

PP/018/V/R2



**LABORATORIUM
S1 INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN**

PETUNJUK PRAKTIKUM EDISI KURIKULUM OBE

SISTEM OPERASI



Penyusun

Arfiani Nur Khusna, S.T., M.Kom

Faisal Fajri Rahani, S.Si., M.Cs.

Ir. Nuril Anwar, S.T., M.Kom.

2023

HAK CIPTA

PETUNJUK PRAKTIKUM SISTEM OPERASI

Copyright© 2023,

Hak Cipta dilindungi Undang-Undang

Dilarang mengutip, memperbanyak atau mengedarkan isi buku ini, baik sebagian maupun seluruhnya, dalam bentuk apapun, tanpa izin tertulis dari pemilik hak cipta dan penerbit.

Diterbitkan oleh:

Program Studi Informatika

Fakultas Teknologi Industri

Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

Penulis : Arfiani Nur Khusna, S.T., M.Kom
Faisal Fajri Rahani, S.Si., M.Cs.
Ir. Nuril Anwar, S.T., M.Kom.

Editor : Laboratorium S1 Informatika, Universitas Ahmad Dahlan

Desain sampul : Laboratorium S1 Informatika, Universitas Ahmad Dahlan

Tata letak : Laboratorium S1 Informatika, Universitas Ahmad Dahlan

Ukuran/Halaman : 21 x 29,7 cm / 104 halaman

Didistribusikan oleh:



Laboratorium S1 Informatika

Universitas Ahmad Dahlan

Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166
Indonesia

KATA PENGANTAR

Sistem operasi merupakan bagian penting dari setiap sistem komputer. Demikian pula, kuliah tentang sistem operasi merupakan bagian penting dari setiap pendidikan sains komputer. Bidang ini mengalami perubahan yang cepat, karena komputer sekarang lazim di hampir setiap lingkungan kehidupan sehari-hari dari perangkat embedded di mobile melalui alat perencanaan yang paling canggih untuk pemerintah dan perusahaan multinasional. Namun, konsep dasarnya tetap cukup jelas, dan berdasarkan inilah dibuatlah modul praktikum ini.

Modul praktikum ditulis sebagai panduan praktikum untuk kuliah sistem operasi pada program studi Teknik Informatika UAD. Ini memberikan gambaran yang jelas tentang konsep yang mendasari sistem operasi. Sebagai prasyarat, kita berasumsi bahwa pembaca terbiasa dengan struktur data dasar, organisasi komputer, dan bahasa tingkat tinggi. Kami berharap para mahasiswa akan mendapatkan manfaat dari modul praktikum ini sehingga memudahkan pelaksanaan praktikum sistem operasi dan memahami konsep sistem operasi.

Selamat belajar sistem operasi dan mendapatkan manfaat dari implementasi sistem operasi modern.

Yogyakarta, 10 Juli 2023

Penyusun

DAFTAR PENYUSUN

Arfiani Nur Khusna, S.T., M.Kom.



S1 : Teknik Informatika UAD – Indonesia
 S2 : Teknik Informatika STMIK AMIKOM Yogyakarta – Indonesia
 Bidang Keahlian: Sistem Informasi
 Email : arfiani.khusna[at]tif.uad.ac.id

Menyusun Bab 1, 2, 5, 6, 7

Faisal Fajri Rahani S.Si., M.Cs.



S1 : Elektronika Instrumentasi UGM – Indonesia
 S2 : Ilmu Komputer UGM – Indonesia
 Bidang Keahlian : Sistem cerdas, Sistem Kendali, Pengolahan Citra
 Email : faisal.fajri[at]tif.uad.ac.id.

Menyusun Bab 3,4

Ir. Nuril Anwar, S.T., M.Kom.



S1 : Teknik Informatika UAD – Indonesia
 S2 : Teknik Informatika UII – Indonesia
 Bidang Minat : Computer Network & Security, Digital Forensics.
 Email : nuril.anwar[at]tif.uad.ac.id

Menyusun Bab 8,9,10

KONTRIBUSI PENULIS

Nomor Bab	Daftar Penulis
Bab I	Arfiani Nur Khusna, S.T., M.Kom
Bab II	Arfiani Nur Khusna, S.T., M.Kom
Bab III	Faisal Fajri Rahani, S.Si., M.Cs.
Bab IV	Faisal Fajri Rahani, S.Si., M.Cs.
Bab V	Arfiani Nur Khusna, S.T., M.Kom
Bab VI	Arfiani Nur Khusna, S.T., M.Kom
Bab VII	Arfiani Nur Khusna, S.T., M.Kom
Bab VIII	Ir. Nuril Anwar, S.T., M.Kom
Bab IX	Ir. Nuril Anwar, S.T., M.Kom
Bab X	Ir. Nuril Anwar, S.T., M.Kom

HALAMAN REVISI

Yang bertanda tangan di bawah ini:

Nama : Arfiani Nur Khusna, S.T., M.Kom

NIPM : 19850126 200909 011 0925468

Jabatan : Dosen Pengampu Mata Kuliah Sistem Operasi

Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum Sistem Operasi untuk Program Studi Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

No	Keterangan Revisi	Tanggal Revisi	Nomor Modul
1	Perubahan template baru	10 Juli 2023	PP/018/V/R1
2			PP/018/V/R2

Yogyakarta, 10 Juli 2023

Penyusun



Arfiani Nur Khusna, S.T., M.Kom

NIPM. 19850126 200909 011 0925468

HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : Murein Miksa Mardhia S.T., M.T.

NIPM : 19891019 201606 011 1236278

Jabatan : Kepala Laboratorium S1 Informatika

Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Gasal Tahun Akademik 2023/2024 di Laboratorium Praktikum S1 Informatika, Program Studi Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 10 Juli 2023

Mengetahui,
Ketua Kelompok Keilmuan

Dr. Ardiansyah, S.T., M.Cs.
NIPM. 19790723 200309 111 0932301

Kepala Laboratorium Praktikum
S1 Informatika



Murein Miksa Mardhia S.T., M.T.
NIPM. 19891019 201606 011 1236278

VISI DAN MISI PRODI INFORMATIKA

VISI

Menjadi program studi yang unggul dan inovatif dalam bidang rekayasa perangkat lunak dan sistem cerdas dengan dijiwai nilai-nilai Islam

MISI

1. Mengimplementasikan nilai-nilai AIK pada semua aspek kegiatan.
2. Memajukan ilmu pengetahuan dan teknologi Rekayasa Perangkat Lunak dan Sistem cerdas melalui pendidikan, penelitian, dan pengabdian kepada masyarakat.
3. Mengembangkan kerjasama dalam pendidikan, penelitian, dan pengabdian kepada masyarakat di tingkat lokal, nasional, maupun internasional.
4. Menyelenggarakan tata kelola program studi yang unggul dan inovatif.
5. Berperan aktif dalam kegiatan yang menunjang profesi dosen.

TATA TERTIB LABORATORIUM S1 INFORMATIKA

DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum untuk mengisi materi dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap INHAL).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan yang akan mengikuti inhal diwajibkan mendaftarkan diri dan membayar administrasi inhal kepada laboran inhal paling lambat H-1 jadwal inhal.
4. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
 - a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
 - b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
 - c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
 - d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
5. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
6. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
7. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
8. Selama praktikum dilarang *NGENET/NGE-GAME*, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
9. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
10. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.

11. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.
12. Praktikan dilarang mengubah *setting software/hardware* komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.
13. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.
14. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

ASISTEN PRAKTIKUM

1. Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
2. Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
3. Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
 - a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
 - b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
 - c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
 - d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
5. Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
6. Asisten mencatat dan merekap praktikan dengan status INHAL setiap minggu serta wajib mengumumkan mekanisme INHAL di awal pertemuan praktikum.
7. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
8. Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasistensi sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
9. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
10. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
11. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
12. Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Pelaksanaan (waktu dan metode) INHAL sama seperti praktikum mingguan/reguler.
4. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
5. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
6. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat izin dari Kepala Laboratorium Prodi Informatika.

Yogyakarta, 10 Juli 2023

Kepala Laboratorium Praktikum
S1 Informatika



Murein Miksa Mardhia S.T., M.T.
NIPM. 19891019 201606 011 1236278

DAFTAR ISI

Contents

HAK CIPTA	1
KATA PENGANTAR.....	2
DAFTAR PENYUSUN.....	3
KONTRIBUSI PENULIS	4
HALAMAN REVISI.....	5
HALAMAN PERNYATAAN.....	6
VISI DAN MISI PRODI INFORMATIKA.....	7
TATA TERTIB LABORATORIUM S1 INFORMATIKA	8
DOSEN/KOORDINATOR PRAKTIKUM.....	8
PRAKTIKAN	8
ASISTEN PRAKTIKUM.....	9
LAIN-LAIN	10
DAFTAR ISI.....	11
DAFTAR GAMBAR	15
DAFTAR TABEL.....	16
SKENARIO PRAKTIKUM SECARA DARING	17
PRAKTIKUM 1: PENGENALAN CPU SIMULATOR	19
1.1. DESKRIPSI CAPAIAN PEMBELAJARAN.....	19
1.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN	19
1.3. TEORI PENDUKUNG.....	19
1.4. HARDWARE DAN SOFTWARE	30
1.5. PRE-TEST.....	30
1.6. LANGKAH PRAKTIKUM	30
1.7. POST TEST.....	32
1.8. HASIL CAPAIAN PRAKTIKUM	32
PRAKTIKUM 2: THREADS.....	34
2.1. DESKRIPSI CAPAIAN PEMBELAJARAN.....	34
2.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN	34
2.3. TEORI PENDUKUNG.....	34
2.4. HARDWARE DAN SOFTWARE	36
2.5. PRE-TEST.....	36

2.6.	LANGKAH PRAKTIKUM	36
2.7.	POST TEST.....	38
2.8.	HASIL CAPAIAN PRAKTIKUM	38
PRAKTIKUM 3: SCHEDULING ALGORITHM.....		40
3.1.	DESKRIPSI CAPAIAN PEMBELAJARAN.....	40
3.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	40
3.3.	TEORI PENDUKUNG.....	40
3.4.	HARDWARE DAN SOFTWARE	41
3.5.	PRE-TEST.....	42
3.6.	LANGKAH PRAKTIKUM	42
3.7.	POST TEST.....	47
3.8.	HASIL CAPAIAN PRAKTIKUM	47
PRAKTIKUM 4: SINKRONISASI PROSES.....		49
4.1.	DESKRIPSI CAPAIAN PEMBELAJARAN.....	49
4.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	49
4.3.	TEORI PENDUKUNG.....	50
4.4.	HARDWARE DAN SOFTWARE	50
4.5.	PRE-TEST.....	51
4.6.	LANGKAH PRAKTIKUM	51
4.7.	POST TEST.....	55
4.8.	HASIL CAPAIAN PRAKTIKUM	55
PRAKTIKUM 5: BANKIR ALGORITHM FOR DEADLOCK		57
5.1.	DESKRIPSI CAPAIAN PEMBELAJARAN.....	57
5.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	57
5.3.	TEORI PENDUKUNG.....	57
5.4.	HARDWARE DAN SOFTWARE	58
5.5.	PRE-TEST.....	58
5.6.	LANGKAH PRAKTIKUM	58
5.7.	POST TEST.....	61
5.8.	HASIL CAPAIAN PRAKTIKUM	62
PRAKTIKUM 6: PENJADWALAN PROSES DAN ALOKASI MEMORI.....		64
6.1.	DESKRIPSI CAPAIAN PEMBELAJARAN	64
6.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	64
6.3.	TEORI PENDUKUNG	65
6.4.	HARDWARE DAN SOFTWARE	65

6.5.	PRE-TEST.....	65
6.6.	LANGKAH PRAKTIKUM	65
6.7.	POST TEST.....	68
6.8.	HASIL CAPAIAN PRAKTIKUM	68
PRAKTIKUM 7: SIMULASI PAGING		70
7.1.	DESKRIPSI CAPAIAN PEMBELAJARAN.....	70
7.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	70
7.3.	TEORI PENDUKUNG.....	70
7.4.	HARDWARE DAN SOFTWARE	71
7.5.	PRE-TEST.....	71
7.6.	LANGKAH PRAKTIKUM	71
7.7.	POST TEST.....	73
7.8.	HASIL CAPAIAN PRAKTIKUM	73
PRAKTIKUM 8: I/O MANAGEMENT (SPOOLING).....		75
8.1.	DESKRIPSI CAPAIAN PEMBELAJARAN.....	75
8.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	75
8.3.	TEORI PENDUKUNG.....	75
8.4.	HARDWARE DAN SOFTWARE	76
8.5.	PRE-TEST.....	76
8.6.	LANGKAH PRAKTIKUM	76
8.7.	POST TEST.....	77
8.8.	HASIL CAPAIAN PRAKTIKUM	77
PRAKTIKUM 9: MANAGEMENT FILE.....		79
9.1.	DESKRIPSI CAPAIAN PEMBELAJARAN	79
9.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	79
9.3.	TEORI PENDUKUNG.....	79
9.4.	HARDWARE DAN SOFTWARE	80
9.5.	PRE-TEST.....	80
9.6.	LANGKAH PRAKTIKUM	80
9.7.	POST TEST.....	87
9.8.	HASIL CAPAIAN PRAKTIKUM	87
PRAKTIKUM 10: SISTEM KEAMANAN DAN PROTEKSI.....		89
10.1.	DESKRIPSI CAPAIAN PEMBELAJARAN	89
10.2.	INDIKATOR KETERCAPAIAN PEMBELAJARAN	89
10.3.	TEORI PENDUKUNG.....	89

10.4.	HARDWARE DAN SOFTWARE	91
10.5.	PRE-TEST	91
10.6.	LANGKAH PRAKTIKUM	91
10.7.	POST TEST	99
10.8.	HASIL CAPAIAN PRAKTIKUM	100
DAFTAR PUSTAKA.....		102

DAFTAR GAMBAR

Gambar 1. 1 New Program Frame.....	20
Gambar 1. 2 Program list frame	20
Gambar 1. 3 Program instructions frame	20
Gambar 1. 4 Selecting CPU instructions.....	20
Gambar 1. 5 CPU Program Memory.....	21
Gambar 1. 6 Program Edit Functions	21
Gambar 1. 7 Program Removal	21
Gambar 1. 8 Program Control	22
Gambar 1. 9 Register Set.....	22
Gambar 1. 10 Data Memory Page.....	23
Gambar 1. 11 Program Stack Frame	23
Gambar 1. 12 Saving and Loading Programs.....	23
Gambar 1. 13 Console Button	24
Gambar 1. 14 Console Button	24
Gambar 1. 15 Virtual Keyboard.....	24
Gambar 1. 16 Jendela Utama Simulator	27
Gambar 1. 17 Tampilan Instruction View.....	28
Gambar 1. 18 Tampilan Special Register.....	28
Gambar 1. 19 Tampilan Register Set.....	29
Gambar 1. 20 Tampilan Program Stack.....	29
Gambar 1. 21 Tampilan Program Instruction.....	30
Gambar 1. 22 Tampilan Program List.....	30
Gambar 2. 1 Thread dan Proses	35
Gambar 2. 2 Thread dan Sumber Daya yang Dimiliki.....	35
Gambar 10. 1 Diagram Penanganan Malware Berdasarkan Waktu	90
Gambar 10. 2 Tampilan Awal Windows Security	92
Gambar 10. 3 Contoh Ketika Threat (ancaman) Terdeteksi.....	92
Gambar 10. 4 Opsi Penanganan File Threat.....	93
Gambar 10. 5 Fitur Firewall.....	93
Gambar 10. 6 Tampilan Awal Website.....	94
Gambar 10. 7 Hasil Pindai File.....	95
Gambar 10. 8 Contoh Link Website Yang Terdeteksi Berbahaya.....	95
Gambar 10. 9 Contoh Link Website Yang Tidak Terdeteksi Berbahaya	96
Gambar 10. 10 Tampilan Awal Windows Defender Firewall	97
Gambar 10. 11 Custom Pengaturan Firewall Berdasarkan Tipe Network	97
Gambar 10. 12 12 Windows Defender Firewall with Advanced Security	98
Gambar 10. 13 Aturan Outbound Mengontrol Lalu Lintas Keluar	99

DAFTAR TABEL

Table 1.1 CPU Simulator Instruction	25
---	----

SKENARIO PRAKTIKUM SECARA DARING

Nama Mata Praktikum : Sistem Operasi

Jumlah Pertemuan : 11

TABEL SKENARIO PRAKTIKUM DARING

Pertemuan ke	Judul Materi	Waktu (Lama praktikum sampai pengumpulan posttest)	Skenario Praktikum (Dari pemberian pre-test, post-test dan pengumpulannya serta mencantumkan metode yang digunakan misal video, whatsapp group, Google meet atau lainnya)
1	Pengenalan CPU Simulator	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
2	THREADS	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
3	Algoritma Penjadwalan	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
4	Sinkronisasi Proses	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
5	Bankir Algorithm for Deadlock	96 jam 90 menit	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom

		96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	3. Untuk Materi berupa video disampaikan dalam youtube prodi
6	Penjadwalan Proses dan Manajemen Memori & Teknik Alokasi Memori	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
7	Simulasi Peging	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
8	I/O Management (Spooling)	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
9	File Management	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
10	Sistem Keamanan & Proteksi	96 jam 90 menit 96 jam (pengumpulan posttest) 90 menit (kegiatan praktikum)	1. Praktikum dilaksanakan secara asinkron 2. Media praktikum berupa pretest, posttest dan materi dapat di akses melalui Gclassroom 3. Untuk Materi berupa video disampaikan dalam youtube prodi
11	Responsi	90 menit	1. Responsi dilakukan secara online 2. Soal responsi tersedia 7 soal dengan kode a-g Pengumpulan responsi dilakukan di google clasroom

PRAKTIKUM 1: PENGENALAN CPU SIMULATOR

Pertemuan ke : 1

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu menjelaskan diagram blok komputer dan dasar-dasar sistem operasi meliputi sejarah perkembangan dan tujuannya, keterkaitan sumber daya dan macam-macam layanan dalam sistem operasi

1.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Menjelaskan sistem operasi komputer serta perkembangan dan fungsi-fungsinya
2. Menerapkan instruksi untuk memindahkan data ke register, membandingkan isi register, memasukkan data ke stack, mengambil data dari stack, melompat ke lokasi alamat, menambahkan nilai dalam register.
3. Menjelaskan fungsi-fungsi dari register khusus CPU antara lain register PC, SR, dan SP.

1.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menerapkan instruksi untuk memindahkan data ke register, membandingkan isi register, memasukkan data ke stack, mengambil data dari stack, melompat ke lokasi alamat, menambahkan nilai dalam register
--------	---------	---

1.3. TEORI PENDUKUNG

Di bawah ini adalah deskripsi dari 4 tahap utama:

1. **Membuat program CPU yang berisi instruksi-instruksi**

Gambar 1. 1 New Program Frame

Masukkan nama program pada **Program Name** text box, contoh MyProgram. Masukkan nomor pada **Base Address** text box (saran = gunakan 0 untuk kasus ini). Kemudian klik tombol **ADD**. Nama program akan muncul pada frame LIST PROGRAM (Gambar 1.2).

Name	Base	Start	Type
MyProgram	0000	0000	R

Gambar 1. 2 Program list frame

2. Adding CPU instructions in the program

Gambar 1. 3 Program instructions frame

Dalam frame **PROGRAM INSTRUCTION** tombol **ADD NEW** aktif. Klik tombol tersebut untuk melihat CPU instruction kamu dapat memilih untuk program yang dibuat di atas (Gambar 1.3).

Memilih CPU instructions untuk memasuki program secara manual:

Gambar 1. 4 Selecting CPU instructions

Pada window **Instruction** yang diperlihatkan di atas, kamu dapat memilih beberapa CPU instruction yang tersedia untuk programmu. Instructions yang terhubung dikategorikan dalam beberapa kelompok. Kamu dapat memilih sebuah kelompok dengan mengklik pada tab grup. Pilih instruction yang diinginkan dari list di bawah **OP CODE**. Jika instruction yang dipilih membutuhkan operand(s) maka tipe yang tersedia akan aktif di bawah **SOURCE OPERAND**

dan frame **DESTINATION OPERAND**. Selanjutnya, klik tombol **NEW** untuk menambah instruction. Kamu dapat menambah instruction ganda tanpa menutup window ini.

Instruction akan muncul pada area memori CPU program seperti yang diperlihatkan pada Gambar 5. Simulator CPU membutuhkan membutuh instruction terakhir pada program yaitu **HLT** instruction (instructio ini memberhentikan proses simulasi. Seperti yang terdapat pada Gambar 1.5 Program sekarang dapat dijalankan.

CPU instructions in program:

INSTRUCTION MEMORY (RAM)			
PAdd	LAdd	Instruction	B
<input type="checkbox"/> 0000	0000	MOV #10, R01	00
<input type="checkbox"/> 0006	0006	HLT	00

Gambar 1. 5 CPU Program Memory

Setiap entry pada view **INSTRUCTION MEMORY** terdapat informasi sebagai berikut:

Padd(Physicial Address), **Ladd**(Logical Adress) dan instruction. Informasi lainnya juga tersedia tapi tidak relevan pada tahap ini.

