https://doi.org/10.37817/ikra-ithabdimas.v7i3
P-ISSN : 2654-5721
E-ISSN : 2654-7546

Workshop Pengenalan Aplikasi CPU OS Simulator untuk Penjadualan First-Come First-Served (FCFS)

Agung Mulyo Widodo[1], Roesfiansjah Rasjidin[2], Muhamad Bahrul Ulum[3], Binastya Anggara Sekti[4], Nixon Erzed[5], Ryan Putra Laksana[6], Budi Tjahjono[7], Hendry Gunawawan[8]

1,3,4,5,6,7,8 Fakultas Ilmu Komputer, Universitas Esa Unggul2 Fakultas Teknik, Universitas Esa Unggul

[1]agung.mulyo@esaunggul.ac.id, [2]roesfiansjah.rasjidin@esaunggul.ac.id,
[3]m.bahrul_ulum@esaunggul.ac.id, [4]anggara@esaunggul.ac.id, [5]nixon@esaunggul.ac.id,
[6]ryan.putra@esaunggul.ac.id, [7]budi.tjahjono@esaunggul.ac.id, [8]hendry.gunawan@esaunggul.ac.id

Abstract

CPU OS Simulator is a simulator application that supports a learning process that synergizes both hardware and software. In this application there is a First Come First Serve (FCFS) scheduling module, an operating system scheduling algorithm that automatically executes requests (request by priority) and processes queues in the order of arrival. Scheduling algorithm for processes (assembly and visualization) using CPU OS Simulator, this application is easier to operate and user-friendly for users. In this type of algorithm, the process that requests CPU first gets CPU allocation first. The scheduling process is managed using the First-In First-Out (FIFO) queuing method, within the scope of the operating system course with a focus on FCFS (First Come First Serve) algorithm scheduling. The workshop method was carried out online using a zoom cloud meeting, with lecturers who were interested in the application and students who wanted to understand the concepts of the operating system. The CPU OS simulator application is a free-ware application so it can be downloaded for free for users (lecturers and/or students). The final result of the training for users will be to see and understand step by step from the instruction to compilation process which is presented in the form of simulation and visualization (histogram) as well as several parameters of the process (average waiting time, average burst time, tick count and instruction count).

***Keywords**: CPU OS Simulator, Scheduling, FCFS, Pipeline Visualization*

Abstrak

CPU OS Simulator merupakan aplikasi simulator yang mendukung proses pembelajaran yang mensinergikan dari sisi perangkat keras maupun perangkat lunak. Dalam aplikasi ini terdapat modul penjadualan *First Come First Serve* (FCFS) merupakan algoritma penjadwalan sistem operasi yang secara otomatis mengeksekusi permintaan (*request by priority*) dan proses antrian sesuai urutan kedatangannya (*arrival*). Algoritma *scheduling* untuk proses (assembly dan visualisasi) dengan menggunakan CPU OS Simulator, aplikasi ini lebih mudah dioperasikan dan *user-friendly* bagi pengguna. Dalam algoritma jenis ini, proses yang me-*request* CPU terlebih dahulu mendapatkan alokasi CPU terlebih dahulu. Proses penjadualan dikelola dengan metode antrian *First-In First-Out* (FIFO), dalam ruang lingkup sistem operasi dengan fokus penjadualan algoritma FCFS (*First Come First Serve*). Metode workshop dilaksanakan secara daring menggunakan zoom cloud meeting, dengan peserta dosen yang berminat akan aplikasi tersebut dan para anak didik yang mau mengenal memahami konsep dari sistem operasi. Aplikasi CPU OS simulator merupakan salah satu aplikasi yang bersifat *free-ware* sehingga dapat di unduh secara gratis bagi para pengguna (dosen dan atau mahasiswa). Hasil akhir dari pelatihan bagi pengguna akan melihat dan mengerti tahapan demi tahapan dari proses intruksi hingga compile yang disajikan bentuk simulasi dan visualisasi (histogram) serta beberapa parameter dari proses tersebut (*average waiting time, average burst time, tick count* dan *instruction count*).

