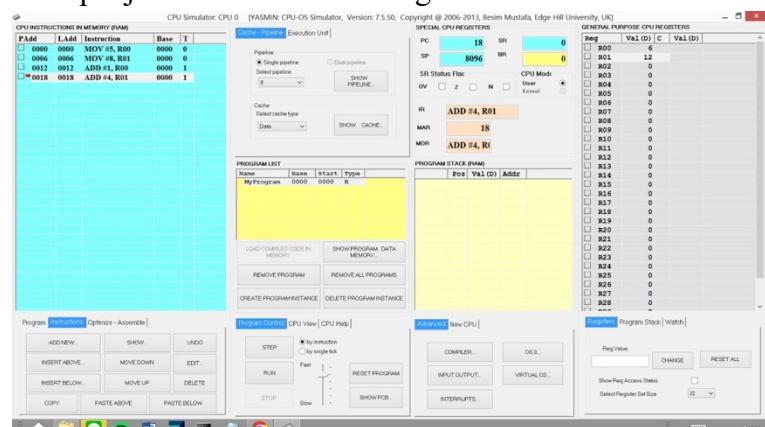


b. Aritmatika

- instruksi yang menambahkan (add) isi R00 dan R01

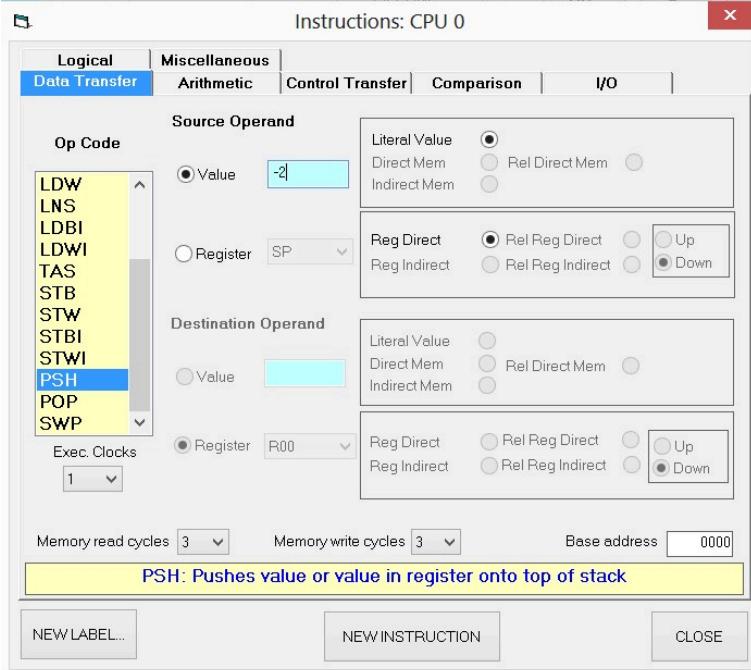


- hasil penjumlahan berada di Register Set

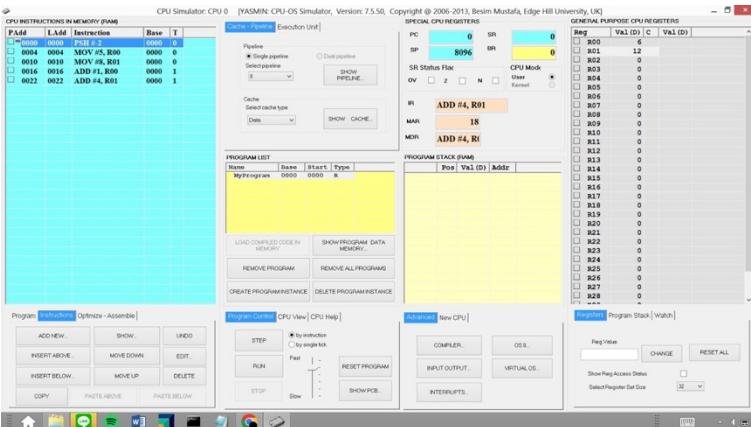


c. Stack Pointer (SP)

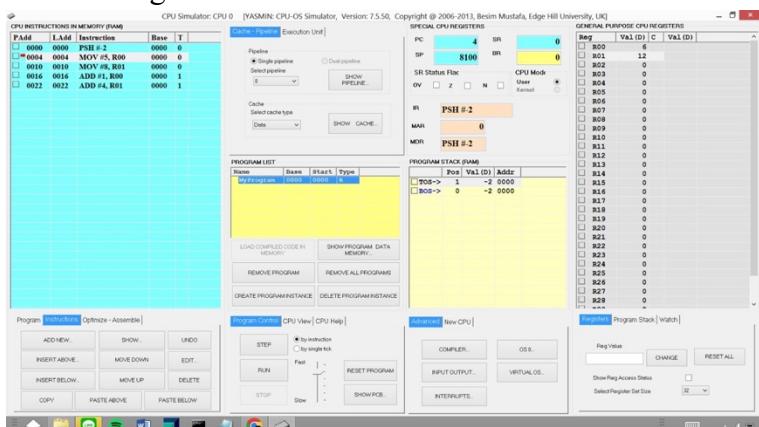
- Instruksi untuk menaruh (push) angka -2 pada stack teratas



- Tampilan Register Set.

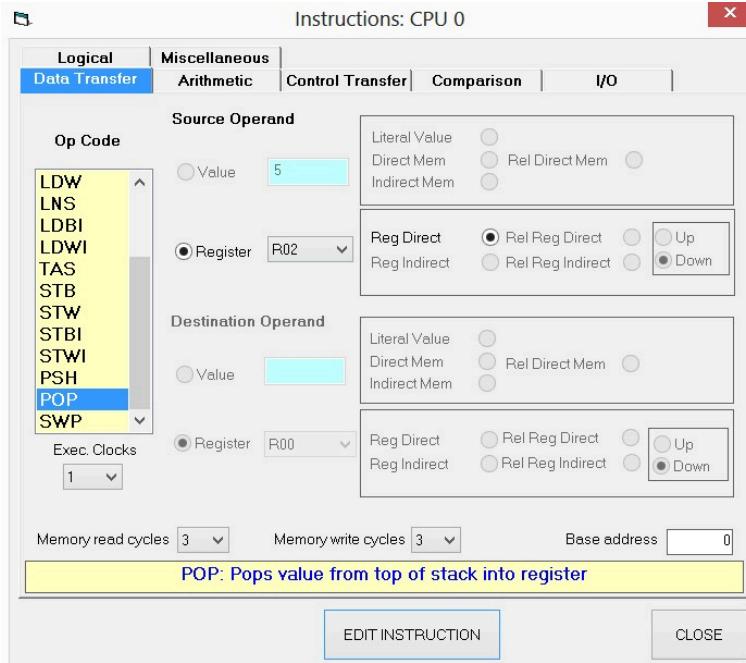


d. Pembanding

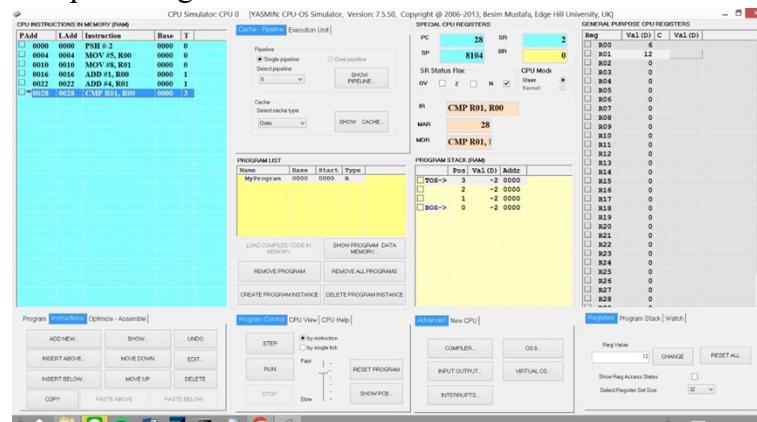


e. Stack Pointer (SP)

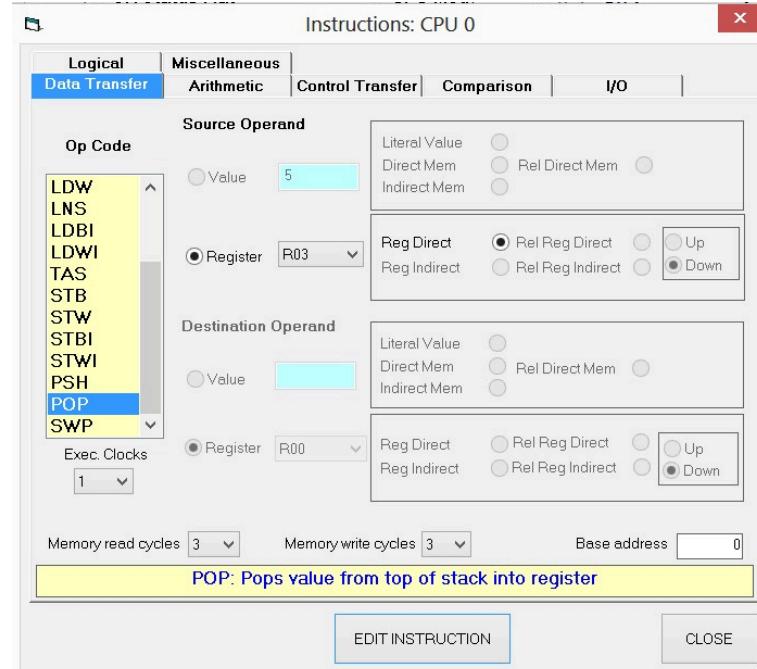
- Instruksi untuk mengambil (pop) nilai teratas dari program stack ke register R02.