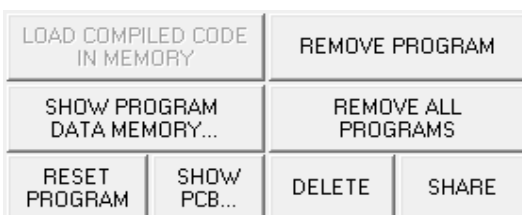
Editing the program instructions:



Gambar 1. 6 Program Edit Functions

Sekali instructions program dimasukkan. Mereka dapat diedit. Untuk melakukannya, pilih instruction yang diinginkan dan gunakan satu dari fungsi edit (**EDIT, DELETE, MOVE UP, MOVE DOWN**) diperlihatkan pada Gambar 1.6 untuk mengedit instruction yang dipilih. Gunakan tombol **INSERT ABOVE...** dan **INSERT BELOW...** untuk memasukkan instruction baru diatas(above) atau di bawah (below) instruction yang dipilih.

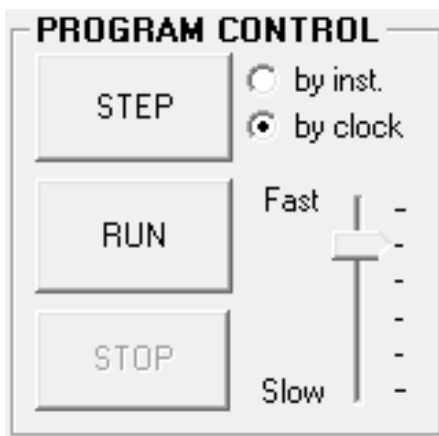
Menghapus program:



Gambar 1. 7 Program Removal

Program CPU dapat dihapus dengan mengklik tombol **REMOVE PROGRAM** diperlihatkan pada Gambar 1.7 Sekali program dihapus, maka program tersebut tidak akan ada lagi dan hilang. Bagaimanapun kamu dapat menyimpan program tersebut sehingga kamu dapat membuka kembali program tersebut nanti (Gambar 1.12)

3. Running the program



Gambar 1. 8 Program Control

Program CPU dapat dijalankan dalam 2 cara yang berbeda : 1) instruction dengan instruction, 2) otomatis dalam sekali jalan. Untuk menjalankan instruction yang dipilih dengan sendirinya, pertama pilih program seperti Gambar 1.5 dan double klik pada program tersebut. alternatifnya, kamu dapat mengklik tombol **STEP** yang diperlihatkan pada Gambar 1.8 (pastikan pilihan **by inst.** Dipilih). Gunakan tombol **STOP** untuk memberhentikan program yang berjalan. Gunakan **slider control** untuk mempercepat atau memperlambat program yang berjalan.

4. Observing and controlling the simulations

Mengamati/mengubah register CPU

Reg	Val (D)	C	Val (D)
<input checked="" type="checkbox"/> R01	10		
<input type="checkbox"/> R02	0		
<input type="checkbox"/> R03	0		
<input type="checkbox"/> R04	0		
<input type="checkbox"/> R05	0		

Gambar 1. 9 Register Set

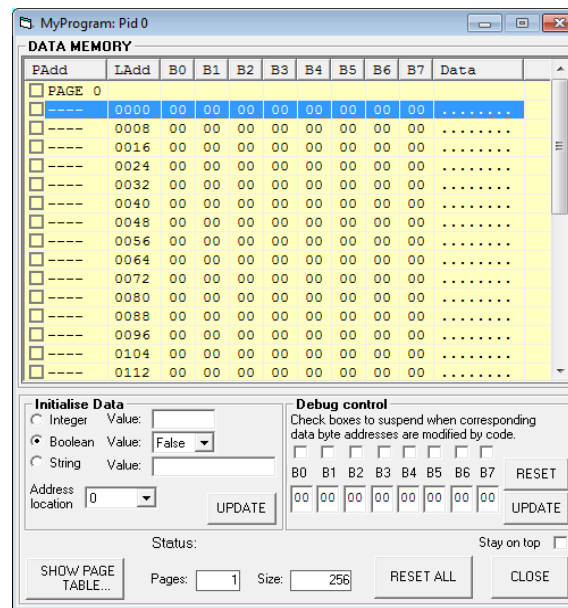
Sebuah instruction yang menulis atau membaca akses register pada frame **REGISTER SET** dan register yang diakses disorot. Frame ini menunjukkan register-register dan nilai mereka. Klik pada kolom **Val** untuk mengubah nilai dari desimal (**D**) ke format hex (**H**) dan sebaliknya.

Note: untuk mengubah nilai register secara manual, pertama pilih dan masukkan nilai pada field **Reg Value** dan klik pada tombol **CHANGE** (Gambar 1.9).

Observing/altering the program data:

Instruction CPU yang mengakses bagian dari memori yang mengandung data dapat menulis atau membaca data yang diakses. Informasi ini tersedia pada window halaman memori pada Gambar 1.10. Klik tombol **SHOW PROGRAM DATA MEMORY...** yang diperlihatkan pada Gambar 1.7 diatas. Pada window ini juga dapat mengedit isi dari data.

Kolom **Ladd** menunjukkan starting address pada setiap garis di display. Setiap garis dari display merepresentasikan 8byte dari informasi, jadi nomor **Ladd** terurut dari kecil hingga 8 dari setiap garis di bawah display. Kolom **B0** ke **b7** terdapat bytes 0 hingga 7. Kolom **Data** menunjukkan karakter yang dapat diperlihatkan sesuai dengan 8 bytes. Bytes tersebut berhubungan ke character yang tidak dapat diperlihatkan yang ditunjukkan sebagai dots. Data bytes hanya dapat diperlihatkan dalam format hex.

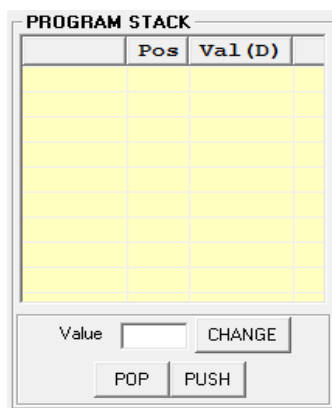


Gambar 1. 10 Data Memory Page

Untuk mengubah nilai dari bytes manapun pertama pilih garis yang mengandung bytes. Kemudian gunakan informasi pada frame **Initialize Data** untuk memodifikasi nilai dari bytes pada garis yang dipilih sebagai format **Integer**, **Boolean** atau **String**. Kamu harus mengklik tombol **UPDATE** untuk mengubah.

Cek check box **Stay on top** untuk memastikan window selalu berada di atas window lainnya ketika masih memperbolehkan mengakses windows di bawahnya.

(melihat/mengubah stack program):

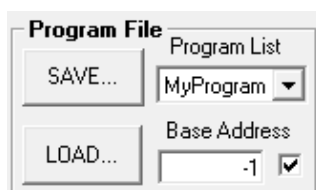


Gambar 1. 11 Program Stack Frame

Programs menjadikan **PROGRAM STACK** untuk menyimpan informasi penting sementara seperti **subroutine return address** dan **subroutine parameters** dan informasi relevan lainnya. Terdapat instruction yang dapat mendorong (**PSH**) data di atas stack dan dapat memunculkan data (**POP**) data dari atas stack ke register.

Kamu dapat mendorong dan memunculkan data secara manual dengan mengklik tombol **PUSH** dan **POP**. Kamu juga dapat memodifikasi entry stack dengan memilihnya, memasukkan nilai baru pada textbox nilai, dan mengklik **CHANGE**.

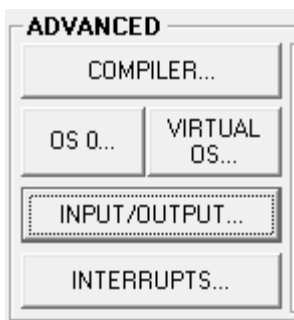
Saving and loading the CPU programs:



Gambar 1. 12 Saving and Loading Programs

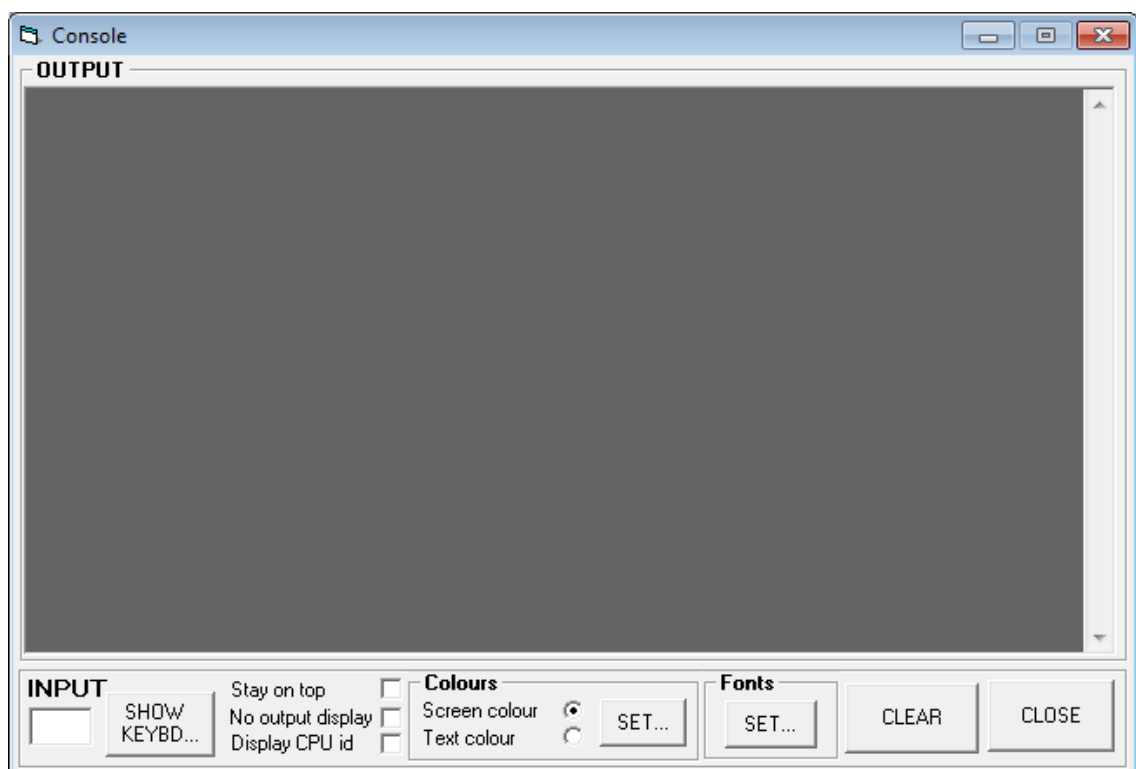
Untuk menyimpan program, pilih program dengan mencari program dari down list dan klik tombol **SAVE...** untuk memuat program yang tersimpan klik tombol **LOAD**

Observing the displayed information:



Gambar 1. 13 Console Button

Program dapat memperlihatkan informasi dan menerima data dari konsol yang disimulasi. Instruction **OUT** digunakan untuk memperlihatkan informasi dan instruction **IN** digunakan untuk menerima input dari konsol. Untuk menunjukkan konsol klik tombol **INPUT/OUTPUT...** ditunjukkan pada Gambar 1.13. Konsol window ditunjukkan pada Gambar 1.14.



Gambar 1. 14 Console Button

Cek check box **Stay on top** untuk memastikan window selalu berada di atas windows lainnya ketika masih memperbolehkan akses windows di bawahnya. Klik pada **SHOW KEYBD...** untuk menunjukkan keyboard kecil virtual untuk menginput data pada Gambar 1.15.



Gambar 1. 15 Virtual Keyboard

Cek check box **Lower Case** untuk menginput karakter kecil. Keyboard ini adalah bagian dari keyboard standard. Kamu juga dapat menginput dengan menulis di text box **INPUT** yang ditunjukkan pada Gambar 1.14

Table 1.1 CPU Simulator Instruction

Instruction	Description
Data transfer instructions	
MOV	Move data to register; move register to register e.g. MOV #2, R01 moves number 2 into register R01 MOV R01, R03 moves contents of register R01 into register R03
LDB	Load a byte from memory to register
LDW	Load a word (2 bytes) from memory to register
STB	Store a byte from register to memory
STW	Store a word (2 bytes) from register to memory
PSH	Push data to top of hardware stack (TOS); push register to TOS e.g. PSH #6 pushes number 6 on top of the stack PSH R03 pushes the contents of register R03 on top of the stack
POP	Pop data from top of hardware stack to register e.g. POP R05 pops contents of top of stack into register R05
Arithmetic instructions	
ADD	Add number to register; add register to register e.g. ADD #3, R02 adds number 3 to contents of register R02 and stores the result in register R02. ADD R00, R01 adds contents of register R00 to contents of register R01 and stores the result in register R01.
SUB	Subtract number from register; subtract register from register
MUL	Multiply number with register; multiply register with register
DIV	Divide number with register; divide register with register
Control transfer instructions	

JMP	Jump to instruction address unconditionally e.g. JMP 100 unconditionally jumps to address location 100
JLT	Jump to instruction address if less than (after last comparison)
JGT	Jump to instruction address if greater than (after last comparison)
JEQ	Jump to instruction address if equal (after last comparison) e.g. JEQ 200 jumps to address location 200 if the previous comparison instruction result indicates that the two numbers are equal.
JNE	Jump to instruction address if not equal (after last comparison)
CAL	Jump to subroutine address
RET	Return from subroutine
SWI	Software interrupt (used to request OS help)
HLT	Halt simulation
Comparison instruction	
CMP	Compare number with register; compare register with register e.g. CMP #5, R02 compare number 5 with the contents of register R02 CMP R01, R03 compare the contents of registers R01 and R03 Note: If $R01 = R03$ then the status flag Z will be set If $R03 > R01$ then none of the status flags will be set If $R01 > R03$ then the status flag N will be set
Input, output instructions	
IN	Get input data (if available) from an external IO device
OUT	Output data to an external IO device

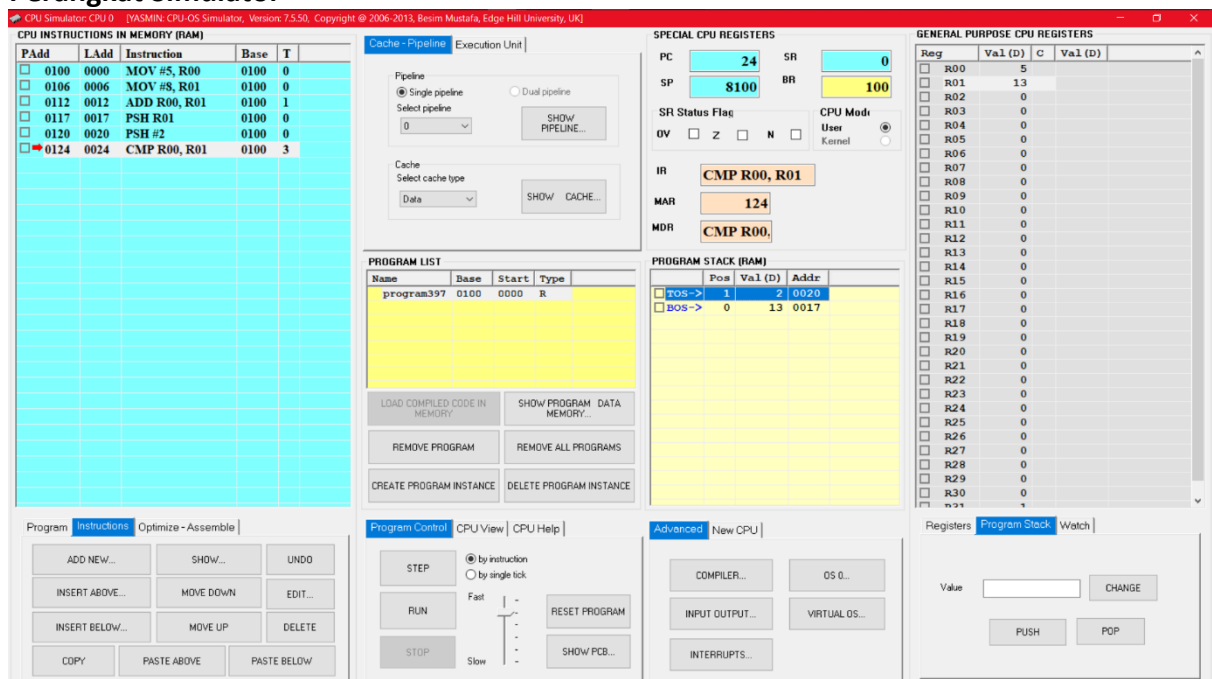
Model Pemrograman

Model pemrograman arsitektur komputer mendefinisikan komponen-komponen arsitektur pada tingkat bawah (low level architectural components) yang mencakup:

- Set instruksi prosesor
- Register-register
- Jenis-jenis pengalamatan instruksi dan data
- Interrupts* dan *exceptions*

Pada model pemrograman ini terdapat interaksi antar komponen diatas. Ini merupakan model pemrograman tingkat rendah (low-level) yang memungkinkan suatu komputasi terprogram.

Perangkat Simulator

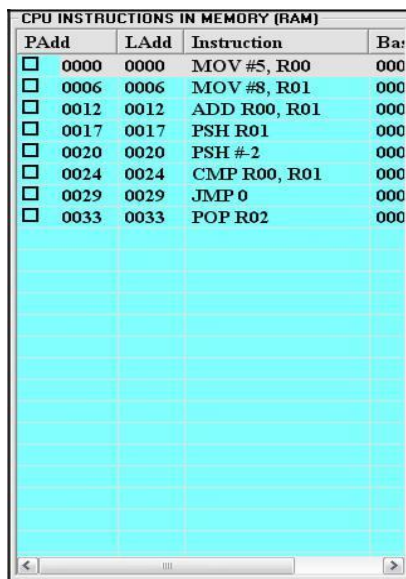


Gambar 1. 16 Jendela Utama Simulator

Jendela utama terdiri dari beberapa tampilan, yang merepresentasikan beberapa bagian fungsional prosesor yang disimulasikan, yaitu:

- Instruction memory (RAM)
- Special registers
- Register set
- Program stack

Tampilan *instruction memory*



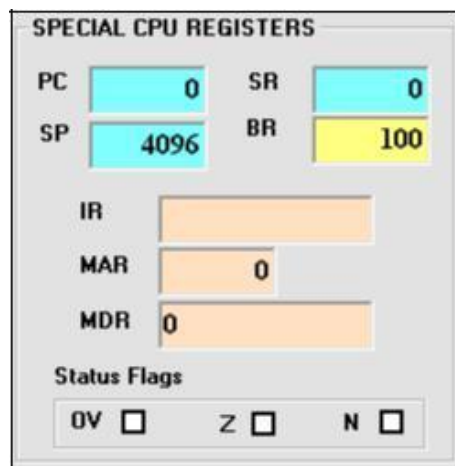
PAdd	LAdd	Instruction	Base
<input type="checkbox"/> 0000	0000	MOV #5, R00	000
<input type="checkbox"/> 0006	0006	MOV #8, R01	000
<input type="checkbox"/> 0012	0012	ADD R00, R01	000
<input type="checkbox"/> 0017	0017	PSH R01	000
<input type="checkbox"/> 0020	0020	PSH #2	000
<input type="checkbox"/> 0024	0024	CMP R00, R01	000
<input type="checkbox"/> 0029	0029	JMP 0	000
<input type="checkbox"/> 0033	0033	POP R02	000

Gambar 1. 17 Tampilan Instruction View

Tampilan berisi instruksi-instruksi dari program. Instruksi ditampilkan dalam urutan instruksi mnemonik low-level (format assembler), tidak dalam bentuk kode biner. Hal ini bertujuan untuk memperjelas dan membuat kode lebih mudah dibaca.

Tiap instruksi mempunyai dua alamat: alamat fisik (Padd) dan alamat logika (Ladd). Tampilan ini juga menyajikan alamat dasar (Base) terhadap tiap instruksi. Urutan instruksi pada program yang sama akan mempunyai alamat dasar yang sama.

Tampilan *Special Registers*



SPECIAL CPU REGISTERS	
PC	0
SP	4096
SR	0
BR	100
IR	
MAR	0
MDR	0
Status Flags	
OV	<input type="checkbox"/>
Z	<input type="checkbox"/>
N	<input type="checkbox"/>

Gambar 1. 18 Tampilan Special Register

Gambar 1.18 menyajikan register-register dengan fungsi-fungsi khusus yang telah ditentukan:

PC: Program Counter, berisi alamat instruksi berikutnya yang akan dieksekusi.

IR: Instruction Register, berisi instruksi yang sedang dieksekusi saat ini.

SR: Status Register, berisi informasi yang memberikan hasil dari instruksi yang dieksekusi terakhir.