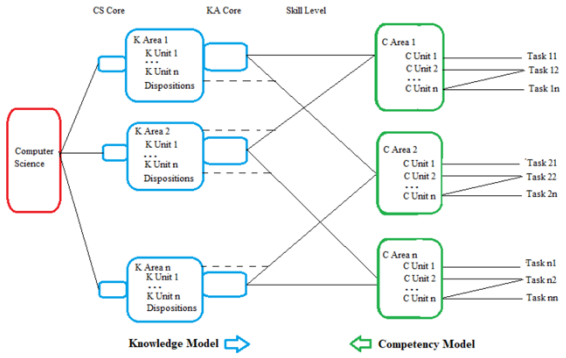
Kata kunci: CPU OS Simulator, Penjadualan, FCFS, Visualisasi Pipeline

1. Pendahuluan

Menjalankan proses pembelajaran pada tahapan akademik maupun vokasi sesuai dengan kompetensi bidang informatika yang diakui nasional dan internasional, sehingga diperlukan adanya kreatif, inovasi, tepat guna dan bermanfaat bagi pendidik dan anak didik. Dan seiring perkembangan platform aplikasi simulasi (sistem operasi), kami menggunakan CPU OS Simulator yang dapat digunakan untuk memudahkan dalam *transfer knowledge* dan *skills* pada tersebut yang tertuang pada kurikulum operasional berbasis OBE (*Outcome Base Education*) pada bidang ilmu program studi "computer science". Aplikasi CPU OS Simulator merupakan aplikasi yang dapat d unduh secara gratis dan hasil dapat menvisualisasikan tahapan proses, beroperasi pada platform - Windows, Linux dan MacOS. Selain aplikasi ini terseda juga antara lain; OS Sim (OS *Concepts Simulator*)[10-11]. Outcome Based Education (OBE) merupakan kurikulum yang mengacu pada luaran atau capaian pembelajaran sehingga tidak semata-mata materi saat di kelas (daring maupun luring) dan harus diaplikasikan namun diperlukan perencanaan mempersiapkan bagaimana anak didik sebagai "raw material" dalam menempuh proses pembelajaran hingga

P-ISSN : 2654-5721
E-ISSN : 2654-7546
https://doi.org/10.37817/ikra-ithabdimas.v7i3

tercapai menjadi lulusan yang telah mempunyai kemampuan untuk menghadapi dunia kerja di tengah berkembang teknologi informasi dan komunikasi. Pendidik harus memposisikan anak didik adalah individu yang sedang berkembang, memiliki potensi, jiwa kreatif untuk menuju bersikap dan berpikir inovatif. ACM merupakan singkatan **Association Computer for Machinery** didirikan pada 15 September 1946[1] oleh Richard Hamming. Lembaga ini pada Maret 2023 versi Beta merekomendasikan kurikulum Computer Science, yang merupakan hasil task force ACM, IEEE-CS dan AAAI. Gambar 1. menggambarkan ilustrasi hubungan *knowledge model vs competency model*, tahapan agar menghasilkan tingkat kompetensi/keahlian dalam bidang *computer science*.



Gambar 1. Model Knowledge versus Competency (Computer Science Curricula , March 2023, p. 12)[1]

Gambar 1, tingkat kompetensi atau keahlian merupakan titik temu mediasi antara model pengetahuan (bidang pengetahuan, unit pengetahuan dan topik) dan m kompetensi (bidang kompetensi, unit kompetensi dan tugas).

Pada workshop ini dalam rangka *transfer knowledge* dan *skills* dengan penerapan sistem operasi, menggunakan aplikasi OS Simulator[3][8][10]. Aplikasi ini dapat pula beroperasi secara umum di OS Windows, OS Linux dan MacOS. Luaran aplikasi dapat membuat menampilkan visualisasi *step-by-step* proses compile → run (start). Modul pelatihan yang diajarkan bagaimana pengenalan penggunaan OS Simulator (OSS) yang beroperasi OS Windows[2] bagi pemula, simulasi *scheduling* dengan algoritma FCFS. Dalam algoritma ini mempunyai karakteristik algoritma penjadualan CPU[6-7][9] Preempetive dan Non-Preempetive, proses operasional dilaksanakan FCFS sederhana dalam menggunakannya, dan kurang effisien dalam performance dibandingkan algoritma lainnya – *Round Robin (RR)* dan *Shortest Remaining Time (SRT)*, dan *Shortest Job First* (SJF)[9].