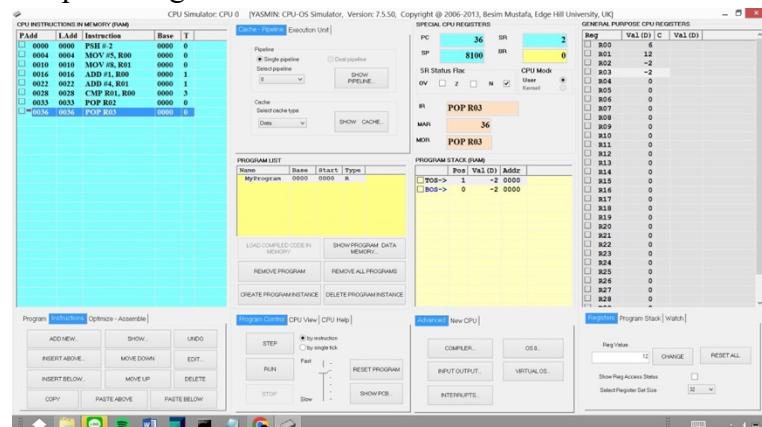
- Tampilan Register Set.



- Instruksi untuk mengambil (pop) nilai teratas dari program stack ke register R03.



- Tampilan Register Set.



PERTEMUAN 10

Menampilkan daftar proses/ thread dan pohon proses yang menggambarkan hubungan antar proses induk/anak :

1. Kode

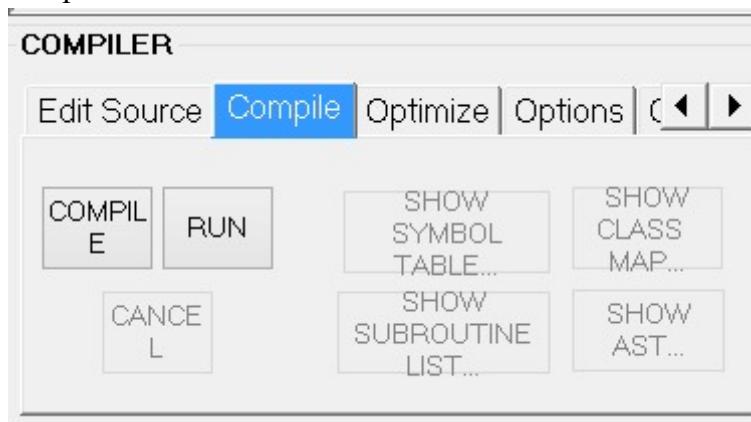
```
program ThreadTest1
sub thread1 as thread
writeln ("In thread1")
while true
wend
end sub

sub thread2 as thread
call thread1
writeln ("In thread2")
while true
wend
end sub

call thread2
<          >
call thread2
writeln ("In main")

do
loop
end
<          >
```

2. Compile dengan menekan tombol compile di kotak compiler pada tab compile



Program Compiler [YASMIN: CPU-OS Simulator, Version: 7.5.50, Copyright © 2006-2013, Besim Mustafa, Edge Hill University, UK]

PROGRAM SOURCE (INPUT)

```
program Threadtest1
sub thread1 as thread
writeln ("In thread1")
while true
wend
end sub

sub thread2 as thread
call thread1
writeln ("In thread2")
while true
wend
end sub

call thread2
writeln ("In main")
c
```

COMPILER MESSAGE

```
1....Found DO statement [1]
1....Found keyword LOOP [1]
0!Found keyword END [2]
Code generation completed...
Display completed...
Display completed...
Display completed...
*** NOTE: Click on numbers in brackets to highlight corresponding source line
```

ASSEMBLY CODE (OUTPUT)

Line	CPU Instruction	Binary Code	Comments
0000	**** NUB THREAD...	0000	
0001	PSH #1	0000FF	0009 Thread priority
0002	SWI 0	0E000000	0009 Thread code address
0003	SWI 5	2E020005	0015 Enter thread code
0004	MOV #22,R02	0000000060102	0016 Copy the value of 22 to _STempRe...
0005	MOV R02,0	0000000060102	0016 Copy the value of R02 to memory location
0006	OUT #16,1	3E000000A020...	0016 Output literal value
0007	MOV #1,R01	0000000010102	0011 Copy the value of 1 to R01
0008	MOV R01,R02	0000000010102	0011 Copy the value of R01 to R02 if r01=...
0009	JMP 68	1D020044	0012 Jump to code at address 68 (UNCODE)
0002	SWI 1	2E020005	0015 Terminate thread process THREAD...
0015	**** MAIN PROG...	0000	
0016	PSH #1	0000FF	0015 Thread priority
0017	SWI 0	0E000000	0015 Thread code address
0018	SWI 5	2E020005	0015 Enter thread code
0019	MOV #38,R02	0000000060102	0016 Copy the value of 38 to _STempRe...
0020	MOV R02,0	0000000060102	0016 Copy the value of R02 to memory location
0019	OUT #16,1	3E000000A020...	0016 Output literal value
0021	JMP 117	1D020075	0019 Jump to code at address 117 (UNCODE)
0121	HLT	2F	0020 Stop simulation
0001	0!Thread	49E2B0474677...	
0002	0!Thread2	49E2B0474677...	
0003	0!main	49E2B0474616...	

COMPILE

Edit Source | Compile | Optimize | Options | Compiler Help |

SOURCE LOAD SAVE PRINT

ASSEMBLY CODE

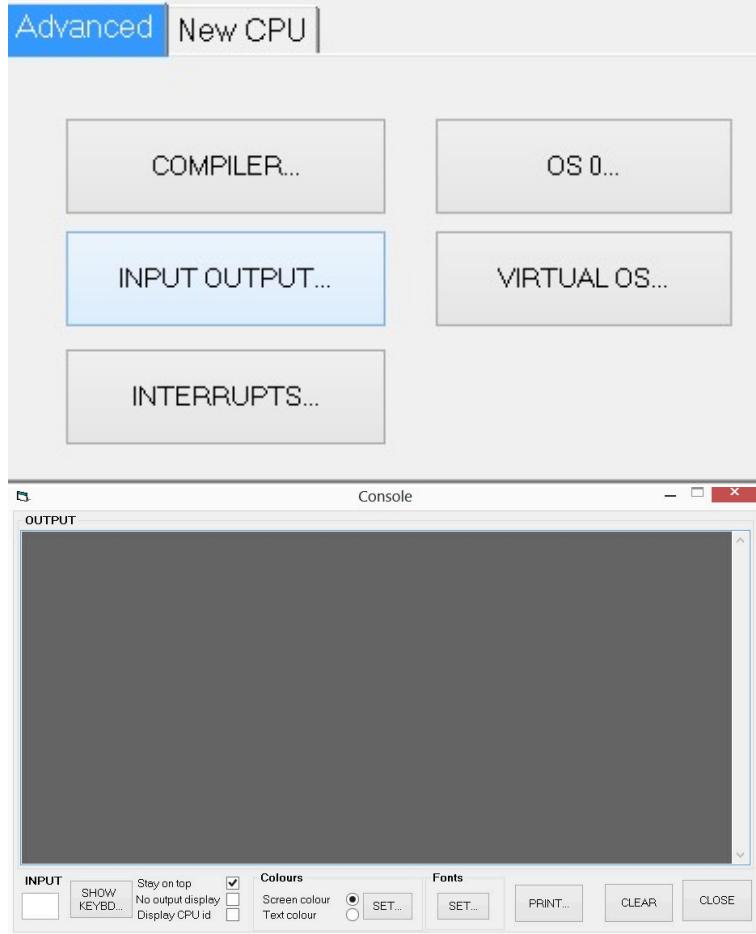
Start Address: 00000000 | Code Size: 122 | Input: Include source | LOAD SHOW

LOAD IN: MEMORY | Select CPU: 0 | Base Address: | SAVE Use compiler options | CLOSE

