|  |
| --- |
| Photo displaying partial image of two pie charts on a canvas-textured page |
| XOS  THE OS |
| |  |  |  | | --- | --- | --- | |  |  |  | |

Contents

[TEAM 1](#_Toc138055505)

[Introduction 2](#_Toc138055506)

[Contents 3](#_Toc138055507)

[APL compiler 3](#_Toc138055508)

[SPL compiler 3](#_Toc138055509)

[XFS-INTERFACE 3](#_Toc138055510)

[XSM Machine Simulator 3](#_Toc138055511)

[Experiments 3](#_Toc138055512)

[Step 1: Setting up the System 3](#_Toc138055513)

[Process 3](#_Toc138055514)

[Step 2: Understanding the File System 4](#_Toc138055515)

[Step 3: Starting the Machine 7](#_Toc138055516)

[Step 4: Running a User Program 9](#_Toc138055517)

[Step 5: Interrupt routines 12](#_Toc138055518)

[Step 6: Getting started with multi-programming 15](#_Toc138055519)

[The New Version of XOS – eXpOS: 19](#_Toc138055520)

[Quick Review 19](#_Toc138055521)

[Installation 19](#_Toc138055522)

# TEAM

|  |  |
| --- | --- |
| Name | Roll number |
| Akkina Aatesh | CB.EN.U4CSE21404 |
| B V V Satyanarayana | CB.EN.U4CSE21409 |
| Bhavana N | CB.EN.U4CSE21411 |
| Nadimpalli Ujwal Srimanth Varma | CB.EN.U4CSE21440 |
| Potu Tejaswi | CB.EN.U4CSE21445 |

# Introduction

XOS, also known as the Experimental Operating System, is a fictional operating system commonly depicted in science fiction literature, movies, and video games. It is often portrayed as an advanced and futuristic operating system with highly advanced capabilities.While XOS does not exist in reality, its characteristics and features are typically designed to showcase cutting-edge technology and futuristic concepts. Some common elements associated with XOS include:Advanced User Interface

XOS often features a sleek and visually appealing user interface, incorporating holographic displays, augmented reality, or gesture-based controls. It may provide users with immersive and intuitive ways to interact with their devices.Artificial Intelligence Integration

XOS is typically portrayed as having a sophisticated artificial intelligence system that can understand and respond to natural language commands, provide personalized assistance, and adapt to the user's preferences and behavior.Seamless Connectivity

XOS may offer seamless integration with various devices, networks, and services, allowing users to effortlessly control and access different technologies within their environment. This can include IoT devices, smart homes, autonomous vehicles, and more.Enhanced Security

XOS often incorporates advanced security features, such as biometric authentication, robust encryption, and real-time threat detection and prevention mechanisms. This ensures the user's data and privacy are safeguarded in the futuristic world.Multi-Platform Compatibility

XOS may be designed to work across different types of devices, including smartphones, tablets, computers, wearables, and even virtual or augmented reality headsets. It aims to provide a consistent and interconnected experience across these platforms.Advanced Automation

XOS is often depicted as being capable of automating various tasks and processes, simplifying the user's daily life. It can control home appliances, schedule appointments, manage personal and professional tasks, and even make autonomous decisions based on user preferences.

# Contents

## APL compiler

The APL compiler in XOS would take APL code written by a programmer and convert it into machine code or an intermediate representation that can be executed by the XOS operating system

## SPL compiler

The SPL compiler in XOS would take SPL code written by a programmer and convert it into machine code or an intermediate representation that can be executed by the XOS operating system

## XFS-INTERFACE

In the context of XOS, a fictional operating system, the XFS-INTERFACE refers to a component or interface designed to facilitate communication and interaction between the XOS operating system and the XFS (eXtensible File System) file system.The XFS-INTERFACE in XOS serves as a bridge or connector that allows the operating system to access and manipulate files and directories stored within the XFS file system. It provides a set of functions, protocols, or APIs (Application Programming Interfaces) that enable applications and system components in XOS to perform file-related operations, such as reading, writing, creating, deleting, and modifying files and directories.

## XSM Machine Simulator

The "xsm" directory contains the XSM (eXperimental String Machine) simulator. This simulator emulates the behavior of the XSM machine, providing a simulated environment for executing programs and testing the operating system’s functionality.