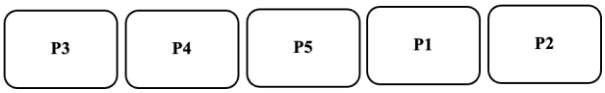
2. Metode Pelaksanaan

Dalam pelaksanaan workshop dilakukan dengan menggunakan zoom meeting **tema Workshop OS Simulators ID : 875 9127 8995 Key : 2023OSS**, dihadiri sebanyak 60 peserta (mahasiswa dan dosen). Manfaat yang diharapkan untuk peserta dapat mengenal dan mengoperasikan OS Simulator pada topik *scheduling* (*step-by-step*) dengan pendekatan algoritma FCFS, serta dapat menampilkan code generate program, visualisasi, dan bahasa assembler. Pada *scheduling* [4][9] terdapat 2 (dua) jenis prioritas; (1) penjadwalan preemptive, adalah penjadwalan pada sistem operasi dengan kemampuan menghentikan sementara proses yang sedang berlangsung proses, perhitungan dengan skala prioritas. Proses dengan prioritas yang dipandang lebih tinggi akan menjadi proses satu-satunya yang menggunakan CPU sampai selesai. Apabila ada prioritas baru yang dipandang lebih tinggi lagi maka proses sebelumnya akan diabaikan; (2) penjadwalan non-preemptive adalah kebalikan dari penjadwalan preemptive di mana proses yang sedang berlangsung tak bisa dihentikan sementara atau di interupsi sehingga CPU akan tetap mendahulukan proses yang sudah berjalan lalu beralih ke proses selanjutnya jika sudah selesai. Dalam penjadualan ini hanya mengizinkan beroperasi satu proses saja, proses tak bisa dihilangkan atau ditunda sementara hingga selesai, *context switch* aktif (*calls*) saat proses di-blok atau diberhentikan.

Tabel 1. FCFS Preemptive 5 Proses

Proses	AT	BT
P1	4	2
P2	5	5
P3	1	6
P4	2	4
P5	3	3

dalam gant chart pre-empetive dari table 1.,



Gambar 2. Gant Chart FCFS Preemptive 5 Proses

Untuk mengoperasikan aplikasi diperlukan persyaratan (minimum) perangkat yang digunakan, sebagai berikut;

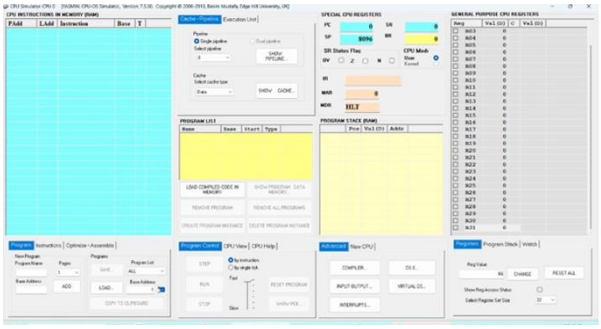
Tabel 2. Spesifikasi yg dibutuhkan [3]

Komponen	Deskripsi
CPU	Prosessor Intel atau AMD 64-bit
RAM	4 GByte
Card Graphics	GPU (Graphics Processing Unit) mendukung OpenGL 3.1 atau lebih (Nvidia GeForce, AMD Radeon, Intel HD Graphics)
Sistem Operasi	Windows 10, Linux dan MacOS
Hard disk	3 GByte free

https://doi.org/10.37817/ikra-ithabdimas.v7i3
P-ISSN : 2654-5721
E-ISSN : 2654-7546

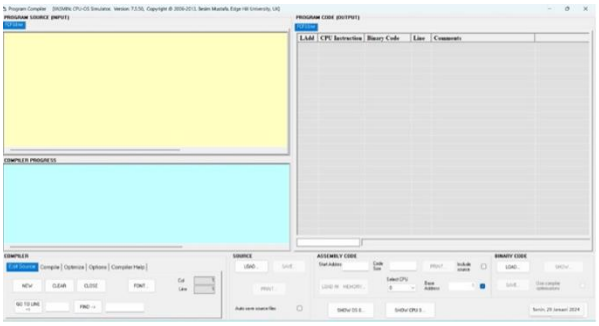
3. Hasil Pembahasan

Langkah awal install aplikasi CPU OS Simulator yang telah di unduh file compress zip, kemudian di-ekstrak. Saat sudah selesai file ektension file *.pkg rubah menjadi *.exe[2][3][8], kemudian klik file exe sebanyak 2 kali untuk proses installasi.



