Operating system tasks

Grading and Policies:

1. Students who use the Internet to plagiarize the task, or copy from his/her colleagues, will take zero.

2. The first task grade is 10.

3. 2. The second task grade is 5.

Required points to be in the project:

1. The project should execute (run) in the terminal

2. The project must provide a simple menu interface using c language to let the user picks the option (1-10) he wants.

3. The project should not be closed after executing only one option, instead the menu must re-appear until the user picks exit.

4.Implement the scheduling algorithm using any programming languages. you will find task 2 at the end of this file.

Task 1 Guidelines

Project Ideas:

#	**Table of contents**
1	University System
2	Hospital System
3	Banking System
4	Library Management System
5	Hotel Management System
6	E-Commerce System
7	Human Resources Management System (HRMS)
8	Government Services System
9	Transportation System
10	Social Media Platform

11	Retail Store Management System
12	Logistics and Supply Chain System
13	Online Examination System
14	Gaming Platform
15	Manufacturing System

1.University System

Scenario

Design and implement a university system where there are **three types of users**:

1. **Dean**: The root user with administrative privileges.
2. **Professors**: Users with limited access to perform specific tasks.
3. **Students**: Users with minimal privileges, primarily for viewing information.

This system must enforce the permissions and restrictions of each role using Linux commands. Implement the following tasks to manage the system.

File Manager Structure

The university system has a shared directory structure:

- `/university`
  - `/dean`: Contains administrative files only accessible by the dean.
  - `/professors`: Contains teaching materials. Professors can read/write, while students can only read.
  - `/students`: Contains student-specific files. Students can access their personal files but cannot modify others.

Tasks:

1. **List files/directories:**
   - Dean, professors, and students can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Dean can update permissions for all files/directories.
3. **Make/delete files/directories:**
   - Dean and professors can create or delete directories under `/professors`.
   - Students can create files in their own folders.
4. **Create symbolic link files:**
   - Dean creates symbolic links to commonly accessed files for all users.

5. **Copy files/directories:**
    - Professors or the dean can copy shared files to other directories for collaboration or backup.
    - Students can copy files within their personal directory but not across restricted folders.

6. **Move files/directories:**
    - Professors can reorganize teaching materials, and students can move their files within their personal directory.

7. **Use redirection to create or update files:**
    - Students can create or update their assignments using redirection (> and >>).
    - Professors can update logs or notes using the same commands.

8. **Set and use aliases for common tasks:**
    - The dean, professors, or students can create aliases to simplify frequently used commands.

9. **View file content using `cat`, `head`, and `tail`:**
    - Professors or students can view the content of a file for quick reference.

10. **Find files or directories using `find` or `grep`:**
    - Dean or professors can search for specific files or directories.
    - Students can find their files by content or name.

2.Hospital System

Scenario

Design and implement a hospital system where there are three types of users:

1. **Administrator:** The root user with administrative privileges.
2. **Doctors:** Users with limited access to patient records and medical reports.
3. **Patients:** Users with minimal privileges, primarily for viewing their own records.

File Manager Structure

The hospital system has a shared directory structure:

- `/hospital`
    - `/admin`: Administrative files only accessible by the administrator.
    - `/doctors`: Patient records and medical reports accessible to doctors.
    - `/patients`: Patient-specific records accessible only to the respective patient.

Tasks:

1. **List files/directories:**
   - Admin, doctors, and patients can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin can update permissions for all files and directories.
3. **Make/delete files/directories:**
   - Admin and doctors can create or delete directories under `/doctors`.
   - Patients can create files in their personal folders.
4. **Create symbolic link files:**
   - Admin creates symbolic links to commonly accessed files, such as hospital policies.
5. **Copy files/directories:**
   - Doctors or the admin can copy patient files for review or backup.
   - Patients can copy their records within their personal directories.
6. **Move files/directories:**
   - Doctors can reorganize patient records.
   - Patients can move their files within their personal directories.
7. **Use redirection to create or update files:**
   - Patients can add notes or updates to their records using > or >>.
   - Doctors can update patient treatment logs.
8. **Set and use aliases for common tasks:**
   - Admin, doctors, or patients can create aliases for commonly used commands.
9. **View file content using `cat`, `head`, and `tail`:**
   - Doctors or patients can quickly review record content.
10. **Find files or directories using `find` or `grep`:**

- Admin or doctors can search for specific patient records or reports.
- Patients can search their files by content or name.

