OS Project Description

Core Components and Data Structures

1. **Memory System** (Lines 50-56)

•Memory is represented as an array of MemoryWord structures

•Each word contains name, value, process ID, and type (VAR, CODE, T_PCB, etc.)

•Fixed size of 60 words

2. **Process Control Block (PCB)** (Lines 58-68)

•Contains process metadata:

- •Process ID
- •Current state
- •Priority level
- •Program counter
- •Memory bounds
- •Timing information
- •Code size

3. **Process States** (Lines 35-41)

•States include: NEW, READY, RUNNING, BLOCKED, TERMINATED

•Used to track process lifecycle

Process Management

1. **Process Creation** (Lines 592-665)

•Allocates memory for process

•Creates PCB

•Loads program code

•Initializes variable space

•Adds process to ready queue

2. **Process Scheduling** (Lines 511-557) Supports three scheduling algorithms:

•Round Robin (RR)

•First In First Out (FIFO)

•Multi-Level Feedback Queue (MLFQ)

3. **Context Switching** (Lines 474-496)

•Saves current process state

•Loads new process state

•Updates process states in memory

•Manages ready queues

Resource Management

1. **Mutex Operations** (Lines 234-281)

•Semaphore wait/signal implementation

•Handles resource locking/unlocking

•Manages blocked processes

2. **Memory Management** (Lines 666-674)

•Initializes memory spaces

•Tracks allocated memory

•Manages memory boundaries

Process Execution

1. **Command Processing** (Lines 366-419) Supports commands:

•print

•assign

•writeFile

•readFile

•printFromTo

•semWait/semSignal

2. **Variable Management** (Lines 283-324)

•Find variables in memory

•Get/Set variable values

•Variable space allocation

System Control

1. **Clock Cycle Management** (Lines 558-591)

•Checks for new processes

•Handles scheduling

•Executes current process

•Updates system state

2. **Main Control Loop** (Lines 738-769)

•System initialization

•Process creation

•Execution cycle

•Termination handling

Queue Management (Lines 675-687)

•Initializes ready queues

•Manages priority queues

•Handles blocked process queues

This operating system simulation implements basic process management, scheduling, and resource allocation features typical in modern operating systems. The design follows a modular approach with clear separation between different system components.

the GUI implementation in a structured way focusing on the main components:

1. Main Components (Lines 18-50)

- Global GTK widget declarations for:
  - Main window and grid
  - Sidebar controls
  - Text views and buffers
  - List views and stores
  - Various control buttons and labels

2. Theme Management (Lines 937-1058)

- Implements dark/light mode switching
- GitHub-inspired color schemes
- Custom CSS styling for:
  - Buttons, labels, and frames
  - Treeviews and scrollbars
  - Console text view
  - Custom font configurations

3. GUI Layout (Lines 477-936)

Created in a hierarchical structure:

1. Main Window
2. Sidebar Panel containing:
   - System info display
   - Scheduler controls
   - Simulation controls
   - Process icons
3. Main Content Area with tabs:
   - Memory view
   - Process list
   - Queue lists
   - Resource monitor
   - Console output

4. Interactive Elements (Lines 364-476)

- Step execution button
- Auto-execution toggle
- Reset simulation button
- Process addition dialog
- Scheduler algorithm selection
- Quantum time adjustment

5. Process Visualization (Lines 1059-1116)

- Custom drawing for process icons
- State-based coloring:
  - Running: Green
  - Ready: Blue
  - Blocked: Red
  - Terminated: Gray
  - New: Yellow

6. Real-time Updates (Lines 1117-1150)

- Memory view updates
- Process list refreshes
- Queue status updates
- Resource state monitoring
- Console output handling

7. Event Handling (Lines 282-363)

- Button click responses
- Scheduler changes
- Process addition
- Theme toggling
- Auto-execution timing

8. Styling and Aesthetics

- GitHub-inspired design
- Custom fonts and colors
- Responsive layouts
- Professional looking widgets
- Modern UI elements

This GUI provides a comprehensive visual interface for monitoring and controlling the operating system simulation, with real-time updates and user-friendly controls.