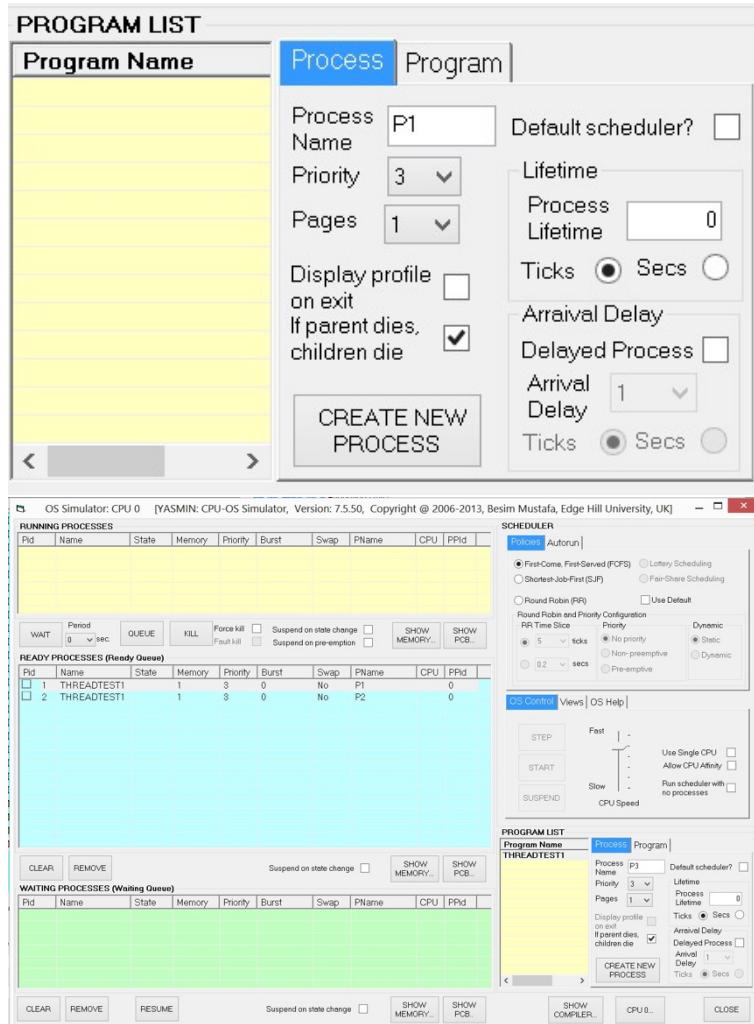
SHOW SYMBOL TABLE | SHOW CLASS MAP | SHOW ASST | SHOW SUBROUTINE LIST | SHOW AST

Cancel | Show New CPU... | Auto save source files

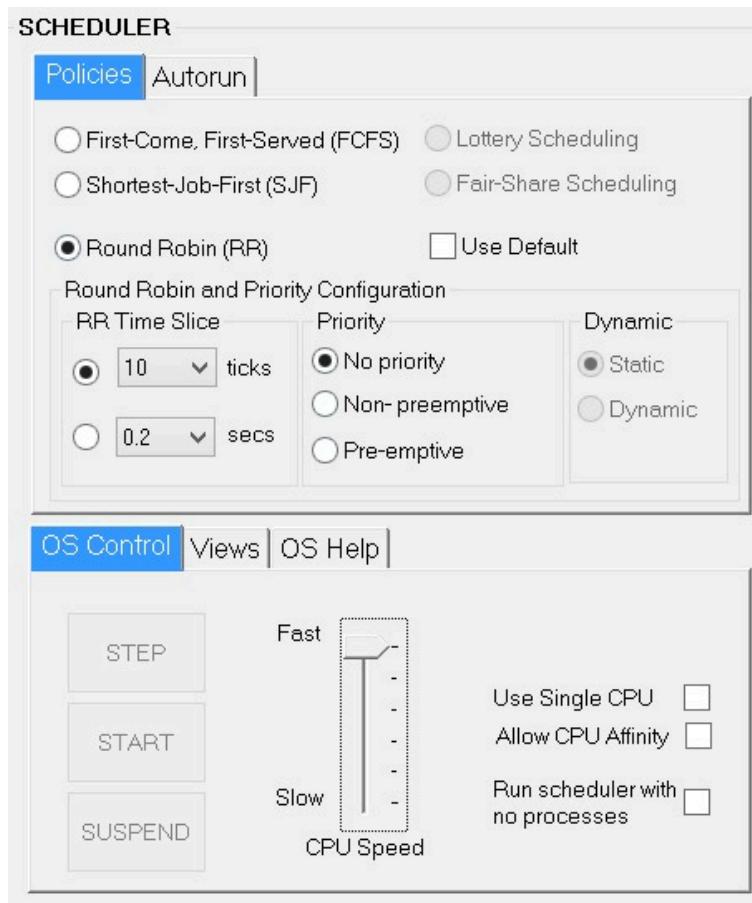
3. Klik tombol input/output... Aktifkan stay on top



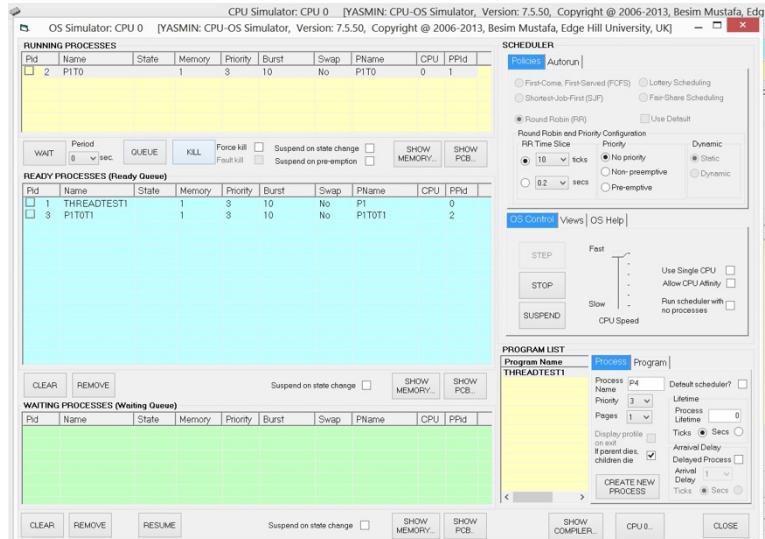
4. Buat satu proses dengan create new process.



5. Jenis penjadwalan adalah round robin, 10 ticks dengan kecepatan simulasi maksimum.



6. Tampilan pada jendela console.



7. Jumlah proses yang dibuat
8. Proses nya yang
9. Thread nya yang
10. Klik tombol VIEW PROCESS LIST. Kemudian klik PROCESS TREE..

Process List

PID	Name	PPID	Pri	CPUT	STA	MEM	CPU	ETM
1	THREADTEST1	0	3	402	Ready	1	0	2105
2	P1T0	1	3	390	Ready	1	0	2105
3	P1TOT1	2	3	380	Running	1	0	2105

Stay on top

PROCESS TREE...

PCB...

PROFILE...

CLOSE

Process List

PID	Name	PPID	Pri	CPUT	STA	MEM	CPU	ETM
1	THREADTEST1	0	3	462	Running	1	0	2121
2	P1T0	1	3	460	Ready	1	0	2121
3	P1TOT1	2	3	450	Ready	1	0	2121

Stay on top

PROCESS TREE...

PCB...

PROFILE...

CLOSE

Process List

PID	Name	PPID	Pri	CPUT	STA	MEM	CPU	ETM
1	THREADTEST1	0	3	562	Ready	1	0	2146
2	P1T0	1	3	550	Running	1	0	2146
3	P1TOT1	2	3	550	Ready	1	0	2146