3.Banking System

Scenario

Design and implement a banking system where there are three types of users:

1. **Admin:** Manages account data, transactions, and platform settings.
2. **Employees:** Access customer data and process transactions.
3. **Customers:** View their account details and transaction history.

File Manager Structure

- `/bank`
  - `/admin`: Contains administrative files and account settings, accessible only by the admin.
  - `/employees`: Contains transaction logs and customer account details, accessible only by employees.
  - `/customers`: Contains customer-specific transaction records and account details.

Tasks:

1. **List files/directories:**
   - Admin, employees, and customers can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin can update permissions for account files and transaction logs.
   - Employees can access but cannot modify admin files.
3. **Make/delete files/directories:**
   - Admin and employees can create or delete directories for customer accounts.
   - Customers can create files for transaction history.
4. **Create symbolic link files:**
   - Admin creates symbolic links for shared policies or transaction records.
5. **Copy files/directories:**
   - Admin or employees can copy account files for backup or sharing.
   - Customers can copy their transaction records for personal use.
6. **Move files/directories:**
   - Employees can reorganize account records and transaction logs.
7. **Use redirection to create or update files:**
   - Customers add feedback to their transaction history using > or >>.
   - Admin can update platform logs.
8. **Set and use aliases for common tasks:**
   - Admin, employees, and customers create aliases for frequently used commands.
9. **View file content using cat, head, and tail:**
   - Employees can view transaction logs, and customers can preview account summaries.
10. **Find files or directories using find or grep:**
    - Admin or employees search for specific transaction records.

Bonus Tasks:

11. **Compress files/directories for archiving:**
    - Admin compresses old transaction logs or account data for backup.
12. **Filter Files by Permissions:**
    - Admin can list files in the `/employees` directory writable only by employees.

4.Library Management System

Scenario

Design and implement a library management system where there are three types of users:

1. **Librarian:** Manages book records, member data, and library policies.
2. **Members:** Borrow books and view their borrowing history.
3. **Visitors:** Browse available books and search for titles.

File Manager Structure

- `/library`
  - `/librarian`: Contains book inventory, policies, and member data, accessible only by the librarian.
  - `/members`: Contains borrowing history and member-specific data.
  - `/visitors`: Contains book catalog information accessible to all visitors.

Tasks:

1. **List files/directories:**
   - Librarian, members, and visitors can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Librarian can update permissions for sensitive member records.
3. **Make/delete files/directories:**
   - Librarian creates or deletes book records and directories.
   - Members can create or update their borrowing history.
4. **Create symbolic link files:**
   - Librarian creates symbolic links for popular books or shared policies.
5. **Copy files/directories:**
   - Librarian copies book records for archiving.
   - Members copy their borrowing history or requests.
6. **Move files/directories:**
   - Librarian moves old book records to an archive.
7. **Use redirection to create or update files:**
   - Members update their borrowing logs or requests.
8. **Set and use aliases for common tasks:**
   - Librarian and members can create aliases for tasks like listing borrowed books.
9. **View file content using cat, head, and tail:**
   - Members view their borrowing history, and librarians check book catalogs.
10. **Find files or directories using find or grep:**
    - Librarian searches for books or member records.

5.Hotel Management System

Scenario
Design and implement a hotel management system where there are three types of users:

1. **Admin:** Manages platform operations, room inventories, bookings, and customer data.
2. **Staff:** Manages bookings, check-ins, and room assignments.
3. **Guests:** View and manage their bookings and personal details.

File Manager Structure
The hotel system has a shared directory structure:

- `/hotel`
  - `/admin`: Contains administrative files, accessible only by the admin.
  - `/staff`: Contains booking data, customer records, and room assignments.
  - `/guests`: Contains guest-specific booking details and preferences, accessible only by the respective guest.