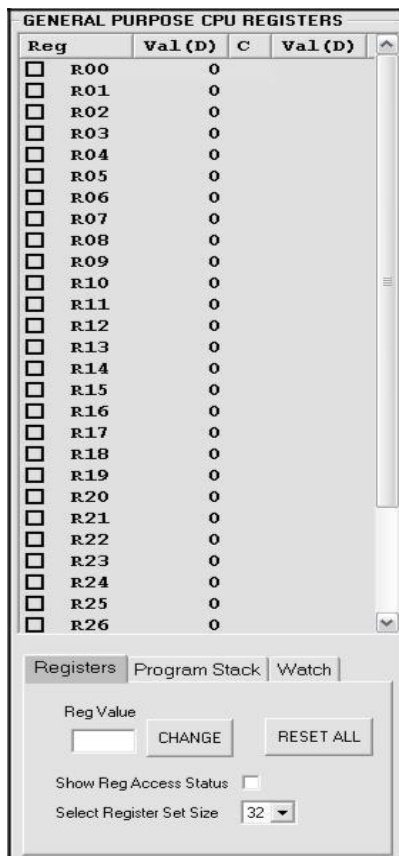
SP: Stack Pointer, register menunjuk ke nilai yang berada pada bagian atas stack.

BR: Base Register, berisi alamat dasar saat ini.

MAR: Memory Address Register, berisi alamat memori yang sedang diakses.

Status Bits: OV: Overflow; Z: Zero; N: Negative

Tampilan Register Set



Gambar 1. 19 Tampilan Register Set

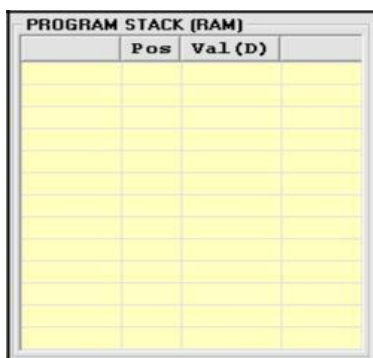
Tampilan register set menunjukkan isi dari semua register-register tujuan umum (general-purpose), yang digunakan untuk menampung nilai sementara ketika instruksi program dieksekusi.

Arsitektur ini mendukung 8 hingga 64 register. Register-register ini sering digunakan untuk menyimpan nilai variabel program pada bahasa tingkat tinggi.

Tidak semua arsitektur memiliki register sebanyak ini. Beberapa lebih banyak (misalnya 128 register) dan beberapa lainnya lebih sedikit (misalnya 8 register). Pada semua kasus, register-register ini bekerja untuk tujuan yang sama.

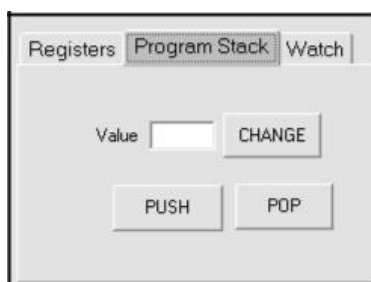
Bagian ini menampilkan nama tiap register (Reg), nilai (Val), dan beberapa informasi lainnya untuk keperluan debug program. Kita juga dapat melakukan reset (RESET) atau mengisi nilai register (CHANGE) secara manual.

Tampilan Program Stack



Gambar 1. 20 Tampilan Program Stack

Program Stack menampung nilai sementara ketika instruksi dieksekusi. Stack menggunakan struktur data LIFO (last-In-First-Out). Stack sering digunakan untuk efisiensi *interrupt handling* dan *sub-routine call*.



Instruksi PUSH dan POP digunakan untuk menyimpan dan mengambil nilai pada bagian teratas stack.

1.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. CPU Simulator

1.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan fungsi dari CPU Simulator!	20
2.	CPL-03	CPMK-01	Sebutkan dan jelaskan instruksi dari Special Register	40
3.	CPL-03	CPMK-01	Jelaskan cara menempatkan instruksi ke instruction memory!	40

1.6. LANGKAH PRAKTIKUM

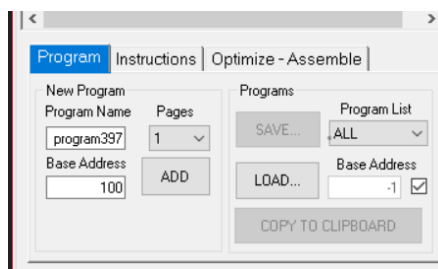
Aturan Penilaian (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum A-B	Hasil praktikum langkah A-B	100

Langkah-Langkah Praktikum:

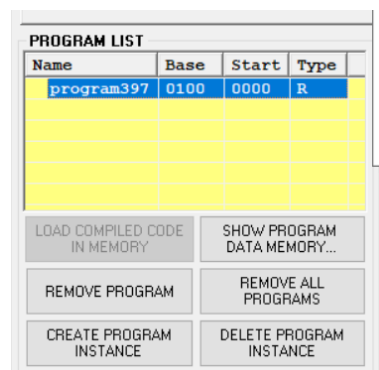
A. Persiapan Praktikum

Pertama-tama perlu menempatkan sejumlah instruksi pada tampilan instruction memory (menggambarkan RAM pada mesin yang nyata) sebelum melakukan eksekusi instruksi. Berikut adalah cara menempatkan instruksi ke instruction memory:



Gambar 1. 21 Tampilan Program Instruction

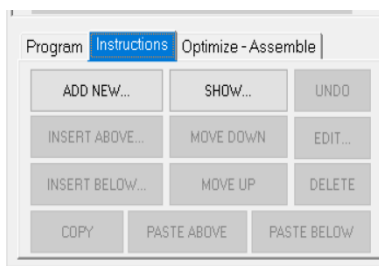
Pada tampilan program instruction, masukkan nama program (misal: program397), dan juga alamat dasar (untuk praktikum ini beri alamat dasar nilai 100). Klik tombol ADD. Maka program baru dengan namanya akan dimasukkan ke tampilan Program List seperti yang disajikan gambar 1.22. Gunakan tombol SAVE.../ LOAD... untuk menyimpan dan mengambil instruksi dari file.



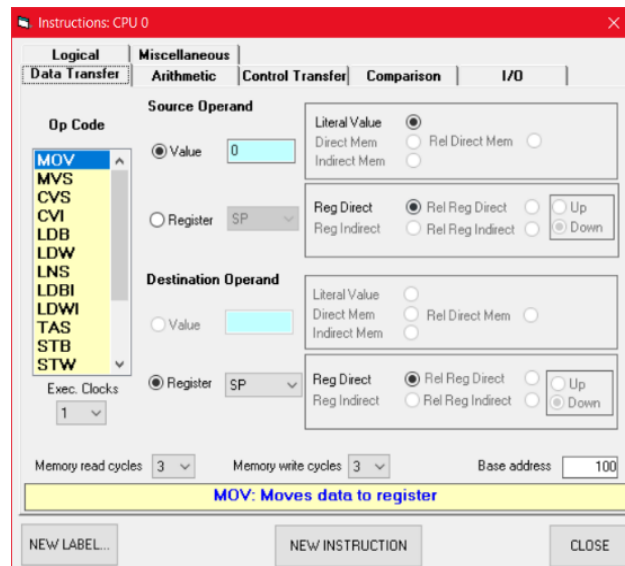
Gambar 1. 22 Tampilan Program List

Gunakan tombol REMOVE PROGRAM untuk menyingkirkan program yang disorot. Gunakan tombol REMOVE ALL PROGRAMS untuk menyingkirkan semua program yang ada di daftar. Sebagai catatan, ketika program disingkirkan maka instruksi-instruksi yang terkait dengannya juga akan dihapus dari tampilan instruction memory.

B. Langkah Praktikum



Untuk memasukkan instruksi ke CPU, klik tombol ADD NEW... yang akan menampilkan jendela Instructions: CPU0.



Gunakan jendela ini untuk memasukkan instruksi. Pada **Appendix** tersedia beberapa instruksi yang dikenali oleh simulator ini dan terdapat pula contoh penggunaannya.

a. Transfer data

1. Buatlah instruksi yang memindahkan (move) angka 5 ke register R00.
2. Eksekusi instruksi diatas (dengan klik dua kali pada tampilan instruction memory).
3. Buatlah instruksi yang memindahkan angka 8 ke register R01 dan eksekusilah.
4. Amati isi R00 dan R01 pada tampilan Register Set.

b. Aritmatika

6. Buatlah suatu instruksi yang menambahkan (add) isi R00 dan R01.
7. Eksekusilah.
8. Amati dimana hasil penjumlahan tersebut disimpan.

c. Stack Pointer (SP)

9. Buatlah instruksi yang menaruh hasil diatas pada program stack, kemudian eksekusilah.
10. Buatlah instruksi untuk menaruh (push) angka -2 pada stack teratas dan eksekusilah.
11. Amati nilai register SP.

d. Pemanding

12. Buatlah instruksi untuk membandingkan nilai register R00 dan R01.
13. Eksekusilah.
14. Amati nilai register SR.
15. Amati bit status OV/Z/N pada status register.
16. Analisa status register tersebut.

e. Stack Pointer (SP)

17. Buatlah instruksi untuk mengambil (pop) nilai teratas dari program stack ke register R02.
18. Eksekusilah.
19. Amati nilai pada register SP.
20. Buatlah suatu instruksi untuk mengambil nilai teratas dari program stack ke register R03.
21. Eksekusilah.
22. Amati nilai pada register SP.
23. Eksekusi lagi instruksi yang terakhir. Apa yang terjadi? Jelaskan.

1.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan langkah - langkah membuat program sederhana yaitu instruksi menambahkan isi R00 dan R01 dengan hasil akhir 10, wajib disertai screenshot.	25
2.	CPL-03	CPMK-01	Jelaskan langkah - langkah membuat instruksi membandingkan register R07 dan R04 (nilainya serah), wajib disertai screenshot.	
3	CPL-03	CPMK-01	Jelaskan setiap langkah praktikum sebelumnya menggunakan bahasa anda sendiri dan berikan kesimpulan!	25

1.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

PRAKTIKUM 2: THREADS

Pertemuan ke : 2

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-01	Mampu menjelaskan diagram blok komputer dan dasar-dasar sistem operasi meliputi sejarah perkembangan dan tujuannya, keterkaitan sumber daya dan macam-macam layanan dalam sistem operasi

2.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Menulis kode sumber untuk membuat thread
2. Menampilkan daftar proses/ thread dan pohon proses yang menggambarkan hubungan antar proses induk/anak.
3. Memodifikasi kode sumber untuk membuat versi tanpa thread dan membandingkannya dengan versi yang menggunakan thread
4. Memahami bagaimana thread-thread berbagi sumber daya dari proses induk

2.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menerapkan penulisan kode sumber thread, menampilkan daftar proses/ thread dan pohon proses yang menggambarkan hubungan antar proses induk, memodifikasi kode sumber untuk membuat versi tanpa thread dan membandingkannya dengan versi yang menggunakan, memahami bagaimana thread-thread berbagi sumber daya dari proses induk.
--------	---------	---

2.3. TEORI PENDUKUNG

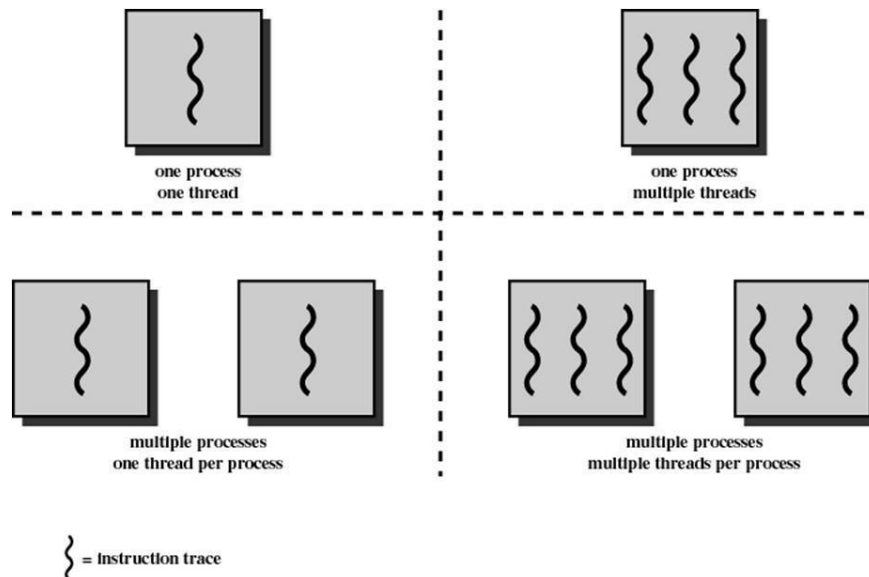
Konsep proses mengandung dua karakteristik:

1. Unit kepemilikan sumber daya

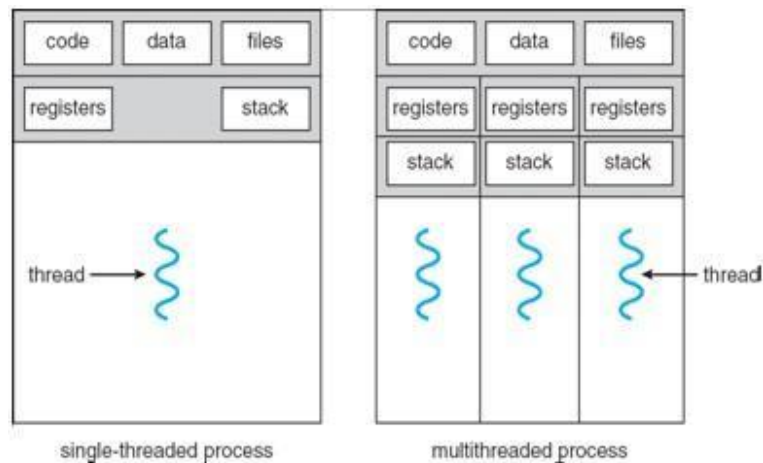
2. Unit pengiriman (dispatching)

Untuk membedakan kedua karakteristik itu, unit dispatching biasanya disebut sebagai thread, atau lightweight process, sedangkan unit kepemilikan sumber daya biasanya masih disebut sebagai proses atau task (Stallings, 2003).

Dengan begitu pada tiap thread terdapat ID thread, program counter, register, dan stack (terkait unit dispatching). Namun saling berbagi dengan thread yang lain (dalam satu proses) terhadap kepemilikan sumber daya yang sama.



Gambar 2. 1 Thread dan Proses



Gambar 2. 2 Thread dan Sumber Daya yang Dimiliki

2.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. CPU Simulator

2.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan apa yang di maksud dengan thread!	25
2.	CPL-03	CPMK-01	Jelaskan perbedaan antara proses dan thread!	25
3.	CPL-03	CPMK-01	Jelaskan keuntungan thread dalam sistem operasi!	25
4.	CPL-03	CPMK-01	Bagaimana hubungan antara multithread dengan multicore/multiprocessor?	25

2.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

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

1. Jalankan Simulator. Tuliskan kode berikut ini ke compiler.

```

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")

do
loop
end

```

2. Compile dan Run kode menggunakan CPU-OS Simulator
 - a. Compile dengan menekan tombol COMPILE pada compiler.
 - b. Melalui CPU Simulator, tampilkan jendela konsol dengan klik tombol INPUT/OUTPUT... aktifkan STAY ON TOP.

- c. Beralihlah ke jendela OS Simulator, buat satu proses dengan CREATE NEW PROCESS.
- d. Pastikan jenis penjadwalan adalah ROUND ROBIN, 10 ticks dengan kecepatan simulasi maksimum.
3. Tekan tombol START dan pada waktu yang bersamaan amati tampilan pada jendela console.
 - a. Berapakah jumlah proses yang telah dibuat?
 - b. Sebutkan mana yang disebut proses dan mana yang disebut thread!
4. Klik tombol VIEW PROCESS LIST. Kemudian klik PROCESS TREE. Identifikasi proses induk dan proses turunan!
5. Gunakan 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.
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?
5. Gunakan tombol KILL untuk menghentikan proses!

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

1. Buatlah program baru dengan memodifikasi source code ThreadTest1. Penambahan code ditandai dengan cetak tebal dan underline.

```

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

```

1. Compile program ThreadTest2. Klik tombol SYMBOL TABLE pada jendela compiler. Amati informasi tentang variabel s1 dan s2.
2. Melalui CPU Simulator, tampilkan jendela konsol dengan klik tombol INPUT/OUTPUT... aktifkan STAY ON TOP.
3. Beralihlah ke jendela OS Simulator, buat satu proses dengan CREATE NEW PROCESS.
4. Pastikan jenis penjadwalan adalah ROUND ROBIN, kecepatan simulasi maksimum.
5. Tekan tombol START dan pada waktu yang bersamaan amati tampilan pada jendela console. Gunakan tombol SUSPEND untuk menghentikan proses sementara.
6. Tekan tombol SHOW MEMORY..., amati alamat memory yang digunakan oleh variable s1 dan s2. Simpulkan apa yang terjadi.

2.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan serta simpulkan perbedaan antara code sebelum di modifikasi dan sesudah di modifikasi!	100

2.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

PRAKTIKUM 3: SCHEDULING ALGORITHM

Pertemuan ke : 3

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-04	Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya
CPMK-03	Mampu menerapkan teknik penjadwalan prosesor, metode evaluasi penjadwalan, pengalokasian data secara contiguous dan non contiguous

3.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

Mensimulasikan algoritma penjadwalan CPU non-preemptive untuk menemukan waktu penyelesaian dan waktu tunggu

3.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-04	CPMK-03	Kemampuan mahasiswa dalam mensimulasikan algoritma penjadwalan CPU non-preemptive untuk menemukan waktu penyelesaian dan waktu tunggu
--------	---------	---

3.3. TEORI PENDUKUNG

A. ALGORITMA PENJADWALAN FCFS CPU

Untuk algoritma penjadwalan FCFS, baca jumlah proses / pekerjaan dalam sistem, waktu burst CPU-nya. Penjadwalan dilakukan berdasarkan waktu kedatangan dari proses terlepas dari parameter lainnya. Setiap proses akan dieksekusi sesuai dengan waktu

kedatangannya. Hitung waktu tunggu dan waktu penyelesaian masing-masing proses sesuai.

B. ALGORITMA PENJADWALAN SJF CPU

Untuk algoritma penjadwalan SJF, baca jumlah proses / pekerjaan dalam sistem, waktu burst CPU-nya. Atur semua pekerjaan sesuai dengan waktu ledakan mereka. Mungkin ada dua pekerjaan dalam antrian dengan waktu eksekusi yang sama, dan kemudian pendekatan FCFS harus dilakukan. Setiap proses akan dieksekusi sesuai dengan lamanya waktu meledaknya. Kemudian hitung waktu tunggu dan waktu penyelesaian masing-masing proses sesuai.

C. ALGORITMA PENJADWALAN ROUND ROBIN CPU

Untuk algoritma penjadwalan round robin, baca jumlah proses / pekerjaan dalam sistem, waktu burst CPU-nya, dan ukuran slice waktu. Irisan waktu ditugaskan untuk setiap proses dalam porsi yang sama dan dalam urutan melingkar, menangani semua proses eksekusi. Ini memungkinkan setiap proses untuk mendapatkan kesempatan yang sama. Hitung waktu tunggu dan waktu penyelesaian masing-masing proses yang sesuai.

D. ALGORITMA PENJADWALAN CPU PRIORITY

Untuk algoritma penjadwalan prioritas, baca jumlah proses / pekerjaan dalam sistem, waktu burst CPU-nya, dan prioritas. Atur semua pekerjaan agar sesuai dengan prioritas mereka. Mungkin ada dua pekerjaan dalam antrian dengan prioritas yang sama, dan kemudian pendekatan FCFS harus dilakukan. Setiap proses akan dieksekusi sesuai dengan prioritasnya. Hitung waktu tunggu dan waktu penyelesaian masing-masing proses yang sesuai.