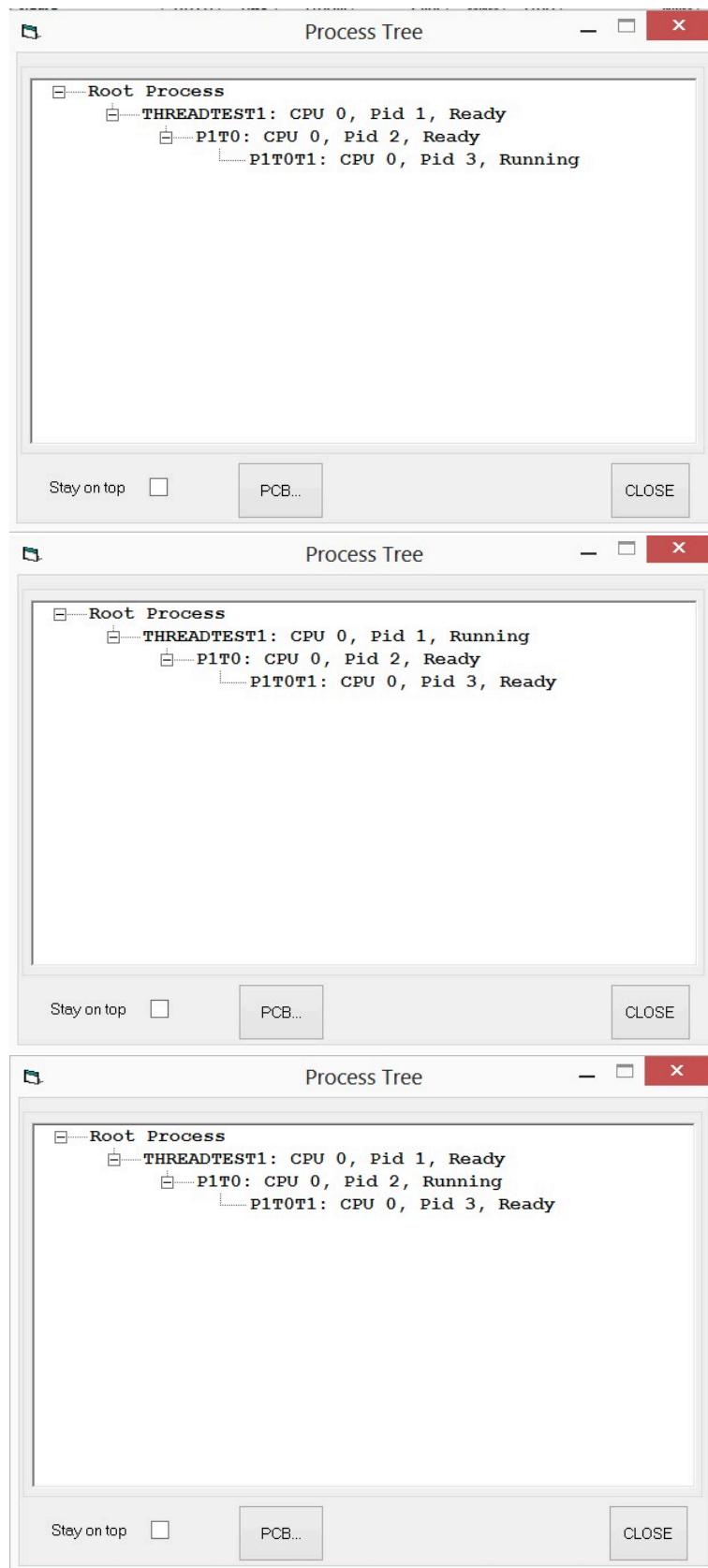
Stay on top

PROCESS TREE...

PCB...

PROFILE...

CLOSE



11. Identifikasi proses induk dan proses turunan!
12. tombol KILL untuk menghentikan proses!

Memodifikasi kode sumber untuk membuat versi tanpa thread dan membandingkannya dengan versi yang menggunakan thread.

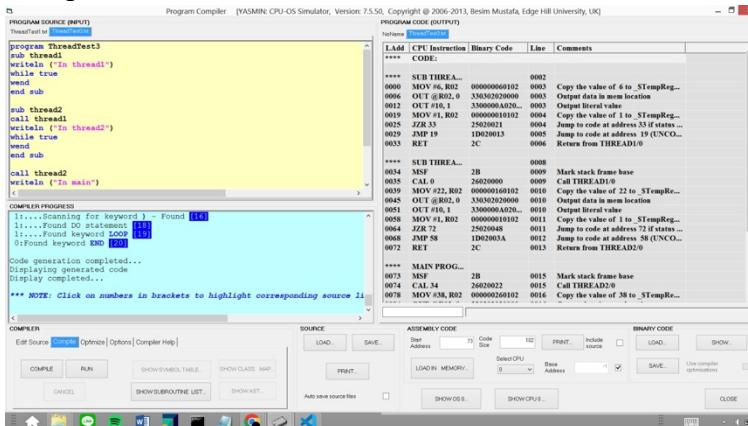
1. Modifikasi source code ThreadTest1 dengan menghapus instance “as thread” pada deklarasi subroutine. Ganti nama program menjadi ThreadTest3.

```
program ThreadTest3
sub thread1
writeln ("In thread1")
while true
wend
end sub

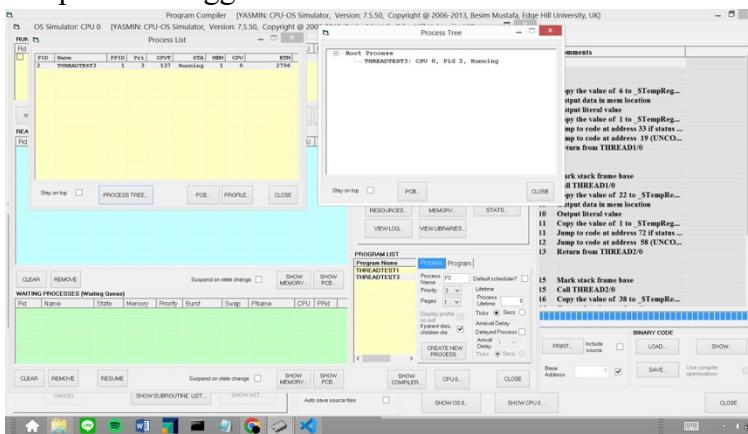
sub thread2
call thread1
writeln ("In thread2")
while true
wend
end sub

call thread2
writeln ("In main")
```

2. Compile code hasil modifikasi tersebut.



3. Buat proses baru lagi, kemudian jalankan di OS simulator.
4. Amati tampilan pada jendela console, apa perbedaan proses tanpa thread dan proses menggunakan thread?



Memahami bagaimana thread-thread berbagi sumber daya dari proses induk.

1. Kode

```

program ThreadTest2
var s1 string(6)
var s2 string(6)

sub thread1 as thread
s1 = "hello1"
writeln ("In thread1")
while true
wend
end sub

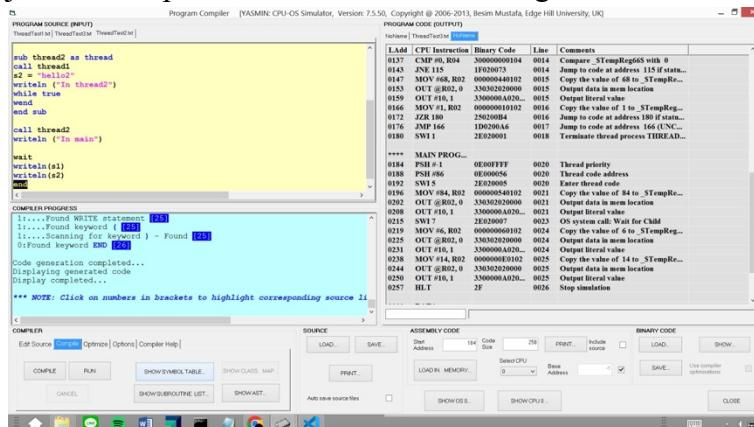
sub thread2 as thread
call thread1
s2 = "hello2"
writeln ("In thread2")
while true
wend
end sub

call thread2
writeln ("In main")

wait
writeln(s1)
writeln(s2)
end

```

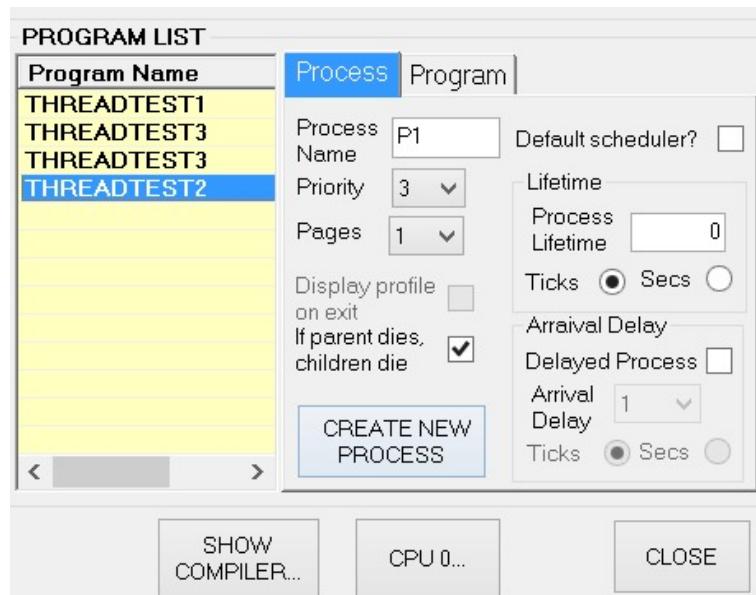
2. Compile program ThreadTest2. Klik tombol SYMBOL TABLE pada jendela compiler. Amati informasi tentang variabel s1 dan s2.



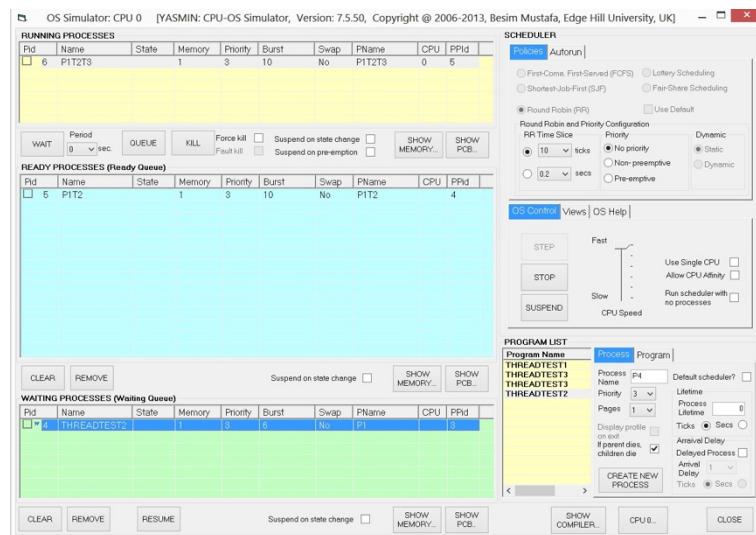
3. Melalui CPU Simulator, tampilkan jendela konsol dengan klik tombol INPUT/OUTPUT... aktifkan STAY ON TOP.