Tasks:

1. **List files/directories:**
   - Admin, staff, and guests can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin can update permissions for all files and directories.
   - Staff can modify booking files, but they cannot access guest-specific directories.
3. **Make/delete files/directories:**
   - Admin and staff can create or delete directories for new bookings or room assignments.
   - Guests can add preferences or notes within their personal folder.
4. **Create symbolic link files:**
   - Admin creates symbolic links to commonly accessed files, such as room availability or hotel policies.
5. **Copy files/directories:**
   - Admin or staff can copy booking data for backup or sharing.
   - Guests can copy their booking details within their folder.
6. **Move files/directories:**
   - Staff can reorganize booking records or move room assignments.
   - Guests can move files within their personal directory for better organization.
7. **Use redirection to create or update files:**
   - Guests can add feedback to their booking or preferences using `>` or `>>`.
   - Staff can update the booking status or guest notes using redirection.
8. **Set and use aliases for common tasks:**
   - Admin, staff, or guests can create aliases for frequently used commands.
9. **View file content using cat, head, and tail:**
   - Staff can view booking records, and guests can preview their booking details or history.
10. **Find files or directories using find or grep:**
    - Admin or staff can search for specific booking records or rooms.

- Guests can search their own booking details or preferences.

6.E-Commerce System

Scenario
Design and implement an e-commerce system where there are three types of users:

1. **Admin:** Manages platform operations, inventory, and customer data.
2. **Sellers:** Upload and manage their products.
3. **Customers:** View and purchase products.

File Manager Structure
The e-commerce system has a shared directory structure:

- `/ecommerce`
    - `/admin`: Contains administrative files, only accessible by the admin.
    - `/sellers`: Contains seller product files. Sellers can read/write, while the admin can also manage these files.
    - `/customers`: Contains customer-specific order details. Customers can only view their respective files.

Tasks:

1. **List files/directories:**
    - Admin, sellers, and customers can list files in their respective directories.

2. **Change permissions of files/directories:**
    - Admin can update permissions for all files and directories.
    - Sellers can modify their product files but cannot access others' directories.

3. **Make/delete files/directories:**
    - Admin and sellers can create or delete product directories.
    - Customers can add notes or preferences to their respective folders.

4. **Create symbolic link files:**
    - Admin creates symbolic links to frequently accessed files, such as platform rules or seller guidelines.

5. **Copy files/directories:**
    - Admin or sellers can copy product files for backup or sharing.
    - Customers can copy their order files within their folders.

6. **Move files/directories:**
    - Sellers can reorganize product files for better management.
    - Customers can move files within their personal directories.

7. **Use redirection to create or update files:**
    - Customers can add feedback to their order history using `>` or `>>`.
    - Admin can update logs for platform policies.

8. **Set and use aliases for common tasks:**
    - Admin, sellers, or customers can create aliases to simplify frequently used commands.

9. **View file content using `cat`, `head`, and `tail`:**
    - Sellers can preview product details, and customers can check order summaries.

10. **Find files or directories using `find` or `grep`:**
    - Admin or sellers can search for specific products by name or content.
    - Customers can search for their orders by content or ID.

7.Human Resources Management System (HRMS)

Scenario

Design and implement an HRMS where there are three types of users:

1. **HR Manager:** Manages employee records, recruitment, and payroll.
2. **Team Leads:** View and manage team performance reports.
3. **Employees:** View personal details like salary slips and leave status.

File Manager Structure

- `/hrms`
    - `/hr_manager`: Contains administrative files accessible only by the HR Manager.
    - `/team_leads`: Performance reports accessible to team leads.
    - `/employees`: Personal files accessible only by respective employees.

Tasks:

1. **List files/directories:**
   - HR manager, team leads, and employees can list files in their respective directories.
2. **Change permissions of files/directories:**
   - HR manager updates permissions for payroll files or sensitive documents.
3. **Make/delete files/directories:**
   - HR manager creates or removes directories for new policies or recruitment drives.
   - Team leads can manage performance reports in their folders.
4. **Create symbolic link files:**
   - HR manager creates symbolic links for company policies or performance review guidelines.
5. **Copy files/directories:**
   - HR manager copies payroll data for backup.
   - Employees can copy their leave status files within their directories.
6. **Move files/directories:**
   - HR manager reorganizes policy files.
   - Employees can move files within their personal folders.
7. **Use redirection to create or update files:**
   - Employees add feedback to their appraisals using > or >>.
   - HR manager updates recruitment logs.
8. **Set and use aliases for common tasks:**
   - HR manager, team leads, and employees create aliases for commonly used commands.
9. **View file content using `cat`, `head`, and `tail`:**
   - HR manager or team leads preview performance or payroll files.
10. **Find files or directories using `find` or `grep`:**
    - HR manager or team leads search for specific files by name or content.