3.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. CPU Simulator

3.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-03	Jelaskan apa yang di maksud dengan algoritma penjadwalan?	30
2.	CPL-04	CPMK-03	Jelaskan apa yang dimaksud dari algoritma penjadwalan FCFS dan SJF CPU beserta cara kerjanya!	30
3.	CPL-04	CPMK-03	Jelaskan apa perbedaan dari algoritma FCFS dan SJF CPU dan diantara keduanya manakah yang paling optimal digunakan? Berikan alasannya!	40

3.6. LANGKAH PRAKTIKUM

Aturan Penilaian (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-04	CPMK-03	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

A. ALGORITMA PENJADWALAN FCFS CPU

```
#include<stdio.h>
#include<conio.h>
main() {    int bt[20], wt[20], tat[20], i, n;
float wtavg, tatavg;
clrscr();
printf("\nEnter the number of processes --  ");
scanf("%d", &n);
for(i=0;i<n;i++)
{
printf("\nEnter Burst Time for Process %d --  ", i);
scanf("%d", &bt[i]);
}
wt[0] = wtavg = 0;
tat[0] = tatavg = bt[0];
for(i=1;i<n;i++)
{
    wt[i] = wt[i-1] +bt[i-1];
    tat[i] = tat[i-1]+bt[i];
    wtavg = wtavg + wt[i];
    tatavg = tatavg + tat[i];
}
printf("\t PROCESS \tBURST TIME \t WAITING TIME\t
TURNAROUND TIME\n");
for(i=0;i<n;i++)
    printf("\n\t P%d \t\t %d \t\t %d \t\t %d", i, bt[i], wt[i],
tat[i]);
printf("\nAverage Waiting Time -- %f", wtavg/n);
printf("\nAverage Turnaround Time -- %f", tatavg/n);
getch();
}
```

INPUT

Masukkan jumlah proses -- 3

Masukkan Burst Time untuk Proses 0 -- 24

Masukkan Burst Time untuk Proses 1 -- 3

Masukkan Burst Time untuk Proses 2 -- 3

OUTPUT

PROCESS	BURST TIME	WAITING TIME	TURNAROUND TIME
P0	24	0	24
P1	3	24	27
P2	3	27	30

Rata-rata Waiting Time-- 17.000000

Rata-rata Turnaround Time -- 27.000000

B. ALGORITMA PENJADWALAN SJF CPU

```
#include<stdio.h>
#include<conio.h>
main()
{
    int p[20], bt[20], wt[20], tat[20], i, k, n, temp;
    float wtavg, tatavg;
    clrscr();
    printf("\nEnter the number of processes -- ");
    scanf("%d", &n);
    for(i=0;i<n;i++)
    {
        p[i]=i;
        printf("Enter Burst Time for Process %d -- ", i);
        scanf("%d", &bt[i]);
    }
    for(i=0;i<n;i++)
        for(k=i+1;k<n;k++)
            if(bt[i]>bt[k])
            {
                temp=bt[i];
                bt[i]=bt[k];
                bt[k]=temp;
                temp=p[i];
                p[i]=p[k];
                p[k]=temp;
            }
    wt[0] = wtavg = 0;
    tat[0] = tatavg = bt[0];
    for(i=1;i<n;i++)
    {
        wt[i] = wt[i-1] +bt[i-1];
        tat[i] = tat[i-1] +bt[i];
        wtavg = wtavg + wt[i];
    }
```

```

        tatavg = tatavg + tat[i];
    }
    printf("\n\t PROCESS \tBURST TIME \t WAITING TIME\t TURNAROUND
    TIME\n");
    for(i=0;i<n;i++)
        printf("\n\t P%d \t\t %d \t\t %d \t\t %d", p[i], bt[i],
        wt[i], tat[i]);
    printf("\nAverage Waiting Time -- %f", wtavg/n);
    printf("\nAverage Turnaround Time -- %f", tatavg/n);
    getch();
}

```

INPUT

Masukkan jumlah proses -- 4
 Masukkan Burst Time untuk Proses 0 -- 6
 Masukkan Burst Time untuk Proses 1 -- 8
 Masukkan Burst Time untuk Proses 2 -- 7
 Masukkan Burst Time untuk Proses 3 -- 3

OUTPUT

PROCESS	BURST TIME	WAITING TIME	TURNAROUND TIME
P3	3	0	3
P0	6	3	9
P2	7	9	16
P1	8	16	24

Rata-rata Waiting Time-- 7.000000

Rata-rata Turnaround Time -- 13.00000

C. ALGORITMA PENJADWALAN ROUND ROBIN CPU

```

#include<stdio.h>
main()
{
    int i,j,n,bu[10],wa[10],tat[10],t,ct[10],max;
    float awt=0,att=0,temp=0;
    clrscr();
    printf("Enter the no of processes -- ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("\nEnter Burst Time for process %d -- ", i+1);
        scanf("%d",&bu[i]);
    }
}

```

```

        ct[i]=bu[i];
    }
    printf("\nEnter the size of time slice -- ");
    scanf("%d",&t);
    max=bu[0];
    for(i=1;i<n;i++)
        if(max<bu[i])
            max=bu[i];
    for(j=0;j<(max/t)+1;j++)
        for(i=0;i<n;i++)
            if(bu[i]!=0)
                if(bu[i]<=t)
                {
                    tat[i]=temp+bu[i];
                    temp=temp+bu[i];
                    bu[i]=0;
                }
            else
            {
                bu[i]=bu[i]-t;
                temp=temp+t;
            }
        for(i=0;i<n;i++)
        {
            wa[i]=tat[i]-ct[i];
            att+=tat[i];
            awt+=wa[i];
        }
    printf("\nThe Average Turnaround time is -- %f",att/n);
    printf("\nThe Average Waiting time is -- %f ",awt/n);
    printf("\n\tPROCESS\t BURST TIME \t WAITING TIME\tTURNAROUND TIME\n");
    for(i=0;i<n;i++)
        printf("\t%d \t %d \t\t %d \t\t %d \n",i+1,ct[i],wa[i],tat[i]);
    getch();
}

```

INPUT

Masukkan jumlah proses -- 3
 Masukkan Burst Time untuk Proses 1 -- 24
 Masukkan Burst Time untuk Proses 2 -- 3
 Masukkan Burst Time untuk Proses 3 -- 3

Masukkan ukuran potongan waktu (time slice) -- 3

OUTPUT

Rata-rata Waiting Time-- 15.666667
 Rata-rata Turnaround Time -- 5.666667

PROCESS	BURST TIME	WAITING TIME	TURNAROUND TIME
1	24	6	30
2	3	4	7
3	3	7	10

D. ALGORITMA PENJADWALAN PRIORITY CPU

```
#include<stdio.h>
main()
{
    int p[20],bt[20],pri[20], wt[20],tat[20],i, k, n, temp;
    float wtavg, tatavg;
    clrscr();
    printf("Masukkan jumlah proses --- ");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        p[i] = i;
        printf("Masukkan Burst Time & Prioritas Proses %d ---",i);
        scanf("%d %d",&bt[i], &pri[i]);
    }
    for(i=0;i<n;i++)
        for(k=i+1;k<n;k++)
            if(pri[i] > pri[k])
            {
                temp=p[i];
                p[i]=p[k];
                p[k]=temp;

                temp=bt[i];
                bt[i]=bt[k];
                bt[k]=temp;

                temp=pri[i];
                pri[i]=pri[k];
                pri[k]=temp;
            }
    wtavg = wt[0] = 0;
    tatavg = tat[0] = bt[0];
    for(i=1;i<n;i++)
    {
        wt[i] = wt[i-1] + bt[i-1];
        tat[i] = tat[i-1] + bt[i];

        wtavg = wtavg + wt[i];
        tatavg = tatavg + tat[i];
    }
    printf("\nPROCESS\t\t\tPRIORITY\t\tBURST TIME\t\tWAITING TIME\t\tTURNAROUND TIME");
    for(i=0;i<n;i++)
        printf("\n%d \t\t\t %d \t\t\t %d \t\t\t %d \t\t\t %d",p[i],pri[i],bt[i],wt[i],tat[i]);

    printf("\nRata-rata Waiting Time is --- %f",wtavg/n);
    printf("\nRata-rata Turnaround Time is --- %f",tatavg/n);
    getch();
}
```

INPUT

Masukkan jumlah proses -- 5

Masukkan Burst Time untuk Proses 0 -- 10 3

Masukkan Burst Time untuk Proses 1 -- 1 1

Masukkan Burst Time untuk Proses 2 -- 2 4
 Masukkan Burst Time untuk Proses 3 -- 1 5
 Masukkan Burst Time untuk Proses 4 -- 5 2

OUTPUT

PROCESS	PRIORITY	BURST TIME	WAITING TIME	TURNAROUND TIME
1	1	1	0	1
4	2	5	1	6
0	3	10	6	16
2	4	2	16	18
3	5	1	18	19
Rata-rata Waiting Time--		8.200000		
Rata-rata Turnaround Time --		12.000000		

1. Tulis program C untuk mengimplementasikan algoritma penjadwalan round robin CPU untuk skenario yang diberikan berikut.
Semua proses dalam sistem dibagi menjadi dua kategori - proses sistem dan proses pengguna. Proses sistem harus diberi prioritas lebih tinggi daripada proses pengguna. Pertimbangkan ukuran kuantum waktu untuk proses sistem dan proses pengguna masing-masing 5 msec dan 2 msec.
2. Tulis program C untuk mensimulasikan algoritma penjadwalan SJF CPU pre-emptive!

3.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-02	Buatlah analisis perbedaan masing2 dari algoritma penjadwalan FCFS dan SJF CPU dengan mencantumkan screenshot hasil running program!	100

3.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-04	CPMK-02	20%		
2.	Praktik	CPL-04	CPMK-02	30%		
3.	Post-Test	CPL-04	CPMK-02	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

PRAKTIKUM 4: SINKRONISASI PROSES

Pertemuan ke : 4

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-04	Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya
CPMK-02	Mampu menjelaskan manajemen proses dan komunikasi antar proses, manajemen memori dan pengalokasian memori, implementasi system file dan prinsip kerja sistem I/O

4.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mengkompilasi kode sumber dengan thread yang berbagi akses ke area global yang tidak diproteksi.
2. Menunjukkan bahwa menulis ke area global yang tidak diproteksi dapat menimbulkan efek samping yang tidak diinginkan ketika diakses oleh thread.
3. Mengkompilasi kode sumber dengan thread yang tersinkronisasi.
4. Menunjukkan bahwa hardware menyediakan bantuan untuk membuat critical region.
5. Mengkompilasi kode sumber dengan mutex untuk membuat wilayah exclusion.
6. Menunjukkan bahwa OS menyediakan bantuan untuk membuat critical region.

4.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-04	CPMK-02	<p>Kemampuan mahasiswa:</p> <ol style="list-style-type: none"> 1. Dapat mengkompilasi kode sumber dengan thread yang berbagi akses ke area global yang tidak diproteksi. 2. Dapat menunjukkan bahwa menulis ke area global yang tidak diproteksi dapat menimbulkan efek samping yang tidak diinginkan ketika diakses oleh thread. 3. Dapat mengkompilasi kode sumber dengan thread yang tersinkronisasi.
--------	---------	---

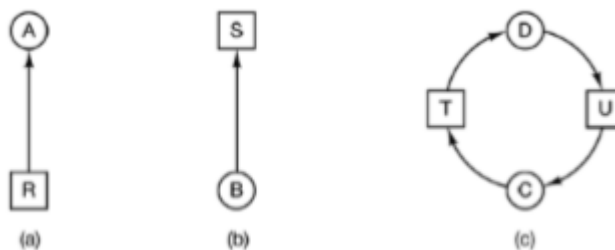
		<ol style="list-style-type: none"> 4. Dapat menunjukkan bahwa hardware menyediakan bantuan untuk membuat critical region. 5. Dapat mengkompilasi kode sumber dengan mutex untuk membuat wilayah exclusion. 6. Dapat menunjukkan bahwa OS menyediakan bantuan untuk membuat critical region
--	--	---

4.3. TEORI PENDUKUNG

Proses konkuren mengakses sumber daya global pada saat yang bersamaan dapat menghasilkan efek samping yang tidak diinginkan. Perangkat keras atau sistem operasi komputer dapat menyediakan mutual exclusion ketika sumber daya tersebut digunakan bersama.

DEADLOCK

Deadlock yang terjadi ketika proses-proses membutuhkan akses ke lebih dari satu sumber daya (yang tidak dapat dipakai bersama) sering terjadi pada sistem operasi modern.



Gambar 4. 1 Deadlock

Keterangan:

- a. Proses A sedang memakai (allocate) sumber daya R
- b. Proses B meminta (request) sumber daya S
- c. Proses C meminta sumber daya T yang sedang dipakai oleh D, namun D tidak (akan) melepas D karena sedang meminta sumber daya U yang dipakai oleh C.

OS dapat menerapkan berbagai teknik untuk menghindari, mencegah, atau mengatasi deadlock yang akan menyebabkan sistem berjalan se-efisien mungkin.

4.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. CPU Simulator

4.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-02	Jelaskan yang dimaksud dengan sinkronisasi proses!	30
2.	CPL-04	CPMK-02	Jelaskan pengertian deadlock beserta cara kerjanya!	40
3.	CPL-04	CPMK-02	Jelaskan yang anda ketahui tentang graf!	30

4.6. LANGKAH PRAKTIKUM

Aturan Penilaian (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.					

Langkah-Langkah Praktikum:

Asumsikan semua proses tiba pada waktu yang bersamaan
Jalankan Simulator.

1. Tuliskan kode berikut ini pada compiler.

```

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

```

Kode di atas akan membuat dua thread yaitu thread1 dan thread2. Masing-masing thread menaikkan nilai variabel global g dalam tiap loop. Ketika dua loop selesai nilai apa yang anda perkirakan dalam variabel g untuk dua kasus tersebut?

Dapatkan anda menduga fungsi statement **wait**?

- a. Sekarang compile kode tersebut, muatkan dalam memori.
 - b. Tampilkan jendela konsol, aktifkan STAY ON TOP.
 - c. Beralihlah ke OS simulator, akan tampak CriticalRegion pada PROGRAM LIST.
 - d. Buatlah proses dengan ketentuan : ROUND ROBIN, 10 ticks, kecepatan maksimum.
 - e. Tekan START.
 - f. Ketika program berhenti, catatlah dua nilai g yang ditampilkan. Apakah nilai ini seperti yang anda perkirakan tadi? Bagaimana pendapat anda.
2. Sekarang kita memodifikasi program seperti dibawah ini. Bagian yang dimodifikasi ditandai dengan tebal (bold).

```
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("thread1 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("thread2 g = ", g)
    writeln("Exiting thread2")
  end sub

  writeln("In main")

  call thread1
  call thread2

  wait

  writeln("Exiting main")
end
```

- a. Hapus kode sebelumnya dengan cara: pada jendela CPU Simulator, klik REMOVE ALL PROGRAMS.
- b. Compile kode diatas dan muatkan ke memori. Pada jendela compiler, carilah baris pada *program code (output)* yang menerjemahkan **synchronise**, catatlah kode assembler beserta keterangannya.
- c. Pada jendela OS simulator buatlah proses baru, kemudian jalankan.

- d. Tunggu hingga program selesai, sembari mengamati yang terjadi pada sub jendela waiting processes,
 - e. Catat dua nilai variabel g yang ditampilkan pada layar consol.
3. Kita modifikasi lagi kode sebelumnya. Bagian yang ditambahkan ditandai dengan tebal.

```

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("thread1 g = ", g)
    leave
    writeln("Exiting thread1")
  end sub

  sub thread2 as thread
    writeln("In thread2")
    enter
    g = 0
    for n = 1 to 12
      g = g + 1
    next
    writeln("thread2 g = ", g)
    leave
    writeln("Exiting thread2")
  end sub

  writeln("In main")

  call thread1
  call thread2

  wait
  writeln("Exiting main")
end

```

- a. Hapus program sebelumnya dari memori.
 - b. Compile kode diatas, muatkan ke memori. Amati hasil kompilasi perintah enter dan leave pada jendela compiler output
 - c. Beralihlah ke OS simulator, buat proses baru, kemudian jalankan.
 - d. Catat 2 nilai variabel g.
4. Jadi apa kesimpulannya? Untuk memahami percobaan diatas, jawablah pertanyaan-pertanyaan berikut ini:
- a. Jelaskan tujuan utama praktikum ini menurut anda.
 - b. Mengapa kita menggunakan variabel global (g) yang sama pada dua thread?

- c. Sudahkan kita menggunakan variabel lokal, dan apakah hasilnya akan berbeda? Lakukan eksperimen kecil dengan sedikit modifikasi pada kode yang ada dan jalankanlah program tersebut.
- d. Pada modifikasi yang pertama ditambahkan kata kunci synchronise. Jelaskan tujuan modifikasi ini. Beri contoh istilah untuk synchronise dalam bahasa pemrograman nyata.
- e. Pada modifikasi yang kedua digunakan kata kunci enter dan leave. Jelaskan fungsi modifikasi ini, dan apa bedanya dengan (d)?
- f. Critical regions seringkali diimplementasikan menggunakan semaphore dan mutex. Jelaskan pengertiannya dan apa perbedaannya.
- g. Berikan contoh nyata untuk suatu critical region (atau mutex region). Beri contoh nyata suatu mutex.
- h. Beberapa arsitektur komputer memiliki instruksi “test-and-set” untuk menerapkan critical region. Jelaskan bagaimana teknik ini bekerja dan mengapa penerapannya pada hardware.
- i. Jika hardware maupun OS tidak menyediakan bantuan, bagaimana anda dapat memproteksi critical region dalam kode anda? Berikan saran anda dan jelaskan dimana perbedaannya dengan metode-metode yang telah diuji coba kan diatas.

Langkah Praktikum Deadlock

1. Pada kertas, buatlah suatu siklus *resource allocation graph* untuk 4 proses dengan 4 sumber daya (bisa sumber daya apa saja). Hanya empat sumber daya dengan tiap proses menggunakan satu sumber daya dan meminta sumber daya yang lainnya. Kondisi ini akan menyebabkan deadlock.
 - a. Gambarkan graf untuk skenario ini dan jelaskan bagaimana deadlock dapat terjadi.
 - b. Berapa proses yang mengalami deadlock untuk kasus ini?

Jalankan Simulator.

2. Tuliskan kode berikut ini pada compiler.

```
Program Deadlocks
  while true
    n = 1
  wend
end
```

- a. Kode diatas akan menjalankan loop selamanya.
 - b. Compile program tersebut, muatkan ke memori.
 - c. Beralihlah ke OS simulator.
 - d. Sorot program Deadlocks pada PROGRAM LIST, klik tombol CREATE NEW PROCESS.
 - e. Pastikan menggunakan ROUND ROBIN dengan 15 ticks.
 - f. Maksimalkan kecepatan simulator. (**jangan klik START dulu**)
 - g. Kemudian ikuti langkah-langkah berikutnya.
3. Simulator dapat mengalokasikan dan men-dealokasikan beberapa sumber daya secara imajiner untuk proses sehingga kita dapat mempelajari permasalahan terkait pengalokasian sumber daya dan deadlock.

- Buatlah satu proses lagi sehingga akan ada dua proses pada antrian ready.
 - Biarkan berada pada antrian ready (**jangan klik START dulu**)
 - Klik tombol VIEW RESOURCES... pada jendela OS simulator, akan muncul jendela System Resources. Ada 4 sumber daya yang teridentifikasi yaitu R0, R1, R2, dan R3.
 - Gambar yang mewakili sumber daya berwarna hijau yang artinya sumber daya tersebut belum dialokasikan ke suatu proses.
 - Klik STAY ON TOP pada jendela resource.
 - Kemudian pilih proses pada antrian ready dan klik tombol *allocate*.
 - Akan tampak nomor ID proses pada kotak USED BY dan warna sumber daya akan berubah menjadi kuning.
 - Kemudian klik (ID) proses yang lain dan lakukan hal yang sama, maka nomor ID akan tampak dalam kotak daftar *requested by*, dan warna menjadi merah. Jelaskan yang terjadi.
 - Berlatihlah membuat kondisi deadlock untuk 2 sumber daya.
 - Sekarang, klik tombol RELEASE ALL untuk men-dealokasikan sumber daya. Sumber daya seharusnya menjadi hijau kembali dan nomor ID proses hilang.
 - Hapus proses pada antrian ready. Untuk melakukannya, klik tombol CLEAR pada jendela READY PROCESSES.
4. Setelah memahami mekanisme pengalokasian sumber daya menggunakan simulator, berikutnya buatlah 4 proses yang berkaitan dengan graf alokasi sumber daya yang tadi anda buat. Kemudian alokasikan sumber daya tersebut untuk menciptakan kondisi deadlock. Bagaimana hasilnya? Jika berhasil, apa yang anda lihat?
 5. Sekali deadlock terbentuk, cobalah menjalankan proses dengan klik pada tombol START. Jelaskan yang terjadi. Proses pada antrian ready belum mengalami deadlock, kejadian tersebut terjadi ketika proses telah berjalan.
 6. Dalam kondisi proses yang deadlock, apa yang dapat anda lakukan untuk mengatasinya? Berikan dua cara untuk mengatasinya, kemudian praktikkan menggunakan simulator.

4.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-03	Jalankan program dan screnshoot setiap prosesnya!	25
2.	CPL-04	CPMK-03	Jelaskan dan berikan kesimpulan terhadap hasil simulasi deadlock yang telah dilakukan!	75

4.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-04	CPMK-03	20%		
2.	Praktik	CPL-04	CPMK-03	30%		
3.	Post-Test	CPL-04	CPMK-03	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

PRAKTIKUM 5: BANKIR ALGORITHM FOR DEADLOCK

Pertemuan ke : 5

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-04	Mampu berpikir logis, kritis, sistematis dan inovatif, dan mampu mengambil keputusan secara tepat di bidang keahliannya
CPMK-02	Mampu menjelaskan manajemen proses dan komunikasi antar proses, manajemen memori dan pengalokasian memori, implementasi system file dan prinsip kerja sistem I/O

5.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

Mensimulasikan algoritma Bankir untuk tujuan penghindaran kebuntuan

5.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-04	CPMK-02	Kemampuan mahasiswa dapat mensimulasikan algoritma Bankir untuk tujuan penghindaran kebuntuan.
--------	---------	--

5.3. TEORI PENDUKUNG

Dalam lingkungan multi-pemrograman, beberapa proses dapat bersaing untuk sejumlah sumber daya yang terbatas. Suatu proses meminta sumber daya; jika sumber daya tidak tersedia pada waktu itu, proses memasuki keadaan menunggu. Kadang-kadang, proses menunggu tidak pernah lagi dapat mengubah keadaan, karena sumber daya yang diminta dimiliki oleh proses menunggu lainnya. Situasi ini disebut jalan buntu. Menghindari kebuntuan adalah salah satu teknik untuk menangani kebuntuan. Pendekatan ini mensyaratkan bahwa sistem operasi terlebih dahulu diberikan informasi tambahan mengenai sumber daya mana yang akan diminta dan digunakan oleh suatu proses selama masa pakainya.