4. Beralihlah ke jendela OS Simulator, buat satu proses dengan CREATE NEW PROCESS.



5. Pastikan jenis penjadwalan adalah ROUND ROBIN, kecepatan simulasi maksimum.



PERTEMUAN 11

A. Algoritma penjadwalan FCFS CPU

The screenshot shows the Dev-C++ IDE interface. The top window displays the source code for the FCFS CPU scheduling algorithm. The bottom window shows the execution results.

Source Code (PENJADWALAN FCFS CPU.cpp):

```
1 #include<iostream>
2 #include<conio.h>
3 #include<conio.h>
4 int main()
5 {
6     int bt[20], wt[20], tat[20], i, n;
7     float wtavg, tatavg;
8     system("cls");
9     cout<<"Enter the number of processes -- ";
10    scanf("%d", &n);
11    for(i=0; i<n; i++)
12    {
13        cout<<"Enter Burst Time for Process "<<i<<" -- ";
14        scanf("%d", &bt[i]);
15        wt[i] = 0;
16        tat[i] = 0;
17        for(k=i+1; k<n; k++)
18        {
19            bt[k] = bt[k]-bt[i];
20            if(bt[k] >= 0)
21            {
22                wt[k] = wt[k]+wt[i];
23                tat[k] = tat[k]+bt[i];
24            }
25        }
26    }
27    printf("\nPROCESS BURST TIME & WAITING TIME & TURNAROUND TIME\n");
28    for(i=0; i<n; i++)
29    {
30        cout<<"Process "<<i<<" BT "<<bt[i]<<" WT "<<wt[i]<<" TA "<<tat[i]<<endl;
31    }
32    printf("\nAverage Waiting Time -- %f", wtavg/n);
33    printf("\nAverage Turnaround Time -- %f", tatavg/n);
34    getch();
35 }
```

Execution Output:

```
Enter the number of processes -- 5
Enter Burst Time for Process 0 -- 10
Enter Burst Time for Process 1 -- 11
Enter Burst Time for Process 2 -- 04
Enter Burst Time for Process 3 -- 29
Enter Burst Time for Process 4 -- 07
      PROCESS      BURST TIME      WAITING TIME      TURNAROUND TIME
      P0            10              0                  10
      P1            11              10                 21
      P2             4               21                 25
      P3            29              25                 54
      P4             7               54                 61
Average Waiting Time -- 22.000000
Average Turnaround Time -- 34.200001
```

B. Algoritma penjadwalan SJF CPU

The screenshot shows the Dev-C++ IDE interface. The top window displays the SJF CPU scheduling algorithm code. The bottom window shows the execution results.

Source Code (PENJADWALAN SJF CPU.cpp):

```
1 #include<iostream>
2 #include<conio.h>
3 #include<conio.h>
4 int main()
5 {
6     int bt[20], wt[20], tat[20], i, k, n, temp;
7     float wtavg, tatavg;
8     system("cls");
9     cout<<"Enter the number of processes -- ";
10    scanf("%d", &n);
11    for(i=0; i<n; i++)
12    {
13        cout<<"Enter Burst Time for Process "<<i<<" -- ";
14        scanf("%d", &bt[i]);
15    }
16    for(i=0; i<n; i++)
17    {
18        for(j=i+1; j<n; j++)
19        {
20            if(bt[i]>bt[j])
21            {
22                bt[i] = bt[i]+bt[j];
23                bt[j] = temp;
24                temp = bt[i];
25                bt[i] = temp;
26            }
27        }
28    }
29    wt[0] = wtavg = 0;
30    tat[0] = tatavg = bt[0];
31    for(i=1; i<n; i++)
32    {
33        bt[i] = bt[i]-bt[i-1];
34        if(bt[i] >= 0)
35        {
36            wt[i] = wt[i-1]+wt[i-1];
37            tat[i] = tat[i-1]+bt[i];
38            wtavg = wtavg + wt[i];
39            tatavg = tatavg + tat[i];
40        }
41    }
42    printf("\nPROCESS BURST TIME & WAITING TIME & TURNAROUND TIME\n");
43    for(i=0; i<n; i++)
44    {
45        cout<<"Process "<<i<<" BT "<<bt[i]<<" WT "<<wt[i]<<" TA "<<tat[i]<<endl;
46    }
47    printf("\nAverage Waiting Time -- %f", wtavg/n);
48    printf("\nAverage Turnaround Time -- %f", tatavg/n);
49    getch();
50 }
```

Execution Output:

```
Enter the number of processes -- 5
Enter Burst Time for Process 0 -- 10
Enter Burst Time for Process 1 -- 11
Enter Burst Time for Process 2 -- 04
Enter Burst Time for Process 3 -- 29
Enter Burst Time for Process 4 -- 07
      PROCESS      BURST TIME      WAITING TIME      TURNAROUND TIME
      P0            10              0                  10
      P1            11              10                 21
      P2             4               21                 25
      P3            29              25                 54
      P4             7               54                 61
Average Waiting Time -- 22.000000
Average Turnaround Time -- 34.200001
```

File Edit Search View Project Execute Tools About Window Help

E:\PENJADWALAN SJF CPU.cpp - [Executing] - Dev-C++ 5.11

```

1 //PENJADWALAN SJF CPU.cpp PENJADWALAN FCFS CPU.cpp PENJADWALAN ROUND ROBIN CPU.cpp PENJADWALAN SJF CPU.cpp
2
3 #include<iostream.h>
4 #include<conio.h>
5 #include<math.h>
6
7 using namespace std;
8
9 main(){
10    int n,i,burst[5],wtavg=0,tat[5],t,ct[5],max;
11    float avg,attavg,temp;
12
13    system("cls");
14    cout<<"Enter the no of processes -- ";
15    scanf("%d",&n);
16
17    for(i=0;i<n;i++){
18        cout<<"Enter Burst Time for Process Id -- ";
19        cin>>burst[i];
20        temp=burst[i];
21        wtavg = wtavg + burst[i];
22        tat[i]=temp;
23        ct[i]=0;
24    }
25
26    for(i=0;i<n;i++){
27        for(j=i+1;j<n;j++){
28            if(burst[j]<burst[i]){
29                temp=burst[i];
30                burst[i]=burst[j];
31                burst[j]=temp;
32                wtavg = wtavg + burst[i];
33                tat[j]=tat[i]+burst[i];
34            }
35        }
36    }
37
38    cout<<"\nEnter PROCESSES (BURST TIME < WAITING TIME< TURNAROUND TIME)\n";
39    for(i=0;i<n;i++){
40        cout<<"\nProcess " << i+1 << " Bt " << burst[i] << " Wt " << wtavg << " Tt " << tat[i];
41    }
42
43    cout<<"\nAverage Waiting Time -- " << wtavg/n;
44    cout<<"\nAverage Turnaround Time -- " << wtavg/n;
45    getch();
46}

```

File Edit Search View Project Execute Tools About Window Help

E:\PENJADWALAN SJF CPU.exe

Enter the number of processes -- 5
Enter Burst Time for Process 0 -- 10
Enter Burst Time for Process 1 -- 11
Enter Burst Time for Process 2 -- 04
Enter Burst Time for Process 3 -- 29
Enter Burst Time for Process 4 -- 07

PROCESS	BURST TIME	WAITING TIME	TURNAROUND TIME
P2	4	0	4
P4	7	4	11
P0	10	11	21
P1	11	21	32
P3	29	32	61

Average Waiting Time -- 13.600000
Average Turnaround Time -- 25.799999

C. Algoritma penjadwalan ROUND ROBIN CPU

File Edit Search View Project Execute Tools About Window Help

E:\PENJADWALAN ROUND ROBIN CPU.cpp - Dev-C++ 5.11

```

1 //PENJADWALAN PRIORITY CPU.cpp PENJADWALAN FCFS CPU.cpp PENJADWALAN ROUND ROBIN CPU.cpp PENJADWALAN SJF CPU.cpp
2
3 #include<iostream.h>
4 #include<conio.h>
5 #include<math.h>
6
7 using namespace std;
8
9 main(){
10    int n,i,burst[5],wtavg=0,tat[5],t,ct[5],max;
11    float avg,attavg,temp;
12
13    system("cls");
14    cout<<"Enter the no of processes -- ";
15    scanf("%d",&n);
16
17    for(i=0;i<n;i++){
18        cout<<"Enter Burst Time for process Id -- ";
19        cin>>burst[i];
20        ct[i]=burst[i];
21    }
22
23    print("Enter the size of time slice -- ");
24    cin>>t;
25
26    max=burst[0];
27    for(i=1;i<n;i++){
28        if(burst[i]>max){
29            max=burst[i];
30        }
31    }
32
33    for(i=0;i<n;i++){
34        for(j=0;j<t;j++){
35            if(burst[i]==0){
36                break;
37            }
38            if(burst[i]>t){
39                temp=burst[i]-t;
40                burst[i]=t;
41                burst[j]=temp;
42            }
43            else {
44                burst[i]=burst[i]-t;
45                temp=t;
46            }
47        }
48    }
49
50    for(i=0;i<n;i++){
51        if(max>burst[i]){
52            max=burst[i];
53        }
54    }
55
56    for(i=0;i<n;i++){
57        for(j=0;j<max;j++){
58            if(burst[i]==0){
59                break;
60            }
61            if(burst[i]>t){
62                temp=burst[i]-t;
63                burst[i]=t;
64                burst[j]=temp;
65            }
66            else {
67                burst[i]=burst[i]-t;
68                temp=t;
69            }
70        }
71    }
72
73    for(i=0;i<n;i++){
74        wtavg=wtavg+burst[i];
75        tat[i]=burst[i]+ct[i];
76        att=att+ct[i];
77    }
78
79    cout<<"\nEnter PROCESSES (BURST TIME < WAITING TIME< TURNAROUND TIME)\n";
80    for(i=0;i<n;i++){
81        cout<<"\nProcess " << i+1 << " Bt " << burst[i] << " Wt " << wtavg << " Tt " << tat[i];
82    }
83
84    cout<<"\nAverage Waiting Time is -- " << wtavg/n;
85    cout<<"\nThe Average Turnaround time is-- " << att/n;
86    getch();
87}