8.Government Services System

Scenario

Design and implement a government services system where there are three types of users:

1. **Government Officer:** Manages public records and oversees operations.
2. **Employees:** Handle citizen requests and maintain records.
3. **Citizens:** Submit applications and check status.

File Manager Structure

- `/gov_services`
  - `/officer`: Contains public records, accessible only by the officer.
  - `/employees`: Contains service requests and processed applications.
  - `/citizens`: Contains citizen-specific applications accessible only to them.

Tasks:

1. **List files/directories:**
   - Officer, employees, and citizens list files in their respective directories.
2. **Change permissions of files/directories:**
   - Officer sets permissions for sensitive public records.
3. **Make/delete files/directories:**
   - Officer or employees create or remove directories for new services.
4. **Create symbolic link files:**
   - Officer creates symbolic links for shared policies.
5. **Copy files/directories:**
   - Officer copies public records for archiving.
   - Citizens copy their applications within their directories.
6. **Move files/directories:**
   - Employees reorganize citizen request files.
7. **Use redirection to create or update files:**
   - Citizens submit feedback using > or >>.
8. **Set and use aliases for common tasks:**
   - Government officer or employees create aliases to simplify frequent operations, such as searching for specific records or generating reports.
9. **View file content using `cat`, `head`, and `tail`:**
   - Government officer or employees can preview specific public records or service logs.
10. **Find files or directories using `find` or `grep`:**
    - Government officer or employees search for specific files or records by name or content.

9.Transportation System

Scenario

Design and implement a transportation system where there are three types of users:

1. **Admin:** Manages vehicle data, routes, and schedules.
2. **Drivers:** Access and manage their schedules and routes.
3. **Passengers:** Book tickets and view schedules.

File Manager Structure

- `/transportation`
  - `/admin`: Contains administrative files, accessible only by the admin.
  - `/drivers`: Contains schedules and route information, accessible only by respective drivers.
  - `/passengers`: Contains booking and ticket files for each passenger.

Tasks:

1. **List files/directories:**
   - Admin, drivers, and passengers can list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin sets permissions for sensitive schedule and route data.
3. **Make/delete files/directories:**
   - Admin or drivers can create or remove directories for routes and schedules.
4. **Create symbolic link files:**
   - Admin creates symbolic links for common schedules or policy documents.
5. **Copy files/directories:**
   - Admin copies route files for archiving.
   - Passengers copy ticket files within their personal directories.
6. **Move files/directories:**
   - Drivers reorganize route schedules.
7. **Use redirection to create or update files:**
   - Passengers add feedback to their booking using `>` or `>>`.
8. **Set and use aliases for common tasks:**
   - Admin and drivers create aliases for frequent operations.
9. **View file content using `cat`, `head`, and `tail`:**
   - Drivers preview their schedules, and passengers view ticket summaries.
10. **Find files or directories using `find` or `grep`:**
    - Admin searches for specific route files, and passengers search their tickets.

10.Social Media Platform

Scenario
Design and implement a social media platform where there are three types of users:

1. **Admin:** Manages content, user accounts, and platform data.
2. **Moderators:** Review posts and manage violations.
3. **Users:** Create and view posts and comments.

File Manager Structure

- `/socialmedia`
  - `/admin`: Contains platform data and reports, accessible only by the admin.
  - `/moderators`: Contains flagged posts and user reports, accessible by moderators.
  - `/users`: Contains user-specific posts and comments.

Tasks:

1. **List files/directories:**
    - Admin, moderators, and users can list files in their respective directories.
2. **Change permissions of files/directories:**
    - Admin updates permissions for flagged posts or sensitive user data.
3. **Make/delete files/directories:**
    - Admin or moderators create or remove directories for flagged content.
4. **Create symbolic link files:**
    - Admin creates symbolic links for platform policies or guidelines.
5. **Copy files/directories:**
    - Admin copies flagged content for archiving.
    - Users copy posts within their directories.
6. **Move files/directories:**
    - Moderators reorganize flagged post directories.
7. **Use redirection to create or update files:**
    - Users add comments to posts using `>` or `>>`.

8. Set and use aliases for common tasks:

- Admin and moderators create aliases for frequent operations, such as managing flagged content or viewing user activity.