Gambar 3. Tampilan Front-End CPU OS Simulators

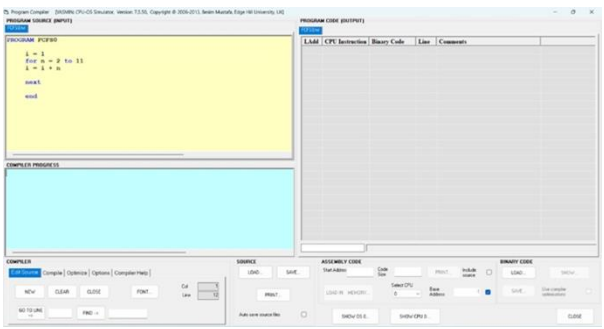
langkah 1; setelah buka aplikasi OSS → klik Advanced → COMPILER



Gambar 4. Tampilan Menu Program Compiler

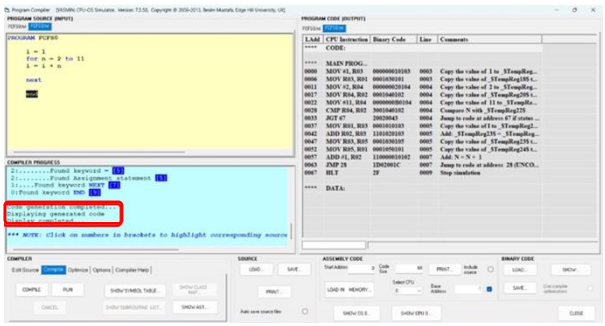
Lalu ketik-kan (dapat menggunakan notepad) script pada program source dengan nama program FCFS0,

```
PROGRAM FCFS0
        i = 1
        for n = 2 to 11
        i = i + n
        next
end
```



Gambar 5. Tampilan Program Source FCFS0

langkah 2; Pilih Compiler (gambar 4 atau 5), klik Compile → COMPILE maka yang akan ditampilkan (gambar 6)

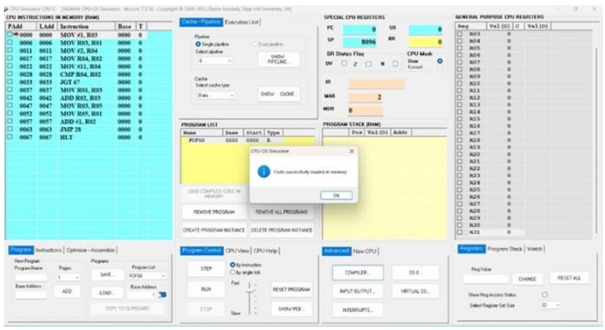


Gambar 6. Tampilan hasil COMPILE dari FCFS0

Pada gambar 6, pada progress compile sukses (persegi warna merah) tanpa ada error (completed);

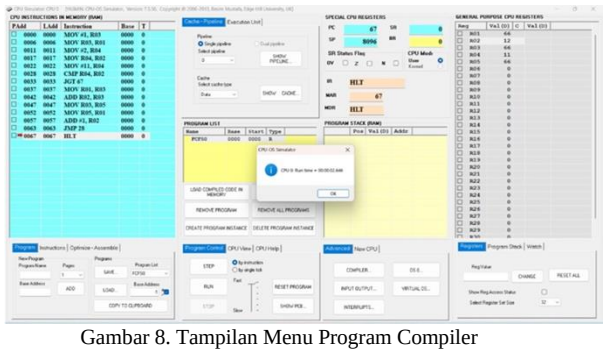
Code generation **completed**...
Displaying generated code
Display **completed**

Kemudian lanjutkan **langkah 3**; klik Assembly Code → Load Memory (CPU Simulator CPU 0, gambar 7)



Gambar 7. Tampilan Menu Program Compiler

langkah 4 (gambar 7) maka aplikasi akan menampilkan code yang diprogramkan sudah tersimpan pada memori kemudian klik RUN (Fast) pada menu Program Control sehingga akan menampilkan (gambar 8),