```

File Edit Search View Project Execute Tools About Window Help

E:\PENJADWALAN ROUND ROBIN CPU.cpp - Dev-C++ 5.11

```

E:\PENJADWALAN ROUND ROBIN CPU.exe
Enter the no of processes -- 5
Enter Burst Time for process 1 -- 10
Enter Burst Time for process 2 -- 11
Enter Burst Time for process 3 -- 04
Enter Burst Time for process 4 -- 29
Enter Burst Time for process 5 -- 07
Enter the size of time slice -- 8

      PROCESS    BURST TIME    WAITING TIME    TURNAROUND TIME
      1          10              27                37
      2          11              29                40
      3          4               16                20
      4          29              32                61
      5          7               28                35

The Average Waiting time is -- 26.400000
The Average Turnaround time is-- 38.599998

```

D. Algoritma penjadwalan PRIORITY CPU

```

E:\PENJADWALAN PRIORITY CPU.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools ASYS Window Help
E:\PENJADWALAN PRIORITY CPU.cpp PENJADWALAN FCFS CPU.cpp PENJADWALAN ROUND ROBIN CPU.cpp PENJADWALAN SJF CPU.cpp
Project 1 #include<iostream>
2 #include<conio.h>
3 #include<math.h>
4 float max[5];
5 int bt[5], pr[5], wt[5], tat[5], i, k, n, temp;
6 float wtavg, tatavg;
7 system("cls");
8 cout<<"Masukkan jumlah proses --- ";
9 scanf("%d",&n);
10 for(i=0;i<n;i++)
11 {
12     cout<<"Masukkan Burst Time & Prioritas Proses Id --- ";
13     cin>>bt[i]>>pr[i];
14 }
15 for(i=0;i<n;i++)
16 {
17     for(j=i+1;j<n;j++)
18     {
19         if(pr[i]>pr[j])
20         {
21             temp=bt[i];
22             bt[i]=bt[j];
23             bt[j]=temp;
24
25             temp=pr[i];
26             pr[i]=pr[j];
27             pr[j]=temp;
28         }
29     }
30 }
31 wtavg = wt[0] = 0;
32 tatavg = tat[0] = bt[0];
33 for(i=1;i<n;i++)
34 {
35     wt[i] = wt[i-1] + bt[i];
36     tat[i] = tat[i-1] + bt[i];
37     wtavg = wtavg + wt[i];
38     tatavg = tatavg + tat[i];
39 }
40
41 printf("Process ID(PRIORITY)\tBURST TIME\tWAITING TIME\tTURNAROUND TIME");
42 for(i=0;i<n;i++)
43 {
44     cout<<pr[i]<<bt[i]<<wt[i]<<tat[i];
45 }
46 cout<<"\n";
47 cout<<"Average waiting Time is --- %.4f",wtavg/n;
48 cout<<"\n";
49 cout<<"Average Turnaround Time is --- %.4f",tatavg/n;
50 getch();
51 
```

```
E:\PENJADWALAN PRIORITY CPU.exe
Masukkan jumlah proses --- 5
Masukkan Burst Time & Prioritas Proses 0 --- 10 3
Masukkan Burst Time & Prioritas Proses 1 --- 11 2
Masukkan Burst Time & Prioritas Proses 2 --- 04 7
Masukkan Burst Time & Prioritas Proses 3 --- 29 1
Masukkan Burst Time & Prioritas Proses 4 --- 07 5

PROCESS      PRIORITY      BURST TIME      WAITING TIME      TURNAROUND TIME
3            1              29                0                  29
1            2              11                29                 40
0            3              10                40                 50
4            5              7                 50                 57
2            7              4                 57                 61

Rata-rata Waiting Time is --- 35.200001
Rata-rata Turnaround Time is --- 47.400002
```

Kesimpulannya adalah algoritma SJF merupakan algoritma dengan waktu tunggu dan waktu putar nya paling cepat dan algoritma PRIORITY adalah yang paling lambat (dalam percobaan diatas). Akan berbeda hasil jika algoritma ROUND ROBIN dan PRIORITY diubah size time of slice dan urutan prioritas nya.

PERTEMUAN 12

1. Proses diwaktu yang bersamaan

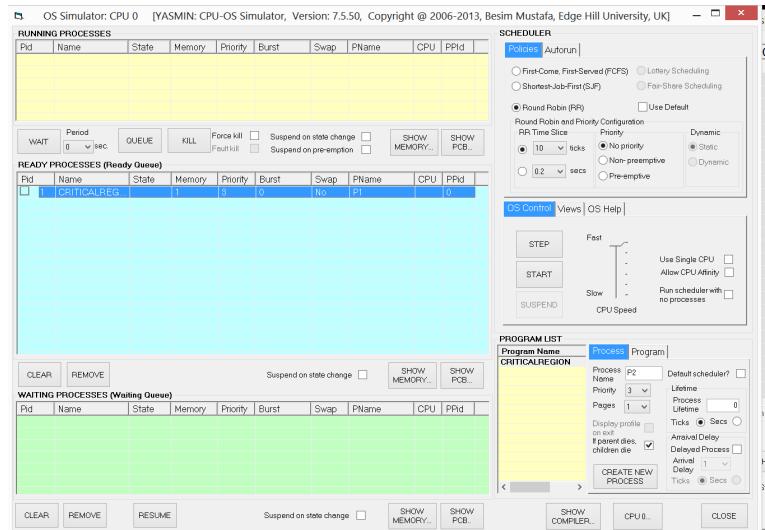
- Kode program

```
program CriticalRegion
    var g integer
    sub thread1 as thread
        writeln("In thread1")
        g = 0
        for n = 1 to 20
            g = g + 1
        next
        writeln("thread1 g = ", g)
        writeln("Exiting thread1")
    end sub
    sub thread2 as thread
        writeln("In thread2")
        g = 0
        for n = 1 to 12
            g = g + 1
        next
        writeln("thread2 g = ", g)
        writeln("Exiting thread2")
    end sub
    writeln("In main")
    call thread1
    call thread2
    wait
    writeln("Exiting main")
end
```

- Mengaktifkan STAY ON TOP



- Di OS simulator buat proses dengan ROUND ROBIN, 10 ticks, kecepatan maksimum.



- Output di jendela console

```
In main
Exiting main
In main
In thread1
In thread2
thrad2 g = 18
Exiting thread2
thrad1 g = 25
Exiting thread1
Exiting main
```

2. Proses dengan synchronise

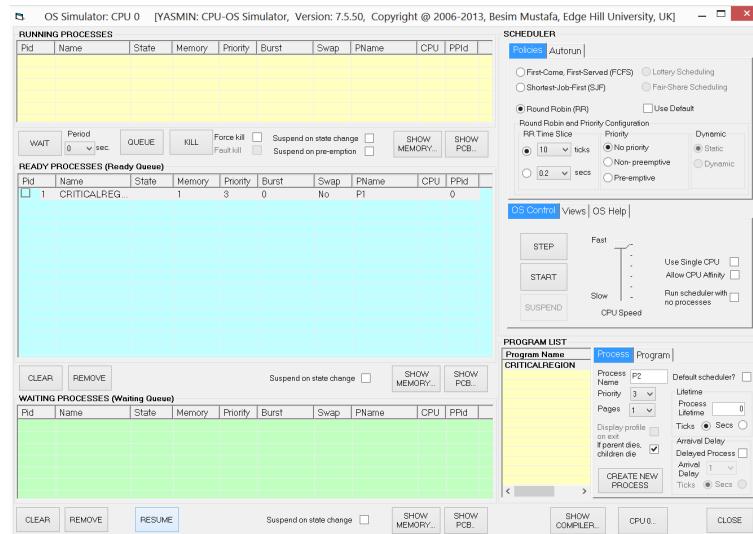
- Kode program

```
program CriticalRegion
    var g integer
    sub thread1 as thread synchronise
        writeln("In thread1")
        g = 0
        for n = 1 to 20
            g = g + 1
        next
        writeln("thrad1 g = ", g)
        writeln("Exiting thread1")
    end sub
    sub thread2 as thread synchronise
        writeln("In thread2")
        g = 0
        for n = 1 to 12
            g = g + 1
        next
        writeln("thrad2 g = ", g)
        writeln("Exiting thread2")
    end sub
    writeln("In main")
    call thread1
    call thread2
    wait
    writeln("Exiting main")
end
```

- Baris di program code (output) yang menerjemahkan synchronise.