# Experiments

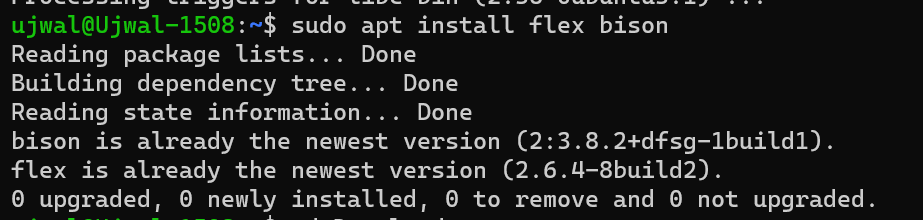
## Step 1: Setting up the System

### Process

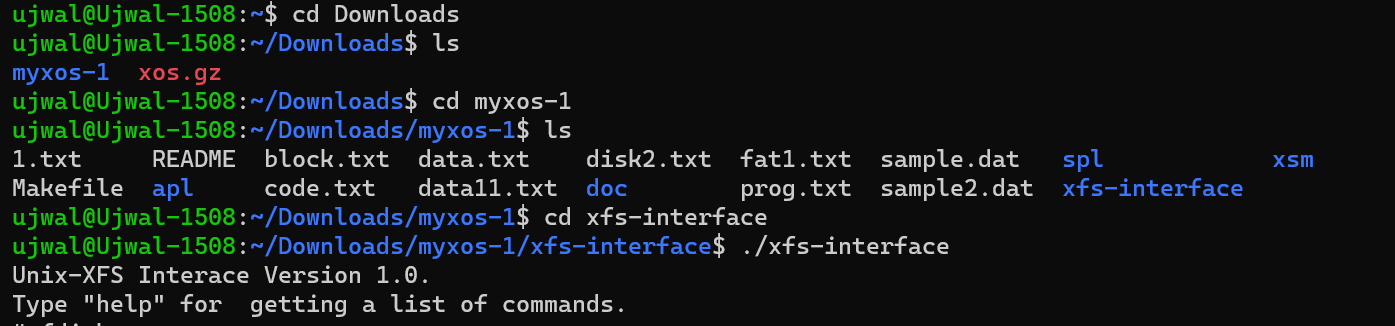
XOS is downloaded from its official page. The link is provided below. We have extracted it and pasted it into the home directory named ujwal

<https://xosnitc.github.io/downloads.html>

We even installed Flex and bison so that XOS works properly



Proof that XOS is installed



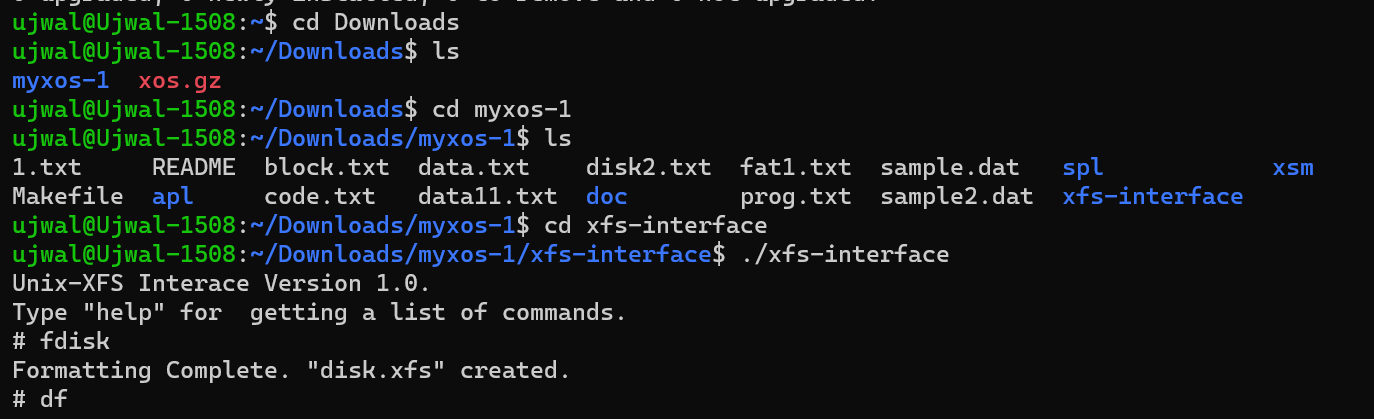
## Step 2: Understanding the File System