Dengan pengetahuan tambahan ini, dapat memutuskan untuk setiap permintaan apakah proses harus menunggu atau tidak. Untuk memutuskan apakah permintaan saat ini dapat dipenuhi atau harus ditunda, sistem harus mempertimbangkan sumber daya yang tersedia saat ini, sumber daya yang saat ini dialokasikan untuk setiap proses, dan permintaan di masa mendatang dan pelepasan setiap proses. Algoritma Banker adalah algoritma penghindaran kebuntuan yang berlaku untuk sistem dengan banyak instance dari setiap jenis sumber daya.

5.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Pemrograman C

5.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-02	Jelaskan Pengertian dari Deadlock dalam Sistem Operasi!	30
2.	CPL-04	CPMK-02	Jelaskan cara kerja Algoritma Bankir!	40
3.	CPL-04	CPMK-02	Jelaskan Tujuan Algoritma Bankir untuk Deadlock!	30

5.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-04	CPMK-02	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

<pre>#include<stdio.h> struct file { int all[10]; int max[10]; int need[10]; int flag; }; void main() { struct file f[10]; int fl; int i, j, k, p, b, n, r, g, cnt=0, id, newr; int avail[10],seq[10]; clrscr(); printf("Enter number of processes -- "); scanf("%d",&n); printf("Enter number of resources -- "); scanf("%d",&r); for(i=0;i<n;i++) { printf("Enter details for P%d",i);</pre>

```

printf("\nEnter allocation\t -- \t");
for(j=0;j<r;j++)
    scanf("%d",&f[i].all[j]);
printf("Enter Max\t\t -- \t");
for(j=0;j<r;j++)
    scanf("%d",&f[i].max[j]);
f[i].flag=0;
}
printf("\nEnter Available Resources\t -- \t");
for(i=0;i<r;i++)
    scanf("%d",&avail[i]);
printf("\nEnter New Request Details -- ");
printf("\nEnter pid \t -- \t");
scanf("%d",&id);
printf("Enter Request for Resources \t -- \t");
for(i=0;i<r;i++) {
    scanf("%d",&newr);
    f[id].all[i] += newr;

    avail[i]=avail[i] - newr;
}

for(i=0;i<n;i++) {
for(j=0;j<r;j++) {
    f[i].need[j]=f[i].max[j]-f[i].all[j];
    if(f[i].need[j]<0)
        f[i].need[j]=0;
    }
}
cnt=0;
fl=0;
while(cnt!=n)
{
    g=0;
    for(j=0;j<n;j++) {
        if(f[j].flag==0) {
            b=0;
            for(p=0;p<r;p++)
            {
                if(avail[p]>=f[j].need[
                    p])
                    b=b+1;
            }
            else
                b=b-1;
        }
        if(b==r)
        {
            printf("\nP%d is visited",j);
            seq[fl++]=j;
            f[j].flag=1;
            for(k=0;k<r;k++)
                avail[k]=avail[k]+f[j].
                all[k];
            cnt=cnt+1;
            printf("(");
            for(k=0;k<r;k++)
                printf("%3d",avail[k]);
            printf(")");
            g=1;
        }
    }
}

```

```

        }
        if (g==0)
        {
            printf("\n REQUEST NOT GRANTED -- DEADLOCK
OCCURRED");
            printf("\n SYSTEM IS IN UNSAFE STATE");
            goto y;
        }
    }
    printf("\nSYSTEM IS IN SAFE STATE");
    printf("\nThe Safe Sequence is -- (");
    for(i=0;i<fl;i++)
    printf("P%d ",seq[i]);
    printf(")");
    y:
    printf("\nProcess\t\tAllocation\t\tMax\t\t\tNeed\n");
    for(i=0;i<n;i++) {
        printf("P%d\t",i);
        for(j=0;j<r;j++)
            printf("%6d",f[i].all[j]);

        for(j=0;j<r;j++)
            printf("%6d",f[i].max[j]);
        for(j=0;j<r;j++)
            printf("%6d",f[i].need[j]);
        printf("\n");
    }

    getch();
}

```

INPUT

Masukkan jumlah proses - 5

Masukkan jumlah sumber daya - 3

Masukkan detail untuk P0

Masukkan alokasi - 0 1 0

Masukkan Maks - 7 5 3

Masukkan detail untuk P1

Masukkan alokasi - 2 0 0

Masukkan Maks - 3 2 2

Masukkan detail untuk P2

Masukkan alokasi - 3 0 2

Masukkan Maks - 9 0 2

Masukkan detail untuk P3

Masukkan alokasi - 2 1 1

Masukkan Maks - 2 2 2

Masukkan detail untuk P4

Masukkan alokasi - 0 0 2

Masukkan Maks - 4 3 3

Masukkan Sumber Daya yang Tersedia - 3 3 2

Masukkan Detail Permintaan Baru –

Masukkan pid - 1

Masukkan Permintaan untuk Sumber Daya - 1 0 2

OUTPUT

P1 dikunjungi (5 3 2)

P3 dikunjungi (7 4 3)

P4 dikunjungi (7 4 5)

P0 dikunjungi (7 5 5)

P2 dikunjungi (10 5 7)

SISTEM DALAM KEADAAN AMAN

Safe Sequence adalah - (P1 P3 P4 P0 P2)

Process	Allocation	Max	Need
P0	0 1 0	7 5 3	7 4 3
P1	3 0 2	3 2 2	0 2 0
P2	3 0 2	9 0 2	6 0 0
P3	2 1 1	2 2 2	0 1 1
P4	0 0 2	4 3 3	4 3 1

5.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-04	CPMK-02	Analisislah cara kerja Deadlock dengan Algoritma Bankir!	30
2.	CPL-04	CPMK-02	Jelaskan Perbedaan dari Teknik Grafik Alokasi Sumber daya dengan Algoritma Bankir!	40
3.	CPL-04	CPMK-02	Dari perbandingan pada point nomor 2 jelaskan alasan anda konsep mana yang paling efektif untuk digunakan!	30

5.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

PRAKTIKUM 6: PENJADWALAN PROSES DAN ALOKASI MEMORI

Pertemuan ke : 6

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-05	Mampu mengkaji/menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-03	Mampu menerapkan teknik penjadwalan prosesor, metode evaluasi penjadwalan, pengalokasian data secara contiguous dan non contiguous

6.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Mendeskripsikan keadaan proses dan transisi akibat timeout.
2. Menjelaskan perbedaan antara penjadwalan pre-emptive dan non-pre-emptive.
3. Menelusuri nilai register PC dalam PCB suatu proses ketika berada dalam antrian Ready.
4. Menjelaskan pemanfaatan nilai Register PC pada penjadwalan Round Robin.
Menjelaskan page swapping ketika proses berpindah dari keadaan ready ke running

6.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-03	<p>Kemampuan mahasiswa:</p> <ol style="list-style-type: none"> 1. Dapat mendeskripsikan keadaan proses dan transisi akibat timeout. 2. Dapat menjelaskan perbedaan antara penjadwalan pre-emptive dan non-pre-emptive. 3. Dapat menelusuri nilai register PC dalam PCB suatu proses ketika berada dalam antrian Ready. 4. Dapat menjelaskan pemanfaatan nilai Register PC pada penjadwalan Round Robin.
--------	---------	---

		5. Dapat menjelaskan page swapping ketika proses berpindah dari keadaan ready ke running.
--	--	---

6.3. TEORI PENDUKUNG

Pada materi kuliah dibahas mengenai berbagai jenis penjadwalan proses dan manajemen memori. Praktikum ini berdasarkan materi tersebut.

6.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. CPU Simulator

6.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-03	Jelaskan apa yang dimaksud dengan manajemen memori dan manajemen proses!	40
2.	CPL-05	CPMK-03	Sebutkan 3 tipe penjadwalan proses pada system operasi!	30
3.	CPL-05	CPMK-03	Sebutkan 2 strategi penjadwalan proses pada system operasi!	30

6.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-03	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

Penjadwalan

1. Masukkan kode ini pada compiler :


```

program LoopForeverTest
  While true
    N = N + 1
  wend
end
      
```

 - a. Compile kode diatas, kemudian muatkan ke memori.
 - b. Beralihlah ke jendela OS Simulator.

- c. Pilih program LoopForeverTest, selanjutnya kita akan membuat 3 proses, namun sebelum tiap proses dibuat aturlah nilai LIFETIME dan pilih SECS untuk tiap proses: **10 seconds, 32 seconds, 6 seconds**.
 - d. Berikutnya memilih nilai timeslot melalui menu pull-down pada sub jendela SCHEDULER/POLICIES/RR Time Slice/Secs, pilih 4.
 - e. Tekan START dan tunggu hingga semua proses selesai.
 - f. Bukalah jendela OS ACTIVITY LOG dengan klik pada tombol VIEW LOG... Amati data log yang relevan kemudian perhatikan urutan pada proses yang berjalan (running processes) dan catat waktu yang dihabiskan oleh tiap proses selama dalam keadaan running.
2. Sekarang, beralihlah ke compiler dan masukkan kode berikut ini. Klik NEW untuk membuat tab editor baru.

```
program LoopForeverTest2
  while true
    n = n + 1
    i = i + n
    p = n + i + 5
  wend
end
```

- a. Compile kemudian muatkan ke memori kode diatas
- b. Beralihlah ke OS Simulator.
- c. Pada program list akan tampak 2 program: LoopForeverTest, dan LoopForeverTest2.
- d. Klik **LoopForeverTest** dan buatlah satu proses.
- e. Klik **LoopForeverTest2** dan buat satu proses.
- f. Sekarang pada antrian ready seharusnya ada 2 proses.
- g. Pilih penjadwalan ROUND ROBIN, tipe prioritas NON-PREEMPTIVE dan RR Time Slice adalah **10 tick**.
- h. Atur pada setengah kecepatan simulasi.

Lakukan langkah-langkah percobaan berikut ini:

1. Sorot proses kemudian klik tombol PCB... amati nilai pada PC REGISTERS (pada PCB Info) pada masing-masing proses, dan catatlah.
2. Klik **START** dan siapkan mouse anda pada tombol SUSPEND (jangan klik).
3. Sewaktu running process kembali ke ready queue, segera klik tombol SUSPEND.
4. Sorot proses pada antrian ready, dan klik tombol PCB... pada antrian ready. Catatlah nilai PC REGISTER nya.
5. Sekarang, beri check pada SUSPEND ON RUN dalam tampilan RUNNING PROCESS. Kurangi kecepatan OS simulation. Klik tombol RESUME dan amati ketika proses yang antri kembali dalam keadaan running, yang menyebabkan simulasi **berhenti (suspend) secara otomatis**.

6. Perhatikan baik-baik pada kotak PC REGISTER dalam **jendela CPU Simulator**. Sekarang klik RESUME pada OS Simulator. Catatlah nilai pada PC REGISTER ketika nilainya berubah. Berikan pendapat anda terhadap nilai tersebut dan jelaskan apa yang terjadi.

Manajemen Memori

1. Masih menggunakan ROUND-ROBIN (RR) dan NON-PREEMPTIVE pada OS Simulator.
2. Masukkan kode berikut ini pada compiler:

```
program ForeverLoop
  While true
    I = 1
  wend
end
```

compile kode diatas, kemudian muatkan ke memori. Beralihlah ke jendela OS Simulator.

3. Klik tombol VIEW MEMORY.... aktifkan STAY ON TOP. Matikan PAGING ENABLED (dengan menghilangkan tanda check). Pilih program ForeverLoop, buatlah proses dengan ketentuan berikut:

Proses	Page
P1	3
P2	5
P3	4
P4	2
P5	6

Perhatikan 4 proses yang terdapat pada jendela READY PROCESS.

Amati data pada kolom SWAP. Beberapa tertulis "Yes", jelaskan artinya. Bandingkanlah dengan tampilan *page* yang tampil pada jendela MAIN MEMORY.

4. Klik VIEW UTILIZATION... informasi apa yang anda lihat? Klik STAY ON TOP. Sekarang, pada jendela OS Simulator, atur kecepatan pada kisaran 90, tekan START. Amati hal-hal berikut ini dan buatlah catatan:
 1. Jelaskan kejadian yang tampak pada jendela MAIN MEMORY (RAM).
 2. Apa yang tampak pada kolom SWAP di jendela RUNNING PROCESSES? Mengapa selalu sama? Apakah ada kalanya berbeda dengan yang ditampilkan pada kolom SWAP di jendela READY PROCESSES? Jelaskan.
 3. Amati bagaimana proses-proses yang ready berganti (swap) keadaan dari waktu ke waktu. Beri penjelasannya.

4. Informasi apa yang anda peroleh dari jendela RESOURCE UTILISATION terkait dengan manajemen memori?

6.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-03	Jelaskan serta simpulkan perbedaan antara code sebelum di modifikasi dan sesudah di modifikasi!	100

6.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-05	CPMK-03	20%		
2.	Praktik	CPL-05	CPMK-03	30%		
3.	Post-Test	CPL-05	CPMK-03	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

PRAKTIKUM 7: SIMULASI PAGING

Pertemuan ke : 7

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-05	Mampu mengkaji/menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-02	Mampu menjelaskan manajemen proses dan komunikasi antar proses, manajemen memori dan pengalokasian memori, implementasi system file dan prinsip kerja sistem I/O

7.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
Mensimulasikan teknik paging manajemen memori

7.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-02	Kemampuan mahasiswa dapat mensimulasikan teknik paging manajemen memori
--------	---------	---

7.3. TEORI PENDUKUNG

Dalam sistem operasi komputer, paging adalah salah satu skema manajemen memori di mana komputer menyimpan dan mengambil data dari penyimpanan sekunder untuk digunakan dalam memori utama. Dalam skema manajemen memori paging, sistem operasi mengambil data dari penyimpanan sekunder dalam blok ukuran yang sama yang disebut halaman. Paging adalah skema manajemen memori yang memungkinkan ruang alamat fisik suatu proses menjadi tidak bersebelahan. Metode dasar untuk menerapkan paging melibatkan memecah memori fisik menjadi blok berukuran tetap yang disebut bingkai dan memecah memori logis menjadi blok dengan ukuran yang sama yang

disebut halaman. Ketika suatu proses akan dieksekusi, halamanhalamannya dimuat ke dalam setiap frame memori yang tersedia dari sumbernya.

Segmentasi merupakan skema manajemen memori yang mendukung cara pandang seorang programmer terhadap memori. Ruang alamat logika merupakan sekumpulan dari segmen-segmen. Masing-masing segment mempunyai panjang dan nama. Alamat diartikan sebagai nama segmen dan offset dalam suatu segmen. Jadi jika seorang pengguna ingin menunjuk sebuah alamat dapat dilakukan dengan menunjuk nama segmen dan offsetnya. Untuk lebih menyederhanakan implementasi, segmen-segmen diberi nomor yang digunakan sebagai pengganti nama segment. Sehingga, alamat logika terdiri dari dua tuple: [segment-number, offset]. Ada beberapa perbedaan antara Segmentasi dan Paging diantaranya adalah Segmentasi melibatkan programmer (programmer perlu tahu teknik yang digunakan), sedangkan dengan paging, programmer tidak perlu tahu teknik yang digunakan.

7.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Program C

7.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Apakah yang dimaksud dengan teknik alokasi memori paging dan segmentasi!	30
2.	CPL-05	CPMK-02	Jelaskan konsep dari teknik alokasi memori paging!	30
3.	CPL-05	CPMK-02	Jelaskan perbedaan dari teknik alokasi memori paging dengan segmentasi!	40

7.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-02	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

```
#include <stdio.h>
#include <conio.h>
int main() {

    int ukuran_memori, ukuran_halaman, jumlah_halaman_memori, jumlah_proses, sisa_halaman, i, j,
    nomor_proses, nomor_halaman, alamat_fisik, offset;
    int jumlah_halaman_proses[10], tabel_halaman[10][20];

    printf("\nMasukkan ukuran memori -- ");
    scanf("%d", &ukuran_memori);
```



```

printf("\nMasukkan ukuran halaman -- ");
scanf("%d", &ukuran_halaman);
jumlah_halaman_memori = ukuran_memori / ukuran_halaman;
printf("\nJumlah halaman yang tersedia dalam memori adalah -- %d ",
jumlah_halaman_memori);
printf("\nMasukkan jumlah proses -- ");
scanf("%d", &jumlah_proses);
sisal_halaman = jumlah_halaman_memori;

for (i = 0; i < jumlah_proses; i++)
{
    printf("\nMasukkan jumlah halaman yang dibutuhkan untuk p[%d]-- ", i + 1);
    scanf("%d", &jumlah_halaman_proses[i]);

    if (jumlah_halaman_proses[i] > sisal_halaman)
    {
        printf("\nMemori sudah penuh");
        break;
    }

    sisal_halaman = sisal_halaman - jumlah_halaman_proses[i];
    printf("\nMasukkan tabel halaman untuk p[%d] --- ", i + 1);
    for (j = 0; j < jumlah_halaman_proses[i]; j++)
        scanf("%d", &tabel_halaman[i][j]);
}

printf("\nMasukkan Alamat Logis untuk mencari Alamat Fisik");
printf("\nMasukkan nomor proses, nomor halaman, dan offset -- ");
scanf("%d %d %d", &nomor_proses, &nomor_halaman, &offset);
nomor_proses--;
nomor_halaman--;

if (nomor_proses >= jumlah_proses || nomor_halaman >= jumlah_halaman_proses[nomor_proses]
|| offset >= ukuran_halaman)
    printf("\nProses atau Nomor Halaman atau offset Tidak Valid");
else {
    alamat_fisik = tabel_halaman[nomor_proses][nomor_halaman] * ukuran_halaman + offset;
    printf("\nAlamat Fisiknya adalah -- %d", alamat_fisik);
}
getch();
return 0;
}

```

INPUT

Masukkan ukuran memori - 1000

Masukkan ukuran halaman - 100

Jumlah halaman yang tersedia di memori adalah - 10

Masukkan jumlah proses - 3

Masukkan jumlah halaman yang diperlukan untuk p [1] - 4

Masukkan pagetable untuk p [1] --- 8 6 9 5

Masukkan jumlah halaman yang diperlukan untuk p [2] - 5

Masukkan tabel halaman untuk p [2] --- 1 4 5 7 3

Masukkan jumlah halaman yang dibutuhkan untuk hal [3] - 5

OUTPUT

Memori Penuh

Masukkan Alamat Logis untuk menemukan

Alamat Fisik Masukkan jumlah proses dan jumlah tenaga dan offset - 2 3 60

7.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Buatlah program C++ untuk mensimulasikan teknik paging dua tingkat!	30
2.	CPL-05	CPMK-02	Buatlah program C++ untuk mensimulasikan teknik manajemen memori segmentasi!	30
3.	CPL-05	CPMK-02	Berikan analisis dan kesimpulan dari 2 program diatas!	40

7.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-05	CPMK-02	20%		
2.	Praktik	CPL-05	CPMK-02	30%		
3.	Post-Test	CPL-05	CPMK-02	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

PRAKTIKUM 8: I/O MANAGEMENT (SPOOLING)

Pertemuan ke : 8

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-05	Mampu mengkaji/menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-02	Mampu menjelaskan manajemen proses dan komunikasi antar proses, manajemen memori dan pengalokasian memori, implementasi system file dan prinsip kerja sistem I/O

8.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

Mensimulasikan dan memahami teknik *Spooling* pada Input dan Output Manajemen.

8.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-02	Kemampuan mahasiswa apat mensimulasikan teknik alokasi memori yang berdekatan berikut ini
--------	---------	---

8.3. TEORI PENDUKUNG

Manajemen I/O sering disebut device manager. Menyediakan “device driver” yang umum sehingga operasi I/O dapat seragam (membuka, membaca, menulis, menutup). Contoh: pengguna menggunakan operasi yang sama untuk membaca file pada SSD, hard-disk, CD-ROM dan floppy disk

Spooling adalah proses transfer data dengan menempatkannya pada temporary area dimana program lain dapat mengaksesnya nanti. Contoh: mencetak dokumen, prosesor akan menempatkan data yang akan dicetak ke temporary area, kemudian akan dibaca oleh printer untuk kemudian dicetak

Spooling sering juga disebut melakukan penjadwalan pemakaian I/O sistem supaya lebih efisien.

Beberapa fungsi manajemen input/output (I/O):

1. Mengirim perintah ke perangkat I/O agar menyediakan layanan.
2. Menangani interupsi perangkat I/O.
3. Menangani kesalahan perangkat I/O.
4. Menyediakan interface ke pemakai.