Gambar 8. Tampilan Menu Program Compiler

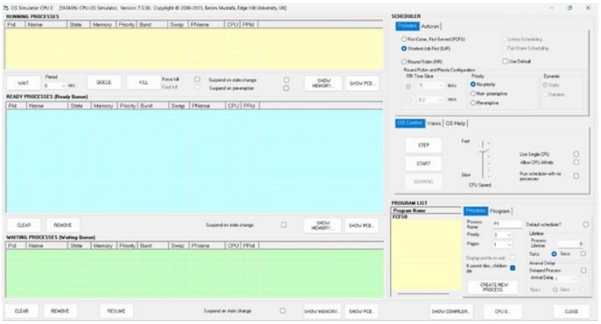
Dari gambar 8 diatas menghasilkan sebagai berikut;

CPU 0: Run time = 00:0002.646 detik
Nilai register: R00 = 0, R01 = 66, R02 = 12; R03 = 66, R04 = 11 dan R05 = 66.

langkah 5; Untuk proses penjadualan FCFS, asumsikan untuk 5 proses dengan prioritas tertentu

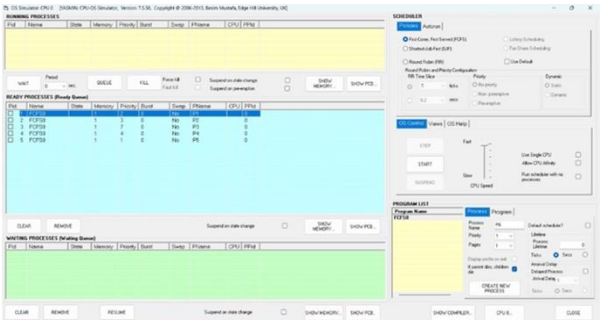
P-ISSN : 2654-5721
E-ISSN : 2654-7546
https://doi.org/10.37817/ikra-ithabdimas.v7i3

(sample), klik Advanced → OS.O akan menampilkan



Gambar 9. Tampilan Menu OS Simulator CPU 0

Dari 5 proses tersebut (program list → process), kita tentukan misalkan P1 → priority 2; P2 → priority 3; P3 → priority 7; P4 → priority 4; P5 → priority 1. Setelah ini maka tersaji sub-menu Ready Process (**langkah 6**),

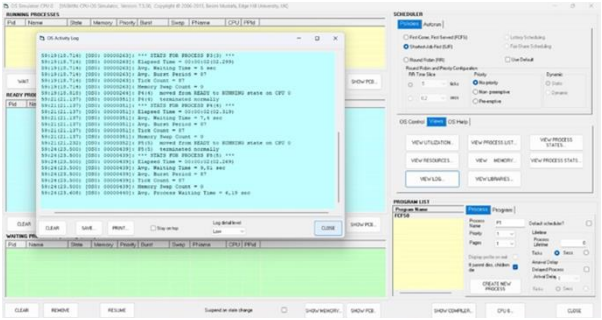


Gambar 10. Tampilan Ready Process (P1 – P5)

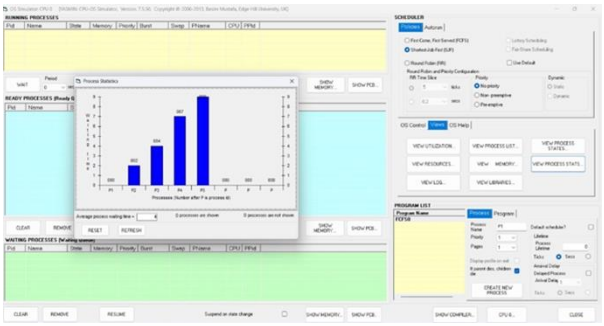
Sebelum meng-klik Start (**langkah 7**), lakukan klik Views sebelah kiri dari OS Control → View Log

```
00:52:15(15.073) [OS0: 00000000]: P1(1)  loaded into
the READY queue [Memory=1 pages, Priority=2]
00:52:22(21.946) [OS0: 00000000]: P2(2)  loaded into
the READY queue [Memory=1 pages, Priority=3]
00:52:29(29.360) [OS0: 00000000]: P3(3)  loaded into
the READY queue [Memory=1 pages, Priority=7]
00:52:34(33.687) [OS0: 00000000]: P4(4)  loaded into
the READY queue [Memory=1 pages, Priority=4]
00:52:39(38.796) [OS0: 00000000]: P5(5)  loaded into
the READY queue [Memory=1 pages, Priority=1]
```