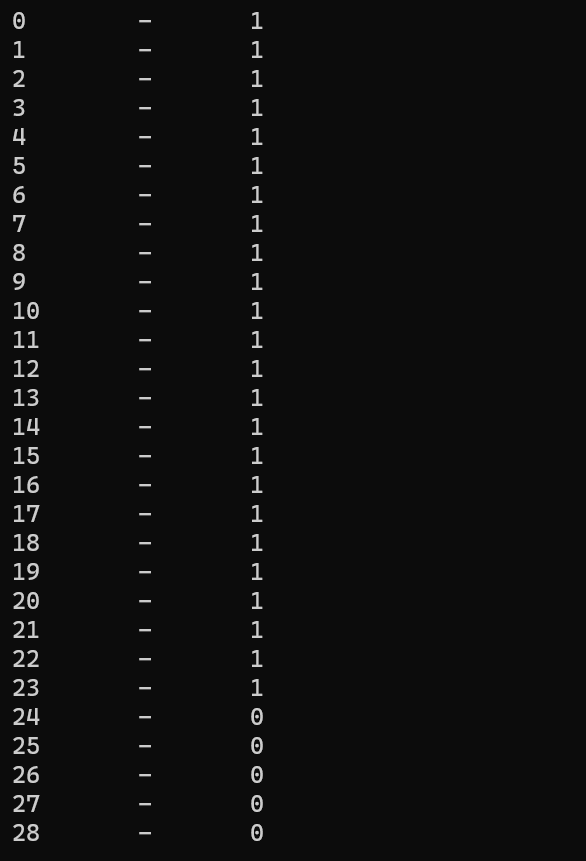
Commands used:-

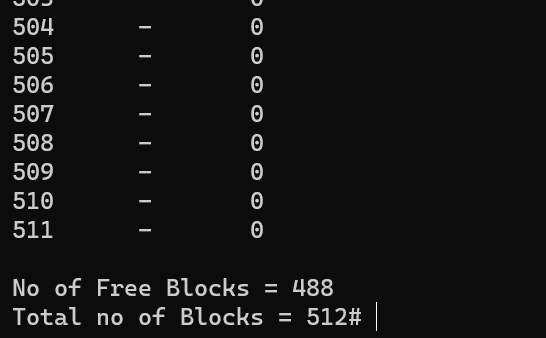
./xfs-interface

Fdisk

Df

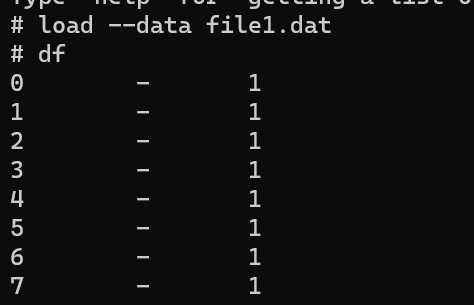


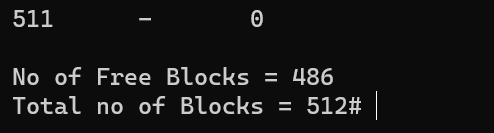




Command used : nano file1.dat

load --data file1.dat





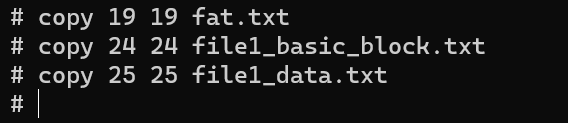
As we can see the number of free blocks after that operation changed from 488-486