9. View file content using `cat`, `head`, and `tail:

- Moderators view content for moderation, and users preview their posts or comments.

10. Find files or directories using `find` or `grep:

- Admin searches for specific user posts, and moderators search for posts that violate community guidelines.

11.Retail Store Management System

Scenario
Design and implement a retail store management system where there are three types of users:

1. **Manager:** Manages inventory, sales reports, and store data.
2. **Cashiers:** Handle transactions and billing.
3. **Customers:** View and manage their purchase records.

File Manager Structure

- `/retail_store`

    - `/manager`: Contains administrative files, sales data, and inventory records, accessible only by the manager.
    - `/cashiers`: Contains billing and transaction records, accessible only by respective cashiers.
    - `/customers`: Contains individual purchase history files.

Tasks:

1. **List files/directories:**
    - Manager, cashiers, and customers can list files in their respective directories.
2. **Change permissions of files/directories:**
    - Manager updates permissions for sensitive inventory or sales data.
3. **Make/delete files/directories:**
    - Manager or cashiers create or delete directories for new product records or transactions.
4. **Create symbolic link files:**
    - Manager creates symbolic links to shared policies or inventory summaries.
5. **Copy files/directories:**
    - Manager copies sales data for archiving.
    - Customers copy their purchase history within their folders.
6. **Move files/directories:**
    - Cashiers reorganize transaction logs.
7. **Use redirection to create or update files:**
    - Customers add feedback to their purchase history using > or >>.
8. **Set and use aliases for common tasks:**
    - Manager or cashiers create aliases for frequent operations.
9. **View file content using `cat`, `head`, and `tail`:**
    - Cashiers preview transaction logs, and customers view purchase summaries.
10. **Find files or directories using `find` or `grep`:**
    - Manager or cashiers search for specific sales data or transactions.

12.Logistics and Supply Chain System

Scenario
Design and implement a logistics and supply chain system where there are three types of users:

1. **Admin:** Manages warehouse data, delivery schedules, and inventory.
2. **Warehouse Staff:** Handle stock and shipments.
3. **Customers:** Track their orders and shipment status.

File Manager Structure

- `/logistics`

- `/admin`: Contains inventory, delivery data, and reports, accessible only by the admin.
- `/warehouse`: Contains stock and shipment details.
- `/customers`: Contains order and shipment tracking files.

Tasks:

1. **List files/directories:**
   - Admin, warehouse staff, and customers list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin updates permissions for inventory and delivery schedules.
3. **Make/delete files/directories:**
   - Admin or warehouse staff create or delete directories for new shipments.
4. **Create symbolic link files:**
   - Admin creates symbolic links for shared delivery or inventory files.
5. **Copy files/directories:**
   - Admin copies delivery data for archiving.
   - Customers copy their order tracking files.
6. **Move files/directories:**
   - Warehouse staff reorganize shipment logs.
7. **Use redirection to create or update files:**
   - Customers add notes to their orders using `>` or `>>`.
8. **Set and use aliases for common tasks:**
   - Admin or warehouse staff create aliases for frequent operations.
9. **View file content using `cat`, `head`, and `tail`:**
   - Warehouse staff review shipment logs, and customers view tracking details.
10. **Find files or directories using `find` or `grep`:**
    - Admin searches for specific inventory records.

13.Online Examination System

Scenario

Design and implement an online examination system where there are three types of users:

1. **Admin:** Sets up exams and manages student data.
2. **Instructors:** Upload exam materials and monitor student progress.
3. **Students:** Take exams and view results.

File Manager Structure

- `/examination`
  - `/admin`: Contains administrative files and configurations, accessible only by the admin.
  - `/instructors`: Contains exam question files and grading data, accessible only by respective instructors.
  - `/students`: Contains student-specific exam submissions and results.

Tasks:

1. **List files/directories:**
   - Admin, instructors, and students list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin updates permissions for exam question files and results.
3. **Make/delete files/directories:**
   - Admin or instructors create or delete exam question files and directories for exam schedules.
4. **Create symbolic link files:**
   - Admin creates symbolic links for shared instructions or policies.
5. **Copy files/directories:**
   - Admin copies grading guidelines for instructors.
   - Students copy their submission backups within their directories.
6. **Move files/directories:**
   - Instructors reorganize grading files for clarity.
7. **Use redirection to create or update files:**
   - Students can add notes to their results or feedback files using > or >>.
8. **Set and use aliases for common tasks:**
   - Admin and instructors create aliases for frequent operations.
9. **View file content using `cat`, `head`, and `tail`:**
   - Instructors preview grading files, and students view results.
10. **Find files or directories using `find` or `grep`:**
    - Admin searches for specific exam data, and instructors find questions by keywords.