8.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Program C/C++

8.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Jelaskan pengertian dari teknik <i>Spooling</i> !	30
2.	CPL-05	CPMK-02	Sebutkan beberapa fungsi manajemen I/O!	30
3.	CPL-05	CPMK-02	Apa yang dimaksud dengan "buffering" dalam konteks manajemen I/O dan bagaimana ini berhubungan dengan teknik Spooling?	40

8.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

```
#include <iostream>
#include <fstream>
using namespace std;

int main() {
    // Open an output file for spooling
    ofstream outputFile("output.txt");

    if (!outputFile) {
        cerr << "Failed to open output file." << endl;
        return 1;
    }
}
```

```

// Spool data to the output file
for (int i = 1; i <= 10; ++i) {
    outputFile << "Line " << i << endl;
}

// Close the output file
outputFile.close();

// Open an input file for spooling
ifstream inputFile("output.txt");

if (!inputFile) {
    cerr << "Failed to open input file." << endl;
    return 1;
}

// Read and display data from the input file
string line;
while (getline(inputFile, line)) {
    cout << "Read from file: " << line << endl;
}

// Close the input file
inputFile.close();

return 0;
}

```

8.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Analisis dari I/O Management dari teknik Spooling yang kalian pelajari dari langkah praktikum, apakah sudah efisien?	100

8.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

PRAKTIKUM 9: MANAGEMENT FILE

Pertemuan ke : 9

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-05	Mampu mengkaji/menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-02	Mampu menjelaskan manajemen proses dan komunikasi antar proses, manajemen memori dan pengalokasian memori, implementasi system file dan prinsip kerja sistem I/O

9.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

Mensimulasikan teknik paging manajemen memori

9.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-02	Kemampuan mahasiswa dapat mensimulasikan teknik paging manajemen memori
--------	---------	---

9.3. TEORI PENDUKUNG

Command Prompt atau lebih dikenal dengan CMD adalah salah satu aplikasi *command line interpreter* (CLI) yang ada di sistem operasi Windows. Perintah CMD berfungsi untuk memberikan berbagai perintah kepada komputer tanpa perlu menavigasi program berbasis GUI seperti File Explorer, Control Panel, dan sebagainya.

CMD sudah ada sejak masa sistem operasi MS-DOS. Namun seiring perkembangan Windows, pengguna semakin jarang menggunakan perintah CMD karena adanya user interface (UI) yang lebih mudah digunakan. Hasilnya, sebagian besar pengguna Windows saat ini kurang familiar dengan perintah CMD. Meskipun demikian, perintah CMD juga memiliki beberapa keunggulan.

File adalah unit dasar dalam penyimpanan data, sedangkan direktori adalah wadah yang digunakan untuk mengorganisasi file. Dalam CMD, file direferensikan dengan nama dan ekstensi tertentu, sedangkan direktori adalah wadah yang berisi file dan direktori lainnya.

Jalur adalah alamat atau lokasi unik dari suatu file atau direktori dalam sistem file. Jalur dapat absolut (mulai dari akar drive) atau relatif (berdasarkan direktori aktif saat ini).

CMD adalah antarmuka baris perintah yang digunakan untuk berinteraksi dengan sistem operasi, termasuk untuk manajemen File. Setiap file memiliki atribut yang mencakup informasi seperti nama, tipe, ukuran, tanggal pembuatan, tanggal modifikasi, dan izin akses. Ini penting dalam manajemen file, terutama ketika pengguna perlu memeriksa atau mengubah atribut file.

1. Perintah cd untuk mengakses atau mengelola file dalam struktur direktori.
2. Perintah dir untuk mencari dan melihat daftar file dan direktori dalam direktori saat ini yang menyesuaikan output dengan berbagai opsi.
3. Perintah findstr untuk mencari teks dalam file.
4. Operasi dasar pada file seperti menyalin (copy), memindahkan (move), menghapus (del), dan mengganti nama (ren) dapat dilakukan. Membuat cadangan file (copy) dan, jika diperlukan, mengembalikannya dari cadangan jika file asli terhapus atau rusak.
5. Perintah type untuk melihat isi file teks dalam CMD, berguna ketika Pengguna perlu membaca atau memeriksa file teks tanpa membukanya di aplikasi teks terpisah.

9.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Program C

9.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	1. Sebutkan dan jelaskan 5 perintah CMD untuk file?	50
2.	CPL-05	CPMK-02	2. Apa fungsi perintah 'dir' dalam CMD?	25
3.	CPL-05	CPMK-02	3. Apa perintah CMD yang digunakan untuk menyalin file dari satu lokasi ke lokasi lain?	25

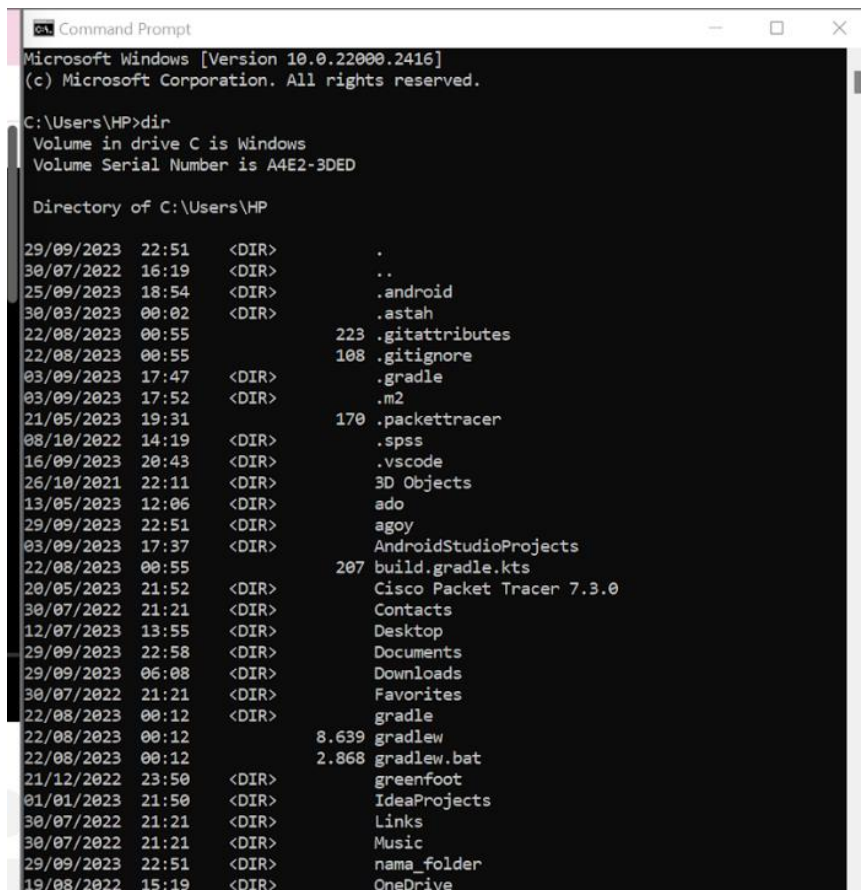
9.6. LANGKAH PRAKTIKUM

Aturan Penilaian (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-05	CPMK-02	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

1. Untuk melihat isi folder, ketik **dir** lalu **ENTER**.



```

Command Prompt
Microsoft Windows [Version 10.0.22000.2416]
(c) Microsoft Corporation. All rights reserved.

C:\Users\HP>dir
Volume in drive C is Windows
Volume Serial Number is A4E2-3DED

Directory of C:\Users\HP

29/09/2023  22:51  <DIR>          .
30/07/2022  16:19  <DIR>          ..
25/09/2023  18:54  <DIR>          .android
30/03/2023  00:02  <DIR>          .astah
22/08/2023  00:55             223 .gitattributes
22/08/2023  00:55             108 .gitignore
03/09/2023  17:47  <DIR>          .gradle
03/09/2023  17:52  <DIR>          .m2
21/05/2023  19:31             170 .packetracer
08/10/2022  14:19  <DIR>          .spss
16/09/2023  20:43  <DIR>          .vscode
26/10/2021  22:11  <DIR>          3D Objects
13/05/2023  12:06  <DIR>          ado
29/09/2023  22:51  <DIR>          agoy
03/09/2023  17:37  <DIR>          AndroidStudioProjects
22/08/2023  00:55             207 build.gradle.kts
20/05/2023  21:52  <DIR>          Cisco Packet Tracer 7.3.0
30/07/2022  21:21  <DIR>          Contacts
12/07/2023  13:55  <DIR>          Desktop
29/09/2023  22:58  <DIR>          Documents
29/09/2023  06:08  <DIR>          Downloads
30/07/2022  21:21  <DIR>          Favorites
22/08/2023  00:12  <DIR>          gradle
22/08/2023  00:12             8.639 gradlew
22/08/2023  00:12             2.868 gradlew.bat
21/12/2022  23:50  <DIR>          greenfoot
01/01/2023  21:50  <DIR>          IdeaProjects
30/07/2022  21:21  <DIR>          Links
30/07/2022  21:21  <DIR>          Music
29/09/2023  22:51  <DIR>          nama_folder
19/08/2022  15:19  <DIR>          OneDrive

```

Gambar 9.1 Tampilan setelah dijalankan untuk melihat isi folder

2. Untuk pindah dari suatu drive ke drive lainnya, ketik **d:** lalu **ENTER**.



```

19/08/2022  15:19  <DIR>          OneDrive
25/09/2023  09:33  <DIR>          Pictures
30/07/2022  21:21  <DIR>          Saved Games
30/07/2022  21:21  <DIR>          Searches
22/08/2023  00:55             348 settings.gradle.kts
22/12/2022  19:48  <DIR>          test
30/07/2022  21:21  <DIR>          Videos

7 File(s)          12.563 bytes
30 Dir(s)          3.693.408.256 bytes free

C:\Users\HP>d:
D:\>

```

Gambar 9.2 Pindah dari drive C: ke drive D:

3. Kemudian ketik lagi **dir** untuk melihat isi direktori.

```

Command Prompt
D:\> dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\

03/12/2021  13:57           149.033  1.pdf
03/12/2021  13:58           57.474  1.xlsx
03/05/2023  19:13       133.623  143210039_Rona Farida_EP-B_Keuangan Internasional
1_J-Curve.pdf
21/03/2023  16:04       114.570  143210039_Rona Farida_EP-B_Keuangan Internasional
1_Kuis1.pdf
03/12/2021  13:56       175.591  2.pdf
21/03/2023  23:15       380.474  2100018333_Dwi Priambodo Prayoga_Post1.pdf
03/12/2021  13:57       114.045  3.pdf
29/09/2023  23:05      <DIR>      333
05/04/2022  07:47      <DIR>      3uTools
05/12/2022  04:00      <DIR>      apex
19/12/2022  12:41        60.347  basdat10_336.sql
02/08/2022  21:22        19.511  biaya.docx
15/09/2023  22:59      <DIR>      bola
30/09/2023  21:19         18      coba
09/08/2023  14:27      <DIR>      couse my skill
25/09/2023  18:55      <DIR>      DICODING

```

Gambar 9.3 Isi dari drive d

4. Jika ingin masuk kedalam folder di dalam drive, ketik **cd folder tujuan** lalu **ENTER**.

```

30/09/2023  22:33      <DIR>      PrakSO
25/09/2023  08:34      <DIR>      praktikum
12/08/2022  20:03       420.710  proposal matic kra.pdf
02/08/2022  17:56        20.570  proposal.docx
03/08/2022  21:58       157.307  proposal.pdf
15/09/2023  22:55      <DIR>      sertifikat
16/03/2023  06:45      <DIR>      smstr 4
08/09/2023  12:57      <DIR>      TA--PBO-Penyewaan-playstation-main
23 File(s)      4.295.214 bytes
27 Dir(s)  318.147.985.408 bytes free

D:\>cd 333
D:\333>

```

Gambar 9.4 cd 333 untuk masuk ke dalam folder 333

Pada Langkah sebelumnya posisi kita berada pada folder 333

5. Jika ingin pindah folder maka ketik **cd\folder tujuan** lalu **ENTER**.

```

23 File(s)      4.295.214 bytes
27 Dir(s)  318.147.985.408 bytes free

D:\>cd 333
D:\333>cd\apex
D:\apex>

```

Gambar 9.5 Pindah dari folder 333 ke folder apex.

6. Kemudian anda akan membuat folder di dalam folder, caranya ketikan **md nama_folder** lalu **ENTER**.

```
D:\>cd 333
D:\333>cd\apex
D:\apex>md latihan
D:\apex>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex

30/09/2023  22:50    <DIR>          .
30/09/2023  22:50    <DIR>          latihan
               0 File(s)                0 bytes
               2 Dir(s)  318.147.981.312 bytes free

D:\apex>
```

Gambar 9.6 Membuat folder Latihan di dalam folder apex

7. Jika ingin masuk ke dalam folder yang telah anda buat, perintahkan **cd nama folder** lalu ENTER.

```
30/09/2023  22:50    <DIR>          .
30/09/2023  22:50    <DIR>          latihan
               0 File(s)                0 bytes
               2 Dir(s)  318.147.981.312 bytes free

D:\apex>cd latihan
D:\apex\latihan>
```

Gambar 9.7 Perintah cd latihan untuk masuk kedalam folder didalam folder apex

Posisi kita berada pada D:\apex\latihan

Kemudian bagaimana caranya membuat file di dalam folder?

8. Membuat isi file di dalam folder dengan perintah **copy con nama_file.txt** (untuk membuat file dengan format .txt atau teks pada notepad) lalu **ENTER**.
9. Kemudian isilah file tersebut, lalu ketik **CTRL + Z** untuk menyimpan.

```
2 Dir(s)  318.147.981.312 bytes free

D:\apex>cd latihan
D:\apex\latihan>copy con catatan.txt
ini adalah catatan^Z
               1 file(s) copied.

D:\apex\latihan>
```

Gambar 9.8 Membuat isi file didalam folder dengan nama catatan.txt

10. Lalu bagaimana jika ingin meng-copy atau menduplikatkan suatu file?
Caranya dengan perintah **copy con nama_file d:\folder_tujuan** lalu **ENTER**.

```

2 Dir(s) 318.147.981.312 bytes free

D:\apex>cd latihan

D:\apex\latihan>copy con catatan.txt
ini adalah catatan^Z
1 file(s) copied.

D:\apex\latihan>copy catatan.txt d:
The file cannot be copied onto itself.
0 file(s) copied.

D:\apex\latihan>copy catatan.txt d:\apex
1 file(s) copied.

D:\apex\latihan>

```

Gambar 9.9 Perintah copy catatan.txt d:\ apex

11. Jika ingin kembali ke folder sebelumnya, ketikkan **cd..**.

```

2 Dir(s) 318.147.981.312 bytes free

D:\apex>cd latihan

D:\apex\latihan>copy con catatan.txt
ini adalah catatan^Z
1 file(s) copied.

D:\apex\latihan>copy catatan.txt d:
The file cannot be copied onto itself.
0 file(s) copied.

D:\apex\latihan>copy catatan.txt d:\apex
1 file(s) copied.

D:\apex\latihan>cd..

D:\apex>

```

Gambar 9.10 Perintah cd..

Posisi kita berada pada folder apex

12. Perintah **dir**

```

D:\apex\latihan>cd..

D:\apex>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex

30/09/2023 23:01 <DIR>      .
30/09/2023 22:56          18 catatan.txt
30/09/2023 22:56 <DIR>      latihan
                1 File(s)      18 bytes
                2 Dir(s) 318.147.981.312 bytes free

D:\apex>

```

Gambar 9.11 Melihat isi folder

13. Kemudian masuk lagi ke dalam folder di dalam folder dengan perintah **cd folder_tujuan** lalu **ENTER**.

```

D:\apex>cd latihan

D:\apex\latihan>

```

Gambar 9.12 Masuk ke dalam folder latihan di dalam folder apex

Posisi kita berada pada direktori D:\apex\latihan

14. Kali ini anda akan memindahkan suatu file ke dalam folder lain dengan perintah **move nama_file d:\folder tujuan**, lalu **ENTER**.

```
D:\apex>cd latihan
D:\apex\latihan>move catatan.txt d:\333
1 file(s) moved.
D:\apex\latihan>
```

Gambar 9. 13 Move catatan.txt d:\333 untuk memindahkan file catatan.txt ke folder 333.

15. Untuk melihat hasilnya, masuk dahulu ke dalam folder tersebut dengan perintahkan **cd\nama_folder** untuk ke folder 333.
16. Lalu perintah **dir** untuk melihat isi folder tersebut.

```
D:\apex\latihan>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex\latihan

30/09/2023  23:11    <DIR>          .
30/09/2023  23:01    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  318.147.981.312 bytes free

D:\apex\latihan>cd\333
D:\333>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\333

30/09/2023  23:11    <DIR>          .
30/09/2023  22:56             18 catatan.txt
               1 File(s)              18 bytes
               1 Dir(s)  318.147.981.312 bytes free
```

Gambar 9. 14 Perintah cd\333 untuk pindah dari folder apex ke folder 333

17. Untuk melihat isi dari suatu file caranya dengan perintah **type nama_file** lalu **ENTER**.

```
D:\333>type catatan.txt
ini adalah catatan
D:\333>
```

Gambar 9. 15 Perintah type catatan.txt dan "ini adalah catatan" adalah isi dari file tersebut

18. Kemudian perintah **cd..** untuk kembali ke folder sebelumnya, lalu ketik **cd nama_folder** untuk masuk ke dalam folder, lalu **ENTER**.

```
D:\333>cd..
D:\>cd latihan
The system cannot find the path specified.
D:\>cd apex
```

Gambar 9. 16 Kembali ke folder Sebelumnya, lalu masuk ke folder apex

Posisi kita sekarang sudah berada pada folder apex

19. Lalu bagaimana caranya untuk mengganti nama suatu folder?

Dengan perintah **rename nama_folder_sebelumnya nama_folder_setelahnya**.

```
D:\>cd apex

D:\apex>rename latihan latihan2

D:\apex>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex

30/09/2023  23:23    <DIR>          .
30/09/2023  22:56                18 catatan.txt
30/09/2023  23:11    <DIR>          latihan2
               1 File(s)              18 bytes
               2 Dir(s)  318.147.981.312 bytes free
```

Gambar 9. 17 Pergantian nama dari yang sebelumnya latihan menjadi latihan2

20. Untuk menghapus suatu folder perintahkan **rmdir nama_folder**

Lalu ketik **dir** untuk melihat isi folder.

```
D:\apex>rmdir latihan2

D:\apex>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex

30/09/2023  23:31    <DIR>          .
30/09/2023  22:56                18 catatan.txt
               1 File(s)              18 bytes
               1 Dir(s)  318.147.981.312 bytes free
```

Gambar 9. 18 Folder latihan2 sudah terhapus

21. Kemudian bagaimana jika ingin menghapus file?

Untuk menghapus file menggunakan perintah **del nama_file**, lalu **ENTER**.

```
D:\apex>del catatan.txt

D:\apex>dir
Volume in drive D is New Volume
Volume Serial Number is F45E-B20B

Directory of D:\apex

30/09/2023  23:37    <DIR>          .
               0 File(s)                0 bytes
               1 Dir(s)  318.147.981.312 bytes free
```

Gambar 9. 19 Dengan perintah del catatan.txt maka file catatan.txt sudah terhapus

22. Perintah **tree** untuk menampilkan path subdirectory pada masing-masing directory yang ada atau jalurnya.

```

D:\>tree
Folder PATH listing for volume New Volume
Volume serial number is F45E-B208
D:..
|-- 333
|-- 3uTools
|-- apex
|   |-- builder
|   |-- core
|   |-- modules
|   |   |-- auto_backup
|   |   |-- catalog
|   |   |-- create_app_wiz
|   |   |-- issues
|   |   |-- quicksql
|   |   |-- rest_ws
|   |-- packaged_apps
|   |-- scripts
|   |-- themes
|   |-- images
|   |   |-- apex
|   |   |   |-- builder
|   |   |   |-- graphics
|   |   |-- apex_ui
|   |   |   |-- css
|   |   |   |-- img
|   |   |   |   |-- favicons
|   |   |   |   |-- icons
|   |   |   |   |   |-- builder
|   |   |   |   |   |-- pkg-apps
|   |   |   |   |   |-- wizard
|   |   |   |-- legacy
|   |   |   |-- rw
|   |   |   |   |-- avatars
|   |   |   |   |-- icons
|   |   |   |   |-- textures
|   |   |   |-- ui
|   |   |-- js
|   |   |   |-- minified
|   |   |   |-- staticData

```

Gambar 9. 20 Terlihat jalur directory ke sub directory

9.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Buat folder yang berisi 2 file.txt; - nama.txt yang berisi NIM Anda - nim.txt yang berisikan Nama Anda	30
2.	CPL-05	CPMK-02	Duplikatkan file nama.txt ke folder lain	20
3.	CPL-05	CPMK-02	Pindahkan file nim.txt ke folder tersebut	20
4.	CPL-05	CPMK-02	Rename file nim.txt menjadi nama_nim.txt	20
5.	CPL-05	CPMK-02	Kemudian lihatlah menggunakan perintah CMD	10

9.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

PRAKTIKUM 10: SISTEM KEAMANAN DAN PROTEKSI

Pertemuan ke : 10

Total Alokasi Waktu : 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-05	Mampu mengkaji/menganalisis implikasi pengembangan atau implementasi ilmu pengetahuan teknologi, menyusun deskripsi saintifik hasil kajian untuk pemecahan masalah dengan mempertimbangkan multidisiplin ilmu
CPMK-02	Mampu menjelaskan sistem keamanan dan proteksi pada sistem operasi yang dipakai.