Langkah 8, kembali ke OS Control → Start (Fast) maka akan menghasilkan (a) view log ter-update (gambar 11), (b) statistik dari 5 proses (gambar 12),

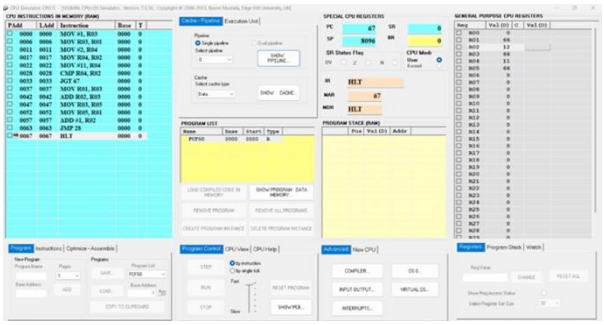


Gambar 11. Tampilan View Log (P1 – P5) pasca klik Start



Gambar 12. Tampilan View → Process Statistic

Dari gambar 11, menghasilkan rata-rata proses waiting time sebesar 6,19 detik. Dan gambar 12, menampilkan dalam bentuk diagram histogram P1 → P5 menghasilkan rata-rata proses waiting time = 4 detik. **Langkah 9**, kemudian klik close maka aplikasi CPU Simulator CPU OS. 0 (gambar 13),



Gambar 13. Tampilan klik close dari dari command Compiler dan OS.0

Langkah ke-10, klik Cache-Pipeline → Show Pipeline (gambar 14).

Gambar 14. Tampilan Instruksi Pipeline CPU 0

Dari gambar 14, menghasilkan beberapa parameter atau variable statistic sebagai berikut

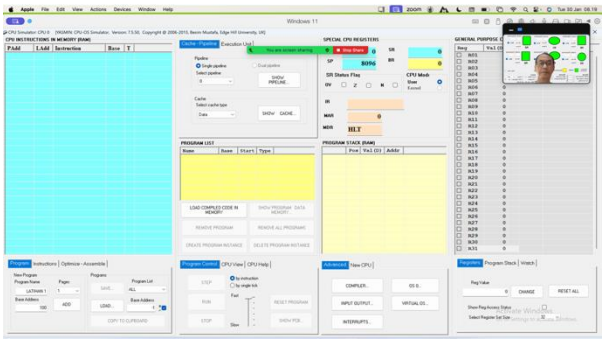
Clocks = 2208; IC = 440, CPI = 5.02 dan SF = 1(*)

(*) IC = Instruction Count, CPI = Cycle per Instruction; SF = Speed-up Factor

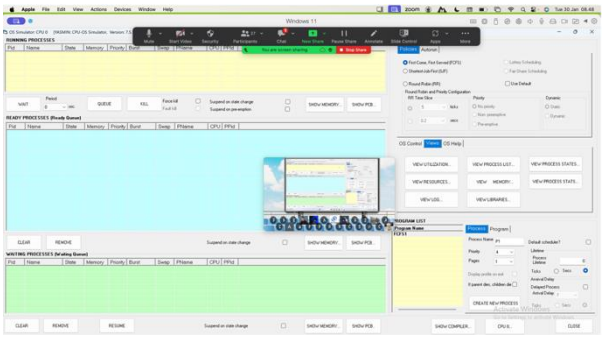
3.1 Foto dokumentasi

Workshop CPU OS Simulator untuk kajian penjadualan (*scheduling*) dilaukan secara daring dengan zoom cloud meeting.

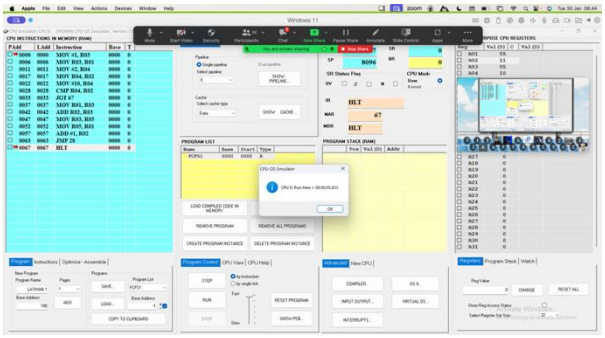
https://doi.org/10.37817/ikra-ithabdimas.v7i3
P-ISSN : 2654-5721
E-ISSN : 2654-7546