LAdd	CPU Instruction	Binary Code	Line	Comments
****	SUB THREA...		0003	
0000	TAS 0, R01	090200000101	0003	Test and set
0006	CMP #0, R01	300000000101	0003	Compare _STempReg94\$ with 0
0012	JEQ 24	1E020018	0003	Jump to code at address 24 if the sta...
0016	PSH #0	0E000000	0003	Push thread start address
0020	SWI 10	2E02000A	0003	OS system call: Blocked Wait

- Di OS simulator buat proses dengan ROUND ROBIN, 10 ticks, kecepatan maksimum.



- Output di jendela console

```
In main
Exiting main
In main
Exiting main
In main
In thread1
thrad1 g = 20
Exiting thread1
In thread2
thrad2 g = 12
Exiting thread2
Exiting main
```

3. Proses dengan enter dan leave

- Kode program

```
program CriticalRegion
    var g integer
    sub thread1 as thread
        writeln("In thread1")
        enter
        g = 0
        for n = 1 to 20
            g = g + 1
        next
        writeln("thrad1 g = ", g)
        leave
        writeln("Exiting thread1")
    end sub
    sub thread2 as thread
        writeln("In thread2")
    end sub
```

```

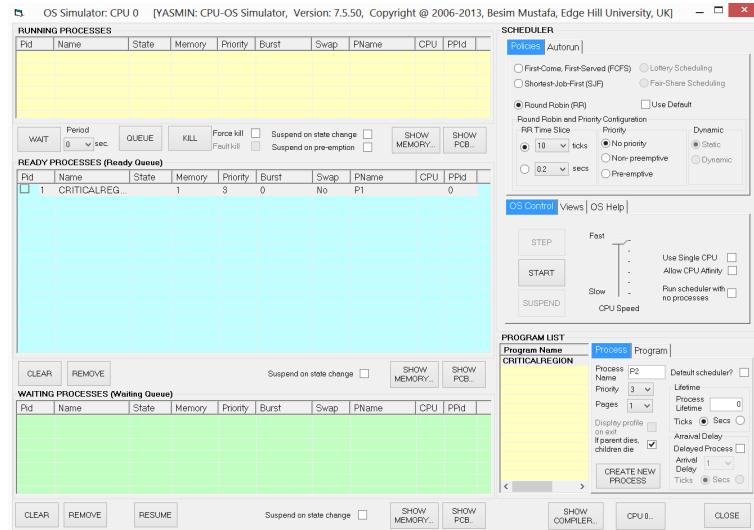
    enter
    g = 0
    for n = 1 to 12
        g = g + 1
    next
    writeln("thrad2 g = ", g)
    leave
    writeln("Exiting thread2")
end sub
writeln("In main")
call thread1
call thread2
wait
writeln("Exiting main")
end

```

- Baris di program code (output) yang menerjemahkan enter dan leave.

0019	SWI 18	2E020012	0005	OS system call: Enter critical region
0130	SWI 19	2E020013	0011	OS system call: Exit critical region

- Di OS simulator buat proses dengan ROUND ROBIN, 10 ticks, kecepatan maksimum.



- Output di jendela console

```

In main
Exiting main
In main
In thread1
In thread2
thradi g = 20
Exiting thread1
thrad2 g = 12
Exiting thread2
Exiting main

```

4. Deadlock

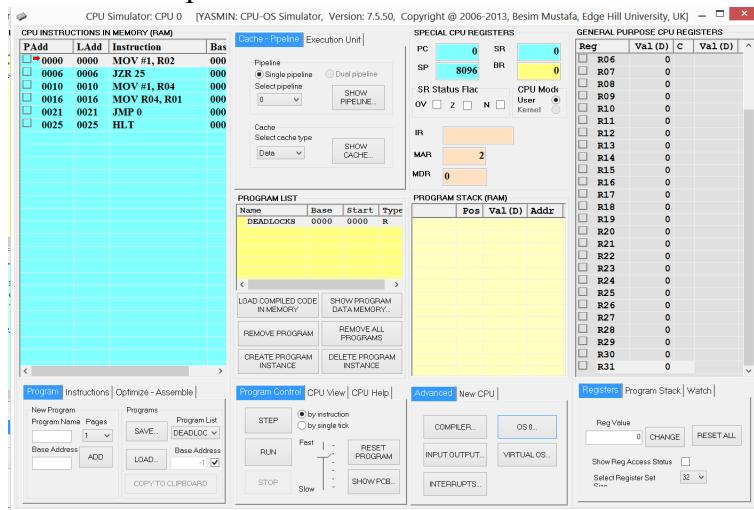
- Kode program

```

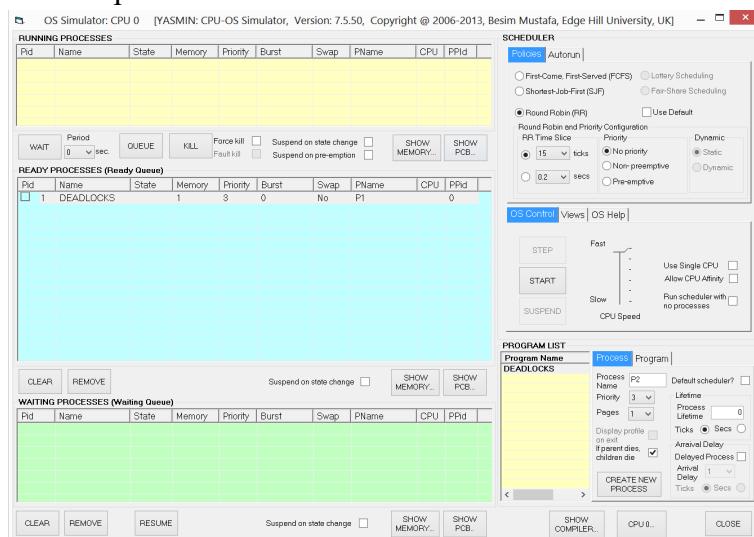
Program Deadlocks
    while true
        n = 1
    wend
end

```

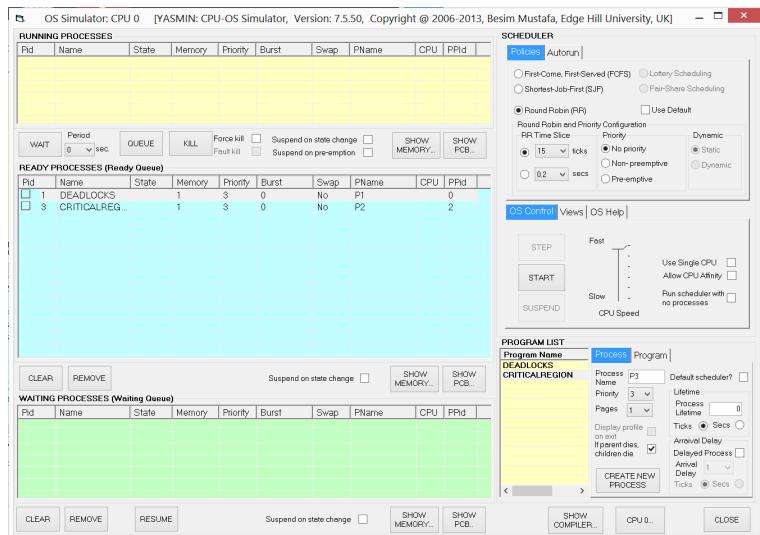
- Hasil compile



- Di OS simulator buat proses dengan ROUND ROBIN, 10 ticks, kecepatan maksimum.