10.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

Menerapkan sistem keamanan dan proteksi pada sistem operasi.

10.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-05	CPMK-02	Kemampuan mahasiswa untuk mengimplementasikan langkah penanganan keamanan dan proteksi sistem operasi.
--------	---------	--

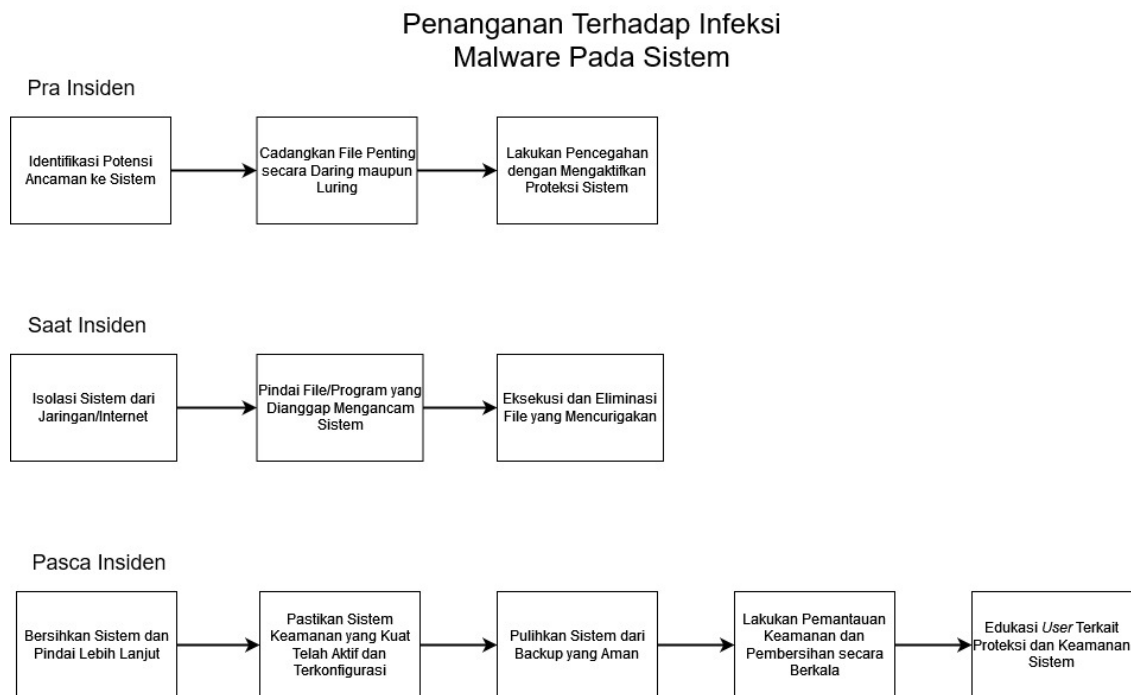
10.3. TEORI PENDUKUNG

Sistem keamanan adalah cara, mekanisme, dan praktik untuk melindungi data dari ancaman, risiko, dan serangan oleh pihak tidak berkepentingan yang dapat merusak integritas, kerahasiaan, dan ketersediaan data pada sistem. Proteksi sebagai mekanisme dalam sistem operasi akan mengendalikan akses ke objek yang melindungi informasi pada perangkat dengan mengatur akses oleh *program*, *processor*, atau *user* terhadap sumber daya sistem.

Keamanan sistem operasi melibatkan berbagai ancaman, termasuk akses ilegal, serangan, dan malware, serta risiko bencana alam atau kesalahan pengguna. Ancaman dapat berasal dari aktivitas manusia seperti *hacking* atau peretasan sistem operasi untuk tujuan

pribadi, dan juga dari jenis malware seperti virus, keylogger, worm, trojan, dan spyware. Selain itu, sistem operasi dan data rentan terhadap kerusakan perangkat keras atau perangkat lunak serta kesalahan *user*.

Untuk mengatasi hal tersebut, harus dilakukan aksi antisipasi dan penanggulangan terkait proteksi sistem operasi, meliputi penggunaan antivirus, firewall, dan sebagainya.



Gambar 10. 1 Diagram Penanganan Malware Berdasarkan Waktu

Antivirus melindungi dari infeksi malware atau program berbahaya. Cara kerjanya dengan mendeteksi, mengidentifikasi, dan menghapus atau menonaktifkan malware yang mengganggu keamanan sistem. Namun, terkadang ini bisa menjadi kendala saat menginstal program yang sebenarnya aman. Antivirus juga dapat menggantikan firewall komputer yang bawaan, yang terkadang lemah. Fungsi lainnya adalah mendeteksi ancaman virus di dalam email, meng-uninstal program, melindungi komputer saat menghubungkan perangkat lain, dan mengunci aplikasi untuk menjaga keamanan dari akses yang tidak sah.

Firewall menetapkan kebijakan kendali akses antara dua jaringan. Secara prinsip, firewall dapat dianggap sebagai sepasang mekanisme: memblokir lalu lintas dan mengizinkan lalu lintas jaringan. Firewall melindungi jaringan anda dari serangan oleh pihak luar, namun firewall tidak dapat melindungi dari serangan yang tidak melalui firewall dan serangan dari seseorang yang berada di dalam jaringan yang sama, serta firewall tidak dapat melindungi dari program-program aplikasi yang ditulis dengan buruk.

Secara konseptual, terdapat dua macam firewall yaitu:

- *Network Level*, mendasarkan keputusannya pada alamat sumber, alamat tujuan dan port yang terdapat dalam setiap paket IP.
- *Application level*, melaporkan lebih rinci dan memeriksa setiap detail yang sangat rinci tentang aktivitas *user* dibanding network level firewall. Firewall ini bisa dikatakan sebagai jembatan.

Windows Defender Firewall menyediakan kemampuan firewall yang dapat diatur dalam profil domain dan profil pribadi, sehingga memungkinkan kontrol lalu lintas berdasarkan aplikasi (firewall tipe aplikasi) dan juga berdasarkan port dan protokol (firewall tingkat jaringan).

10.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Windows Security
3. Windows Defender Firewall
4. Website VirusTotal

10.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Apa yang Anda ketahui tentang proteksi sistem operasi?	25
2.	CPL-05	CPMK-02	Jelaskan langkah-langkah yang dapat Anda ambil untuk melindungi privasi Anda saat menggunakan sistem operasi, termasuk melindungi data pribadi Anda.	50
3.	CPL-05	CPMK-02	Jika Anda mencurigai bahwa komputer Anda telah terinfeksi oleh malware, sebutkan langkah-langkah pertama yang akan Anda ambil untuk mengatasinya.	25

10.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

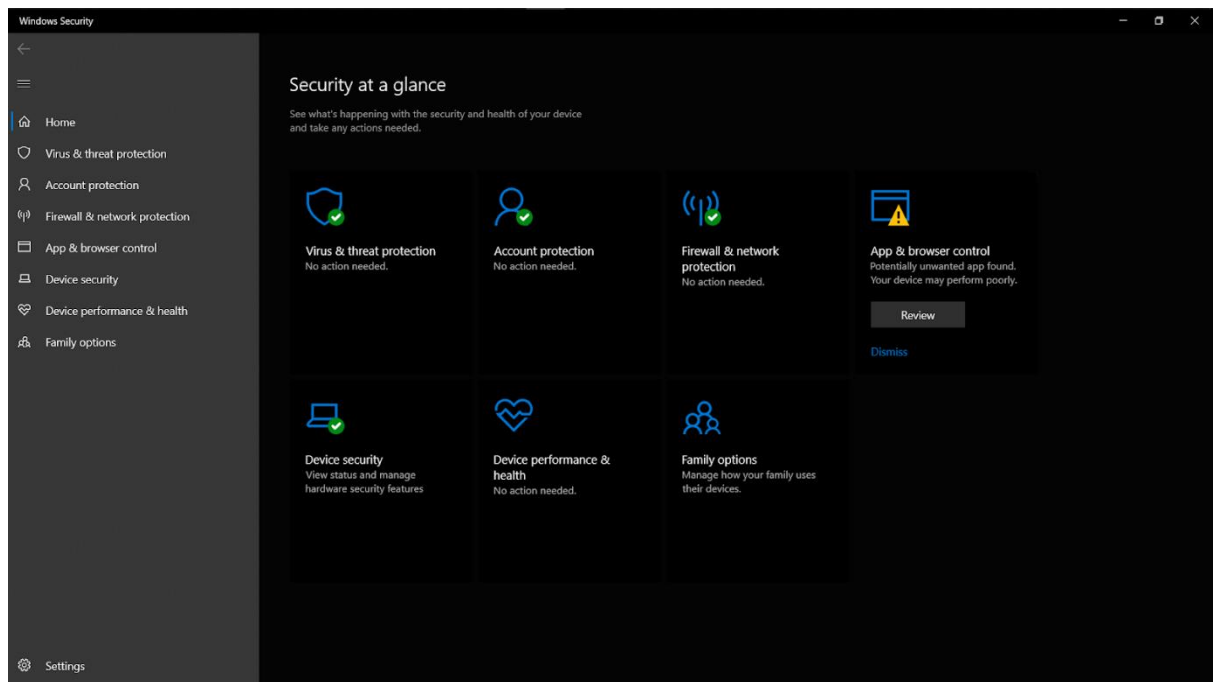
No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

1. Windows Security

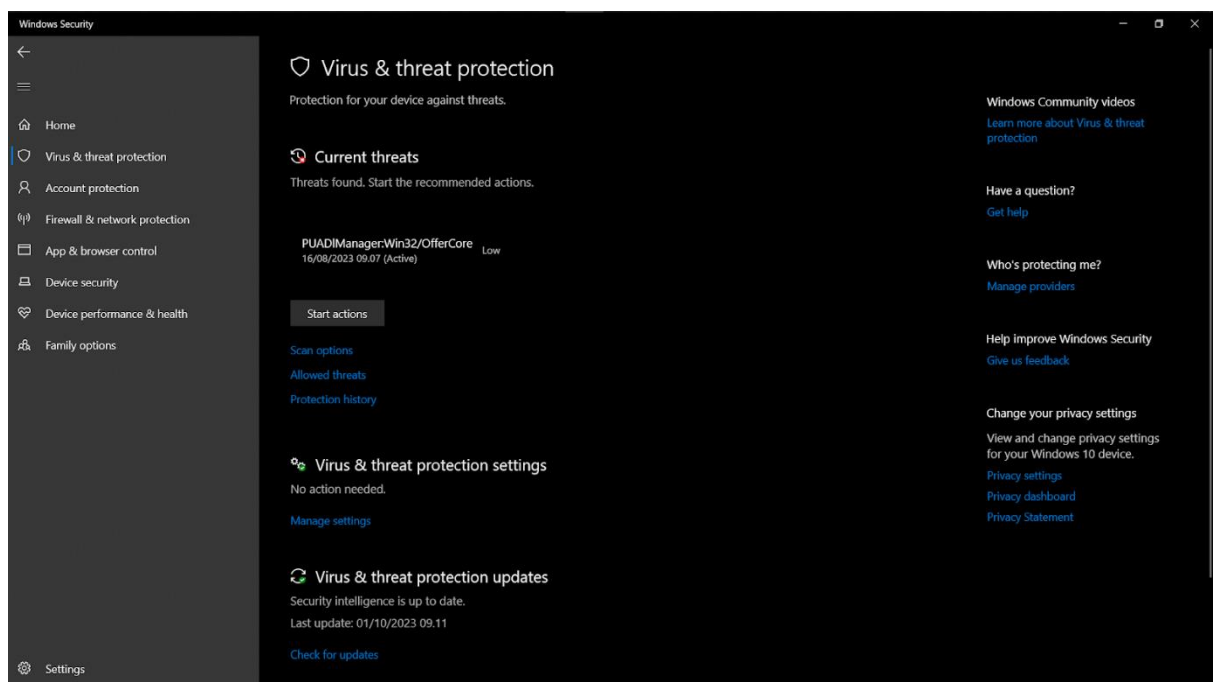
Aplikasi untuk mendeteksi, menghapus, dan mengarantina file berbahaya, lalu dianalisis.

- Jika Windows Security mendeteksi ancaman, maka akan terlihat logo peringatan pada bagian yang terdeteksi tersebut.

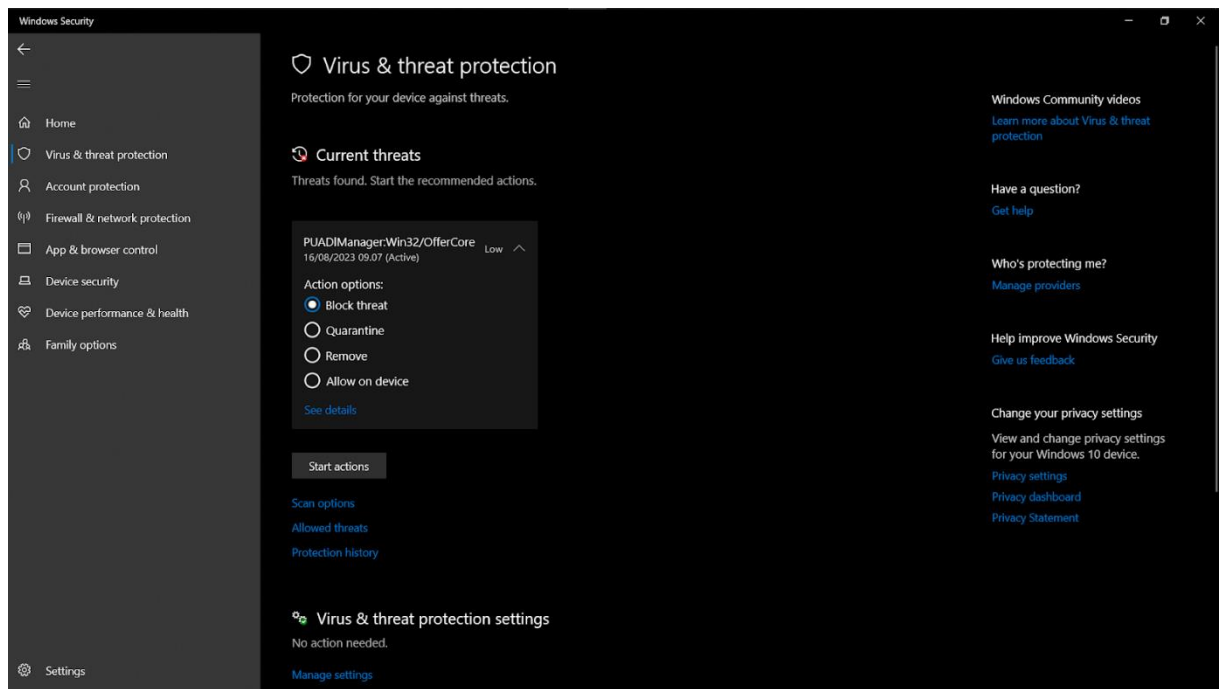


Gambar 10. 2 Tampilan Awal Windows Security

- Untuk menangani ancaman, klik opsi "Start actions", kemudian akan ada pilihan untuk mengisolasi ancaman, menghapusnya, atau tindakan lain yang sesuai dengan ancaman yang terdeteksi.

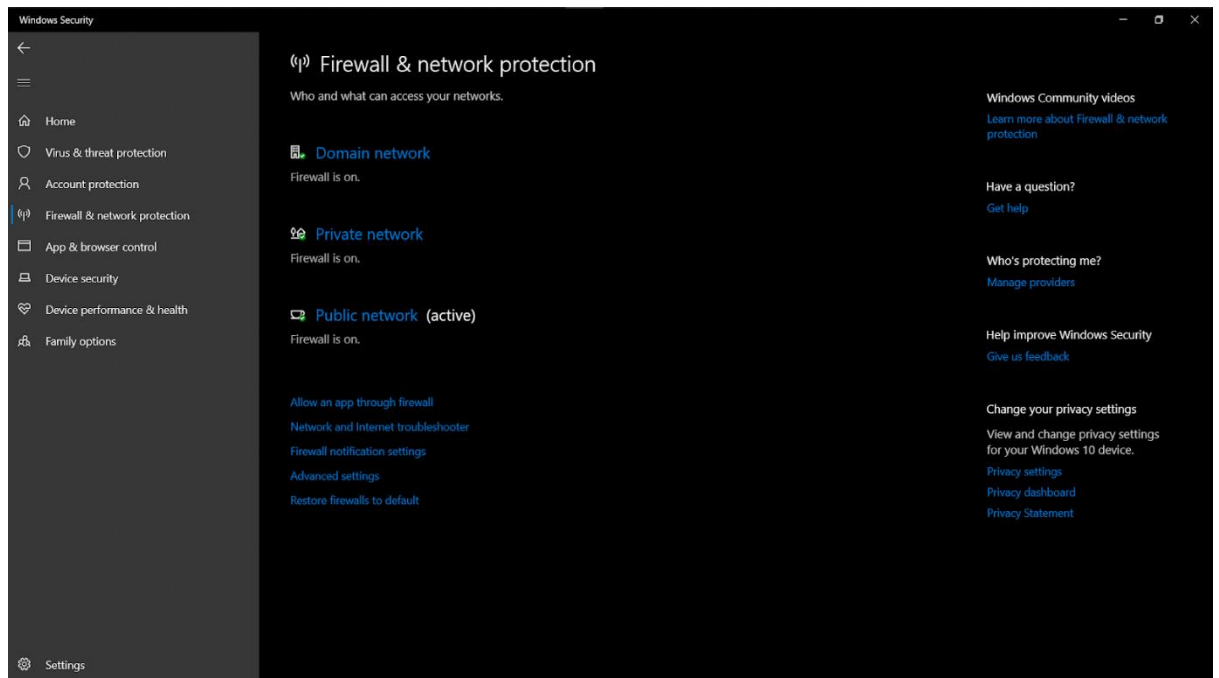


Gambar 10. 3 Contoh Ketika Threat (ancaman) Terdeteksi



Gambar 10. 4 Opsi Penanganan File Threat

- Selain penanganan virus dan ancaman, Windows Security juga menyediakan fitur Firewall. Di bagian ini, terdapat tiga jenis jaringan: "Domain network" (Jaringan domain), "Private network" (Jaringan pribadi), dan "Public network" (Jaringan publik) dimana tiap jaringannya memiliki pengaturan firewall sendiri.



Gambar 10. 5 Fitur Firewall

2. VirusTotal

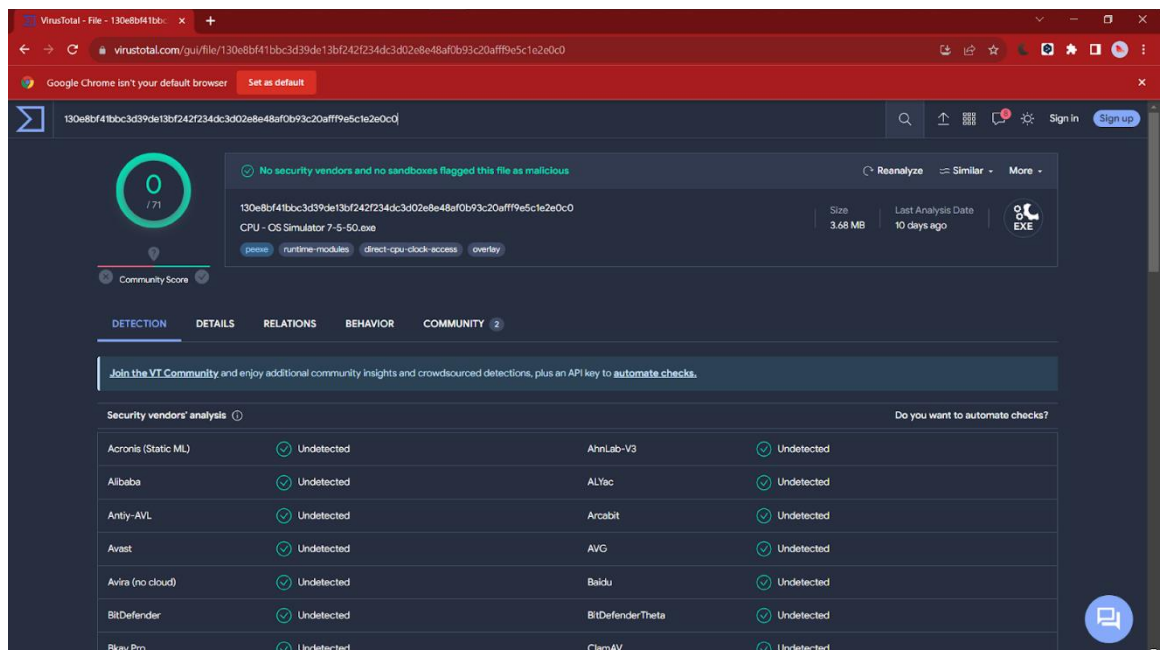
Website yang dapat menganalisis file, domain, IP, dan URL yang mencurigakan untuk mendeteksi malware dan pelanggaran lainnya, menggunakan sejumlah mesin antivirus berbeda dan menghasilkan laporan tentang apakah berkas tersebut mengandung malware atau tidak. Kemudian secara otomatis membagikannya dengan komunitas keamanan.