Gambar 15. Dokumentasi Workshop OS Simulator



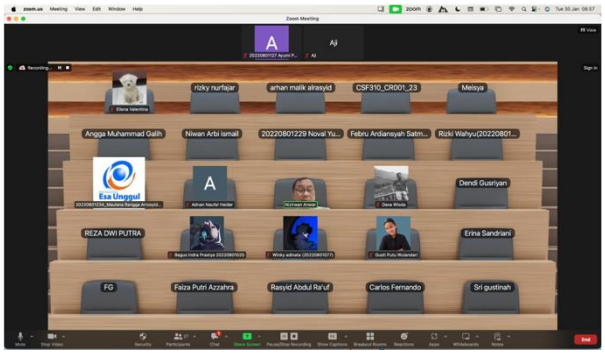
Gambar 16. Dokumentasi Workshop OS Simulator



Gambar 17. Dokumentasi Workshop OS Simulator

Gambar 18. Dokumentasi Workshop OS Simulator

Gambar 19. Dokumentasi Workshop OS Simulator



Gambar 20. Dokumentasi Workshop OS Simulator

Gambar 21. Dokumentasi Workshop OS Simulator

4. Kesimpulan

Aplikasi CPU OS Simulator dapat pula diterapkan untuk algoritma penjadualan antara lain; *Shortest Job-First* (SJF), *Round Robin* (RR), *Lottery Scheduling* (LS) dan *Fair Share Scheduling (*FSS*)* serta modul-modul yang lain yang tak terpisahkan Pada workshop ini dengan menggunakan algoritma *First-Come First-Serve* (FCFS) dalam mengoperasikan dan mengimplementasikan lebih sederhana sebanyak proses yang diasumsikan, dalam pelatihan hanya untuk 5 proses dengan prioritas tertentu (*no priority, non-preeemptive, pre-empetive*). Dampak dari pelatihan para peserta memperoleh pengetahuan, skills, dan pengalaman baru bagaimana sistem operasi dapat dilakukan dengan simulasi (gambar 6, 7, 8 dan 11) dan visualisasi luarannya (gambar 12 dan 14). Dan mengusulkan untuk ke depan melakukan workshop kembali dengan menerapkan 3 (tiga) algoritma yang

belum disajikan dengan prioritas non-preemptive dan pre-empetive.

Daftar Rujukan

[1] Computing Curricula. 2001. Computing Science, Final Report, December 15 2001. ACM and IEEE Computer Society joint report, USA. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2001.pdf

[2] Khusna, Arfiani Nur. dkk, 2020, Petunjuk Praktikum Sistem Operasi, Laboratorium Teknik Informastika, Universitas Ahmad Dahlan, Yogjakarta.

[3] CPU Simulator. 2021, CPU-OS Simulator, Jun. 07, 2021. https://teach-sim.com

[4] Murdocca, Milles J., et.all., 2007, Computer Architecture and Organization an Integrated Approach, Publisher Wiley 1st edition. http://iiusatech.com/murdocca/CAO/, ISBN 978-0471733881.

[5] Mustafa, B. (2009) Evaluating a System Simulator for Computer Architecture Teaching and Learning Support. The Higher Education Academy Subject Centre for Information and Computer Science's 10th Annual Conference, August 25-27, 2009, Canterbury, UK (Also in HEA ITALICS, Volume 9, Issue 1, 100-104, February 2010). DOI: 10.11120/ital.2010.09010100

[6] Hennessy, J. L., Patterson, D. A. 2003. Computer Architecture: A Quantitative Approach. third edition. Morgan Kaufmann publishers.

[7] Patterson, D. A., Hennessy, J. L. 2005. Computer Organization and Design. Third edition, Morgan Kaufmann publishers.

[8] M. Besim and P. Alston, "Understanding Computer Architecture with Visual Simulations: What Educational Value?," in Springer eBooks, 2012, pp. 1–9. doi: 10.1007/978-3-642-28801-2_1.

[9] William Stallings, 2010, Operating Systems, Internals and Design Principles, Pearson: Prentice Hall, ISBN 9780132309981.

[10] SourceForge, OS SIM (Simultor Konsep OS), Perangkat Lunak Sumber Terbuka., 2023, https://sourceforge-net.translate.goog/projects/oscsimulator/files.

[11] "Oscsimulator.sf.net." https://os-sim-os-concepts-simulator.soft112.com/