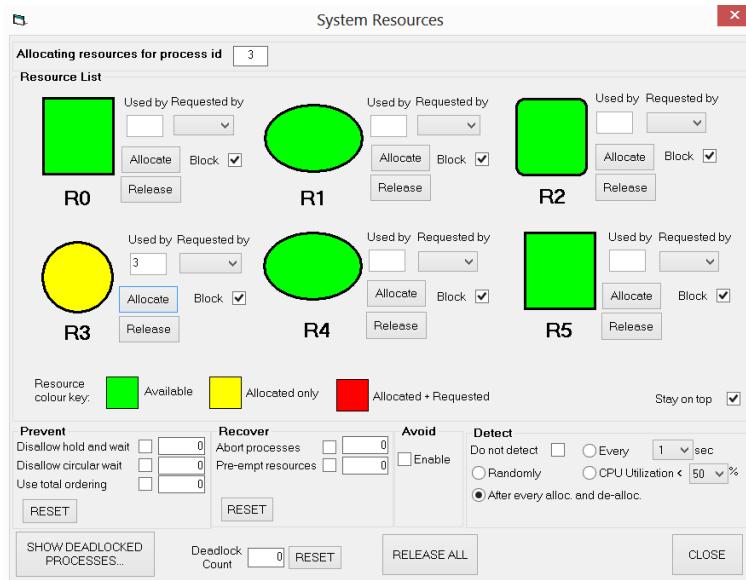
- Menambahkan program lain pada antrian



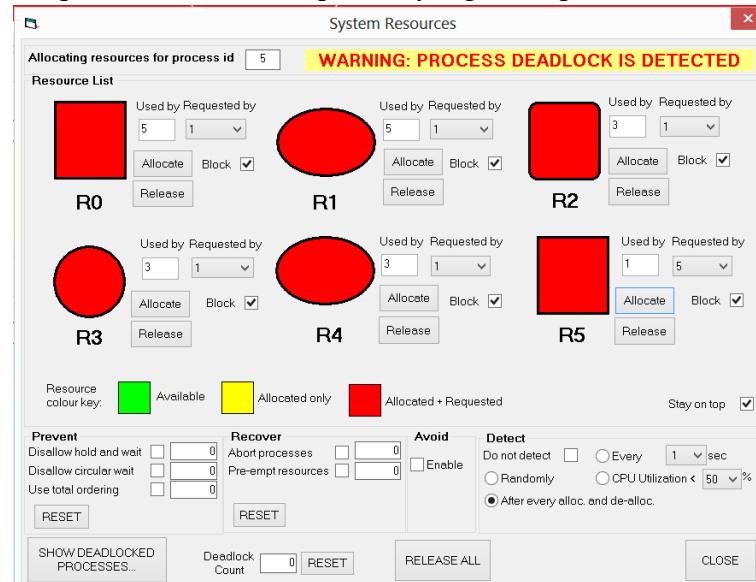
- Tampilan awal pada jendela VIEW RESOURCES



- Tampilan pada jendela VIEW RESOURCES setelah mengalokasikan proses id 3



- Tampilan pada jendela VIEW RESOURCES setelah mengalokasikan proses id 3 dan id 1 pada R yang sama pada R0, R1, R2, R3, R4, R5.



PERTEMUAN 13

1. First Fit

Input :

The screenshot shows a C++ IDE interface with three tabs: C:\12(1).cpp, C:\12(2).cpp, and C:\12(3).cpp. The C:\12(1).cpp tab is active. The code implements the First Fit memory management scheme. It starts by defining constants for block size (max) and file count (n). It then prompts the user for the number of blocks (m) and files (n). For each file, it asks for its size and then iterates through the blocks to find the first suitable fit. If a fit is found, it is marked as used (bf[i]=1) and the fragment size is calculated. If no fit is found, it is marked as unused (bf[i]=0) and the fragment size is set to 0. Finally, it prints the allocation details for each file.

```
#include<stdio.h>
#include<conio.h>
#define max 25
int main()
{
    int maxl,b[10],f[max],l[10],n,m,temp;
    #include<math.h>
    printf("\nMemory Management Scheme - First Fit");
    printf("\nEnter the number of blocks:-");
    scanf("%d",&m);
    printf("\nEnter the number of files:-");
    scanf("%d",&n);
    printf("\nEnter the size of the blocks:-\n");
    for(i=1;i<=m;i++)
    {
        printf("Block %d:",i);
        scanf("%d",&l[i]);
    }
    printf("\nEnter the size of the files :-\n");
    for(i=1;i<=n;i++)
    {
        printf("File %d:",i);
        scanf("%d",&f[i]);
    }
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=m;j++)
        {
            if(bf[j]==0)
            {
                temp=l[i]-f[i];
                if(temp==0)
                {
                    f[i]=j;
                    bf[j]=1;
                    break;
                }
            }
        }
        if(f[i]==0)
        {
            frag[i]=temp;
            bf[i]=0;
        }
    }
    printf("\nAllocation Details :-\n");
    for(i=1;i<=n;i++)
    {
        printf("File No.%d File Size %d Block No.%d Block Size %d Fragment %d\n",i,f[i],bf[i],l[bf[i]],frag[i]);
    }
    getch();
}
```

Output :

The terminal window displays the execution of the program C:\VINA\KULIAH\SEM 3\sistem operasi\12(1).exe. It starts with the title "Memory Management Scheme - First Fit". It prompts for the number of blocks (5) and files (2). It then lists the sizes of the blocks: 1:19, 2:68, 3:20, 4:40, 5:29. It lists the sizes of the files: 1:10, 2:11. Finally, it prints the allocation details:

File_no:	File_size :	Block_no:	Block_size:	Fragement
1	10	1	19	9
2	11	2	68	57

2. Best Fit

Input :

The screenshot shows a C++ IDE interface with three tabs: C:\12(1).cpp, C:\12(2).cpp, and C:\12(3).cpp. The C:\12(1).cpp tab is active. The code implements the Best Fit memory management scheme. It starts by defining constants for block size (max) and file count (n). It then prompts the user for the number of blocks (m) and files (n). For each file, it asks for its size and then iterates through the blocks to find the best fit. If a fit is found, it is marked as used (bf[i]=1) and the fragment size is calculated. If no fit is found, it is marked as unused (bf[i]=0) and the fragment size is set to 0. Finally, it prints the allocation details for each file.

```
#include<stdio.h>
#include<conio.h>
#define max 25
int main()
{
    int maxl,b[10],f[max],l[10],n,m,temp,lowest=10000;
    static int bf[10];
    printf("\nEnter the number of blocks:-");
    scanf("%d",&m);
    printf("\nEnter the number of files:-");
    scanf("%d",&n);
    printf("\nEnter the size of the blocks:-\n");
    for(i=1;i<=m;i++)
    {
        printf("Block %d:",i);
        scanf("%d",&l[i]);
    }
    printf("\nEnter the size of the files :-\n");
    for(i=1;i<=n;i++)
    {
        printf("File %d:",i);
        scanf("%d",&f[i]);
    }
    for(i=1;i<=n;i++)
    {
        for(j=1;j<=m;j++)
        {
            if(bf[j]==0)
            {
                temp=l[i]-f[i];
                if(temp==0)
                {
                    f[i]=j;
                    bf[j]=1;
                    lowest=j;
                    break;
                }
            }
        }
        if(f[i]==0)
        {
            frag[i]=temp;
            bf[i]=0;
        }
    }
    printf("\nAllocation Details :-\n");
    for(i=1;i<=n;i++)
    {
        printf("File No.%d File Size %d Block No.%d Block Size %d Fragment %d\n",i,f[i],bf[i],l[bf[i]],frag[i]);
    }
    getch();
}
```

Output :

```
C:\VINA\KULIAH\SEM 3\sistem operasi\12(2).exe
Enter the number of blocks:5
Enter the number of files:2

Enter the size of the blocks:- 
Block 1:19
Block 2:68
Block 3:20
Block 4:40
Block 5:29
Enter the size of the files :- 
File 1:10
File 2:11

File No File Size      Block No      Block Size      Fragment
1           10            1             19              9
2           11            3             20              9
```

3. Worst Fit

Input :

```
C:\12(0).cpp 2  C:\12(2).cpp 2  C:\12(3).cpp X
Volumes : V:\ C:\ C:\Windows
1  #include<iostream.h>
2  #include<conio.h>
3  #define max 25
4
5  int frag(max),bl(max),f(max),i,j,nb,nf,temp,highest;
6
7  printf("\nMemory Management Scheme - Worst Fit");
8  printf("\nEnter the number of blocks:"); scanf("%d",&nb);
9  printf("\nEnter the number of files:"); scanf("%d",&nf);
10
11  printf("\nEnter the size of the blocks:\n");
12  for(i=1;i<=nb;i++)
13  {
14      printf(" %d",i);
15  }
16  printf("\nEnter the size of the files :\n");
17  for(j=1;j<=nf;j++)
18  {
19      printf(" %d",j);
20  }
21  for(i=1;i<=nb;i++)
22  {
23      for(j=1;j<=nf;j++)
24      {
25          if(br[j]==1) //if br[j] is not allocated
26              if(temp==i)
27                  if(highest<temp)
28                      highest=temp;
29
30      }
31      if(r[i]>highest)
32          highest=r[i];
33  }
34  printf("\nfile_no\tfile_size\tblock_no\tblock_size\tfragment");
35  for(i=1;i<=nb;i++)
36  {
37      printf("\n%d\t%d\t%d\t%d\t%d",i,f[i],r[i],br[i],frag[i]);
38  }
39
40  getch();
41 }
```

Output :

```
C:\VINA\KULIAH\SEM 3\sistem operasi\12(3).exe
Memory Management Scheme - Worst Fit
Enter the number of blocks:5
Enter the number of files:2

Enter the size of the blocks:- 
Block 1:19
Block 2:68
Block 3:20
Block 4:40
Block 5:29
Enter the size of the files :- 
File 1:10
File 2:11

File_no:      File_size :      Block_no:      Block_size:      Fragement
1           10            2             68              58
2           11            4             40              29
```

Kesimpulan :

- First Fit memulai pencarian dari awal daftar memori dan menggunakan blok pertama yang terdaftar.

- Best Fit mencari blok yang pas (kecil) untuk digunakan.
- Worst Fit mencari blok paling besar dan menggunakannya.