14.Gaming Platform

Scenario

Design and implement a gaming platform where there are three types of users:

1. **Admin:** Manages servers, user data, and logs.
2. **Moderators:** Monitor player behavior and manage violations.
3. **Players:** Access games, view profiles, and interact with the community.

File Manager Structure

- `/gaming`

- `/admin`: Contains server logs, configurations, and reports, accessible only by the admin.
- `/moderators`: Contains violation logs and user reports.
- `/players`: Contains user-specific game profiles and save data.

Tasks:

1. **List files/directories:**
   - Admin, moderators, and players list files in their respective directories.
2. **Change permissions of files/directories:**
   - Admin sets permissions for sensitive server or violation logs.
3. **Make/delete files/directories:**
   - Admin or moderators create or remove directories for new logs or reports.
4. **Create symbolic link files:**
   - Admin creates symbolic links for shared policies or guidelines.
5. **Copy files/directories:**
   - Admin copies server logs for archiving.
   - Players copy save data within their directories.
6. **Move files/directories:**
   - Moderators reorganize violation logs.
7. **Use redirection to create or update files:**
   - Players add notes to their profiles or game settings using `>` or `>>`.
8. **Set and use aliases for common tasks:**
   - Admin and moderators create aliases for frequent operations.
9. **View file content using `cat`, `head`, and `tail`:**
   - Moderators preview reports, and players view profile details.
10. **Find files or directories using `find` or `grep`:**
    - Admin searches for specific logs, and moderators search violations by keywords.

15.Manufacturing System

Scenario
Design and implement a manufacturing system where there are three types of users:

1. **Factory Manager:** Oversees production schedules, inventory, and equipment reports.
2. **Workers:** Log tasks, equipment usage, and production output.
3. **Inspectors:** Check production quality, log issues, and create inspection reports.

File Manager Structure

- `/manufacturing`

- `/manager`: Contains production schedules, inventory, and reports, accessible only by the factory manager.
- `/workers`: Contains task logs, equipment usage records, and production output data, accessible only by respective workers.
- `/inspectors`: Contains inspection reports and quality issue logs.

Tasks:

1. **List files/directories:**
   - Manager, workers, and inspectors list files in their respective directories.

2. **Change permissions of files/directories:**
   - Manager sets permissions for production schedules and inventory files.

3. **Make/delete files/directories:**
   - Manager or workers create or delete directories for tasks or logs.
   - Inspectors can log new issues or remove outdated reports.

4. **Create symbolic link files:**
   - Manager creates symbolic links for shared schedules or policies.

5. **Copy files/directories:**
   - Manager copies production schedules or equipment reports for archiving.
   - Workers copy task logs or production outputs for reference.

6. **Move files/directories:**
   - Workers reorganize task logs, and inspectors move resolved issue logs to an archive.

7. **Use redirection to create or update files**
   - Workers add task notes or output details using > or >>.
   - Inspectors log quality checks.

8. **Set and use aliases for common tasks:**
   - Manager, workers, and inspectors create aliases for frequent operations.

9. **View file content using `cat`, `head`, and `tail`:**
   - Inspectors preview reports, and workers review output data.

10. **Find files or directories using `find` or `grep`:**
    - Manager searches for specific schedules, and inspectors search for issue details.

Task 2 Guidelines

You will implement one of the scheduling algorithms using Any programming Languages like (C, C++,Python,Java).

- The number of processes should be user-defined (e.g. user enters them at run time).

- All process data (e.g. process arrival time, burst time, etc.) must be user-defined.

- Validation for the input data must be considered (e.g. user can't enter negative numbers or characters in the input fields).

- Waiting Time, Turnaround Time, and Response Time for each process should be calculated.

- Average Waiting Time, Average Turnaround Time, and Average Response Time should be calculated.

Task 2 ideas:

1- First-Come , First-Served

2- Shortest Job First (Non-Preemptive)

3- Shortest Job First (Preemptive)

4- Priority (Preemptive)

5- Round Robin