- Buka link <https://www.virustotal.com/>
- Unggah file atau masukkan link lalu submit.



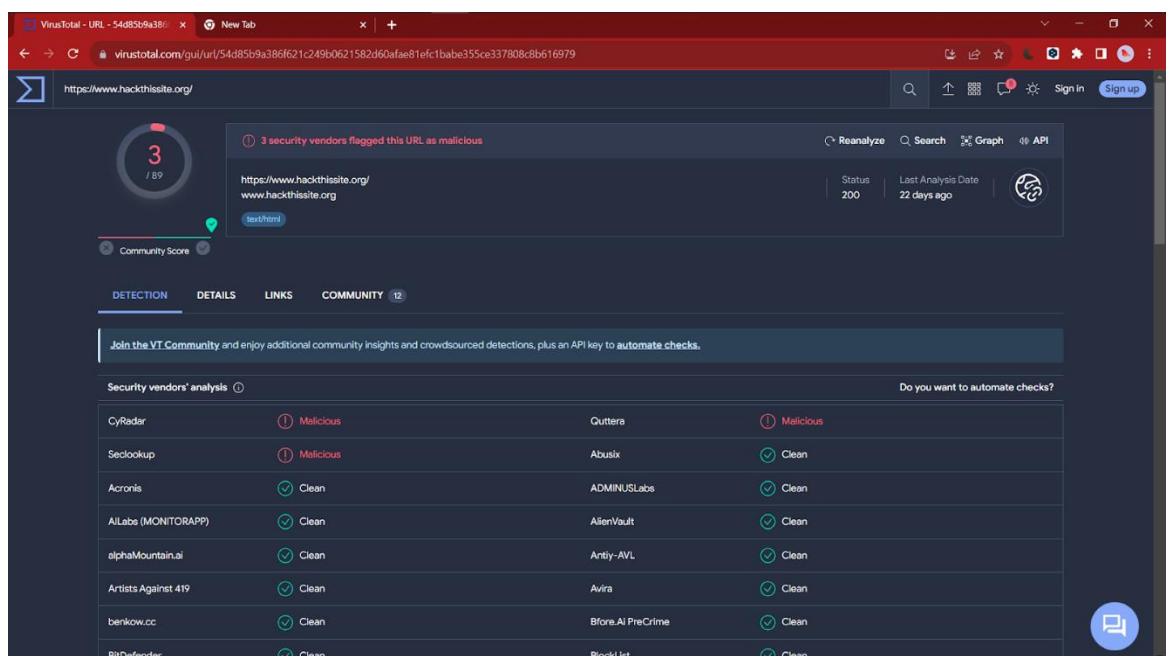
Gambar 10. 6 Tampilan Awal Website

- Tunggu hingga proses scan selesai, VirusTotal akan memberikan laporan yang mencakup informasi tentang seberapa aman file atau URL tersebut menurut sejumlah mesin antivirus yang berbeda.



Gambar 10. 7 Hasil Pindai File

- Jika ada beberapa mesin antivirus yang mendeteksi potensi ancaman atau malware, VirusTotal akan memberikan rincian tentang deteksi tersebut.



Gambar 10. 8 Contoh Link Website Yang Terdeteksi Berbahaya

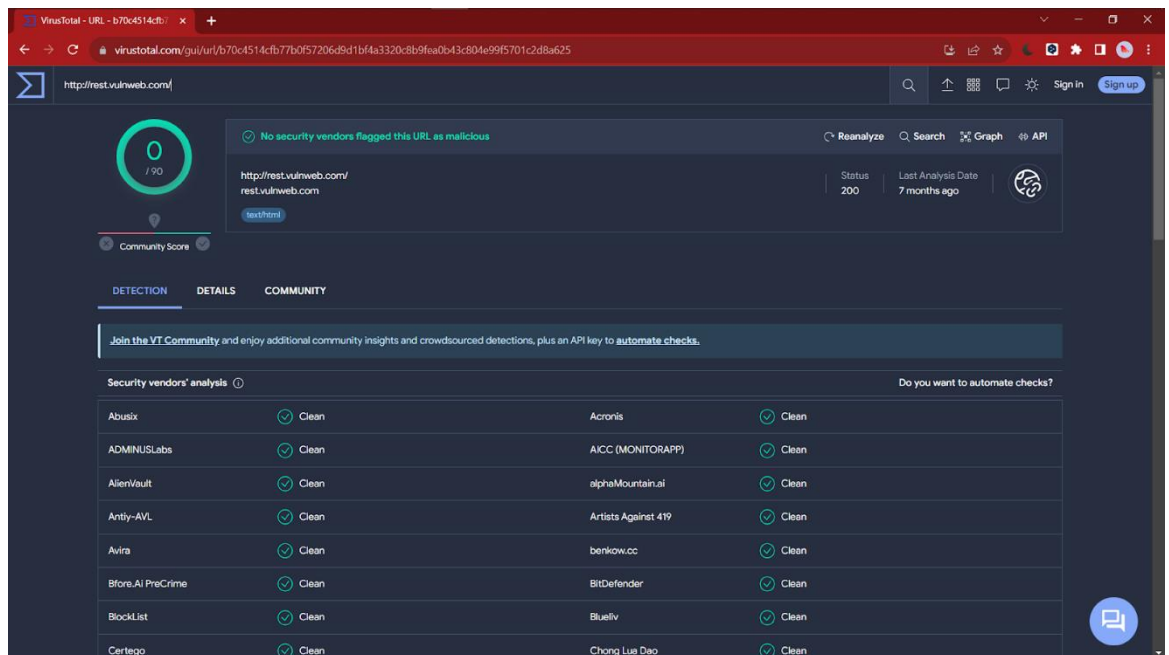
URL yang bisa dicoba:

www.proxysite.com

vertexfy.com

www.hackthissite.org

- Jika file atau URL dinyatakan aman oleh semua mesin antivirus, maka kemungkinan besar aman.

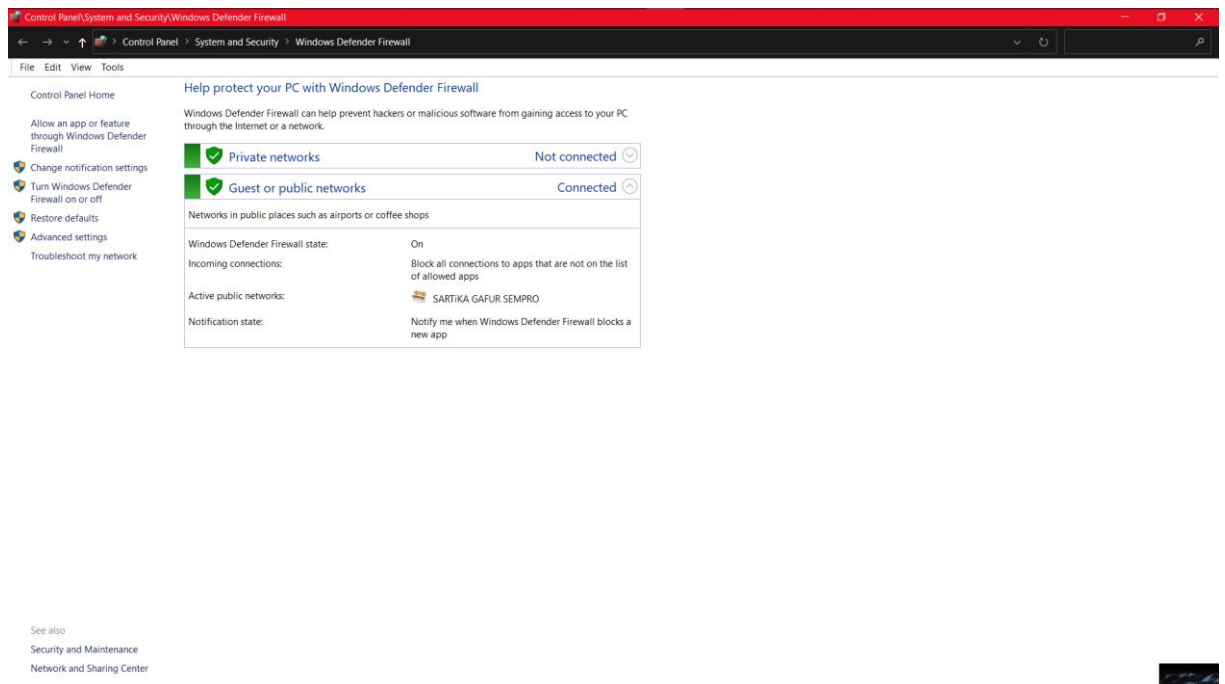


Gambar 10. 9 Contoh Link Website Yang Tidak Terdeteksi Berbahaya

- Berdasarkan hasil scanning, tindakan selanjutnya dapat diputuskan. Jika file atau URL dianggap aman, penggunaannya dapat dilanjutkan. Namun, jika terdapat indikasi ancaman, pertimbangkan untuk menghapus atau mengisolasi file maupun menghindari mengakses URL tersebut.

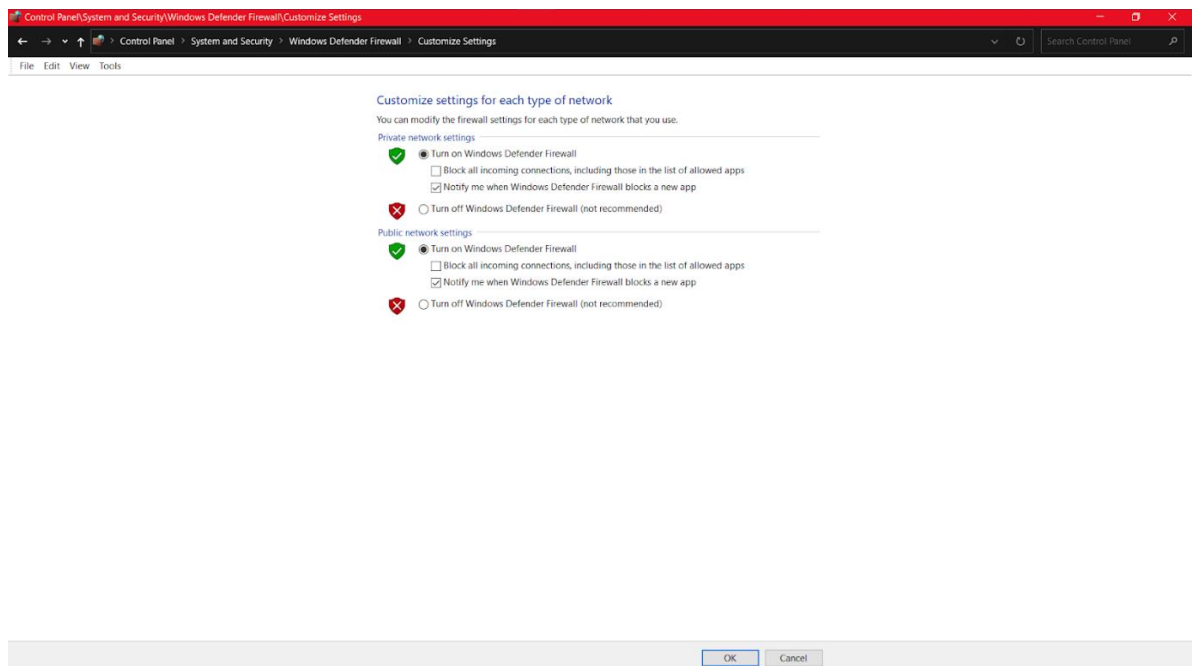
3. Windows Defender Firewall

Windows Defender Firewall membantu mencegah aksi hacking atau malware mendapatkan akses ke sistem melalui Internet atau jaringan. Jika sebuah virus berupaya untuk masuk, firewall akan mencegahnya.



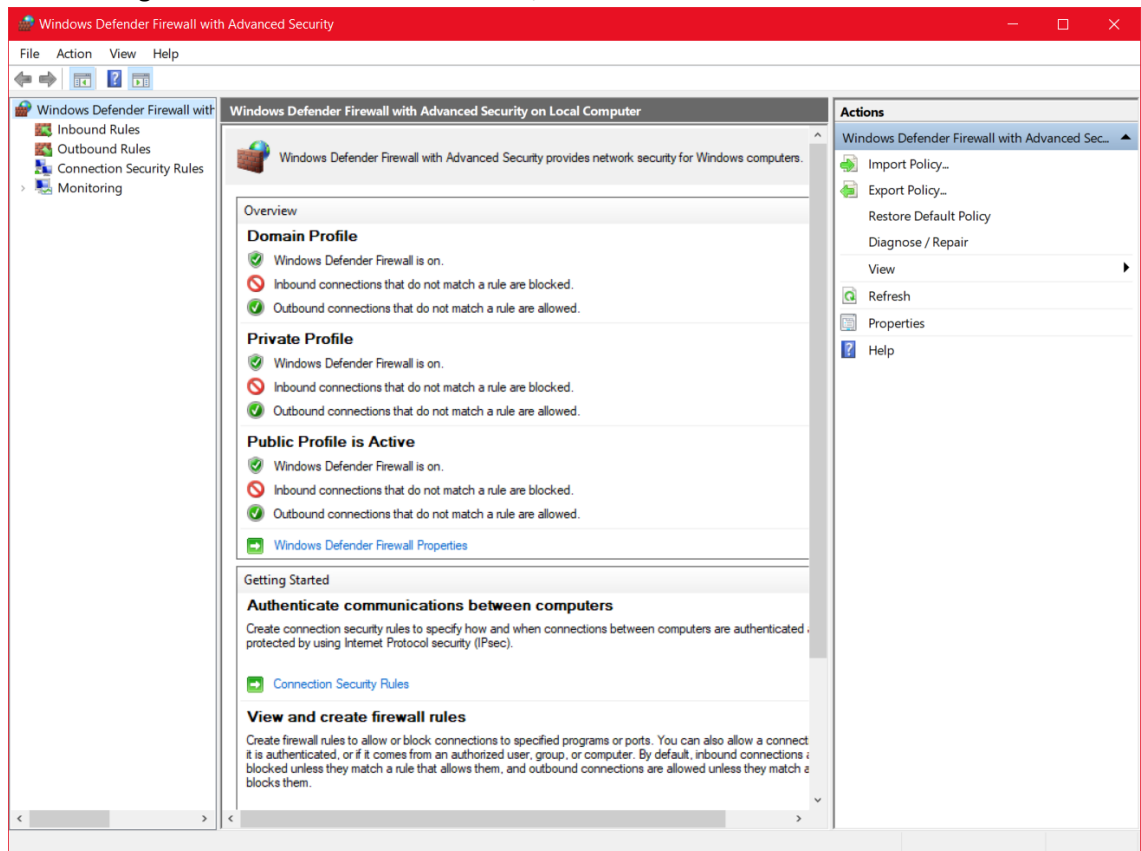
Gambar 10. 10 Tampilan Awal Windows Defender Firewall

- Ketika akan menyalakan Firewall, terdapat pengaturan berbeda berdasarkan jenis network-nya, baik private maupun public yang bisa dinyalakan dan dimatikan sesuai keperluan.



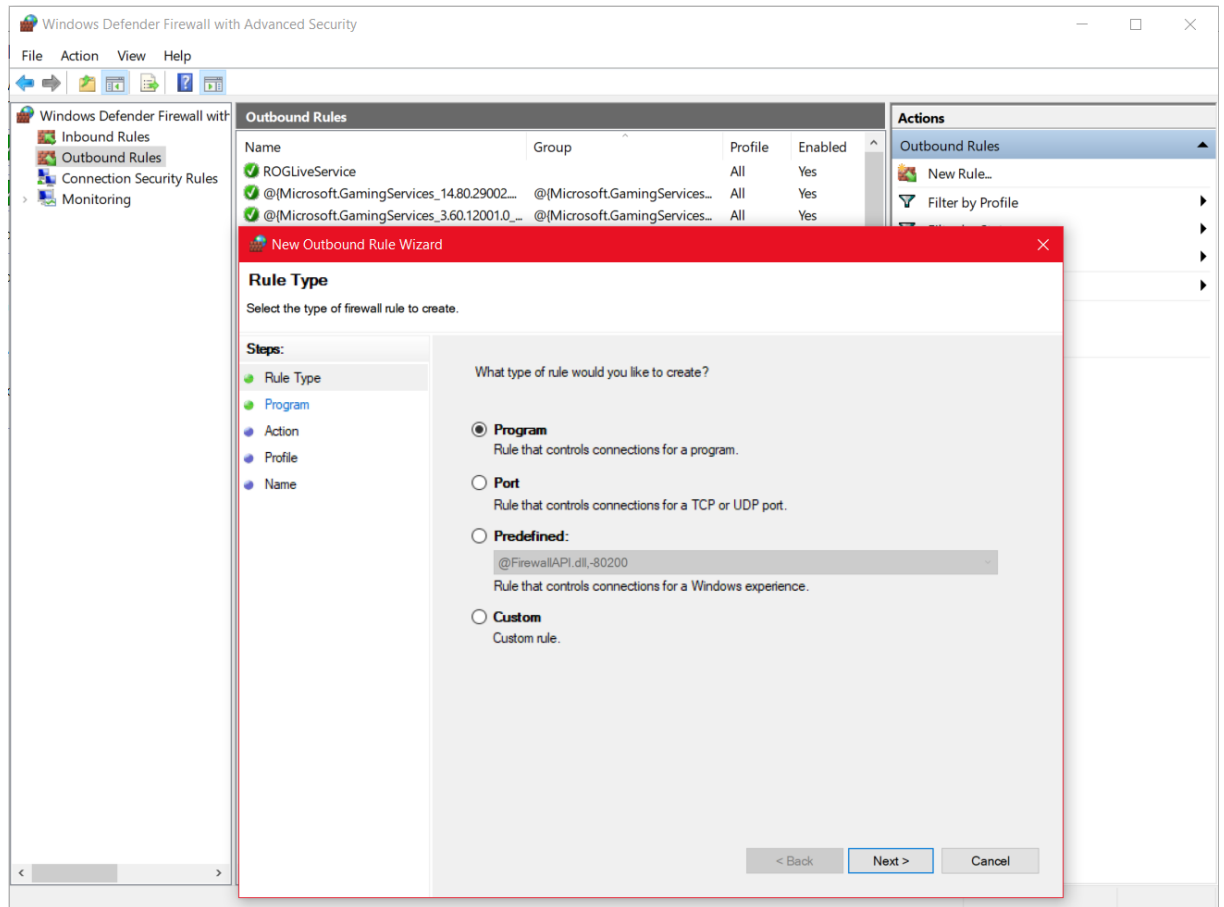
Gambar 10. 11 Custom Pengaturan Firewall Berdasarkan Tipe Network

- Di jendela Windows Defender Firewall with Advanced Security, terdapat daftar "Inbound Rules" (Aturan Masuk) dan "Outbound Rules" (Aturan Keluar) di panel sebelah kiri yang memungkinkan untuk mengelola aturan firewall secara rinci, termasuk aturan inbound dan outbound.



Gambar 10. 12 12 Windows Defender Firewall with Advanced Security

- Dengan mengonfigurasi aturan firewall, perizinan pemblokiran aplikasi, port, protokol, dan alamat IP tertentu untuk lalu lintas dapat dilakukan dengan aturan untuk memastikan bahwa aturan firewall sesuai dengan kebutuhan di berbagai lingkungan jaringan.



Gambar 10. 13 Aturan Outbound Mengontrol Lalu Lintas Keluar

10.7. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100**):

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-05	CPMK-02	Bagaimana Anda dapat mengidentifikasi dan menghapus perangkat lunak jahat (malware) dari sistem operasi Anda tanpa menggunakan perangkat lunak antivirus? Jawab pertanyaan ini dengan rinci.	15
2.	CPL-05	CPMK-02	Analisis satu link URL yang tidak terproteksi SSH dan satu file (selain .docx, .jpg, .mp4, .mkv, .mp3, .png, .pdf) menggunakan website VirusTotal Website. Screenshot dan tuliskan langkah-langkahnya.	55
3.	CPL-05	CPMK-02	Aktifkan Firewall dan blokir salah satu website berita, screenshot bukti bahwa website tersebut berhasil terblokir, tuliskan langkah-langkahnya menggunakan bahasa Anda sendiri.	30

10.8. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
Total Nilai						

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : NIM :	Asisten: Paraf Asisten:	Tanggal: Nilai:
-------------------------------	--	----------------------------------

--

DAFTAR PUSTAKA

1. Silberschatz, A., Galvin, P., Gagne, G., 2013, *Operating System Concept*, Ed.9, John Wiley and Sons (Asia), Inc.
2. Tanenbaum, A.S., 2015, *Modern Operating System*, 4rd Ed., Pearson Education International, Prentice Hall.



**LABORATORIUM
S1 INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS AHMAD DAHLAN**



2023