Commands used:-

copy 19 19 fat.txt

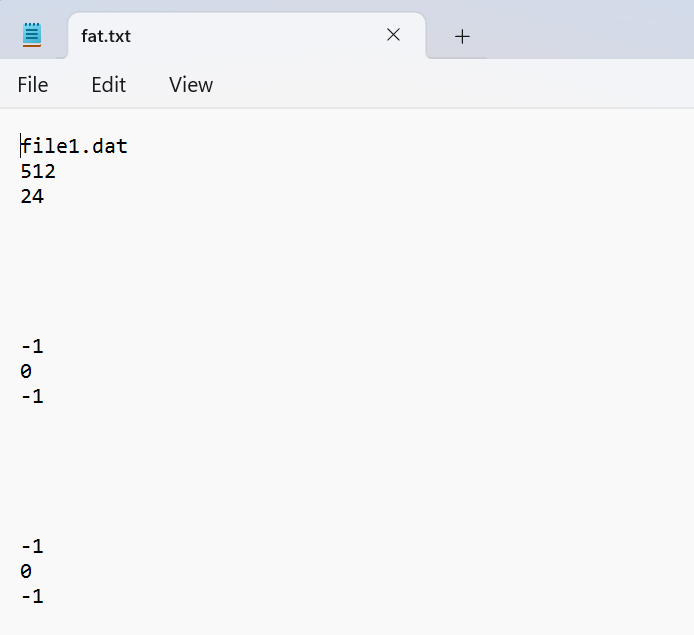
copy 24 24 file1\_basic\_block.txt

copy 25 25 file1\_data.txt



Files :-

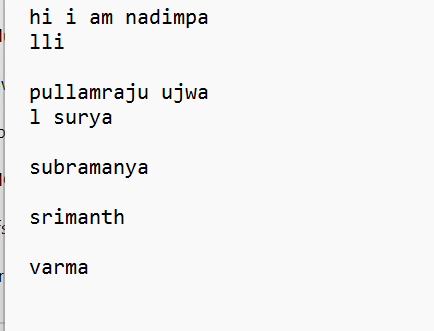
Fat.txt



File1\_basic\_block.txt



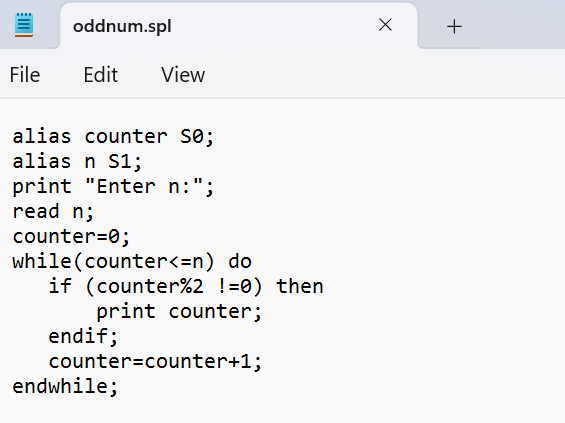
File1\_data.txt

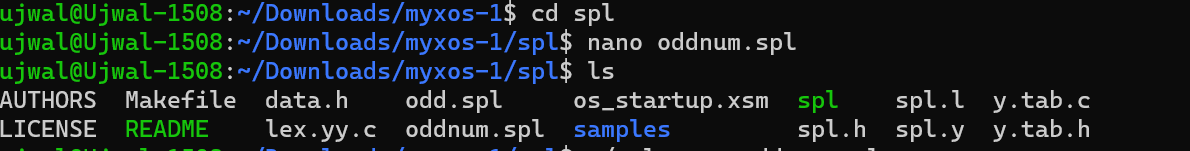


## Step 3: Starting the Machine

nano oddno.spl

./spl --os oddnum.spl

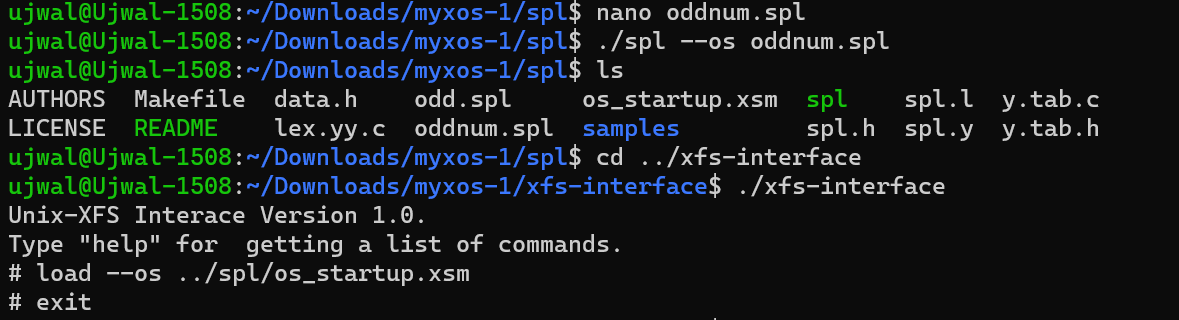




Commands used:-

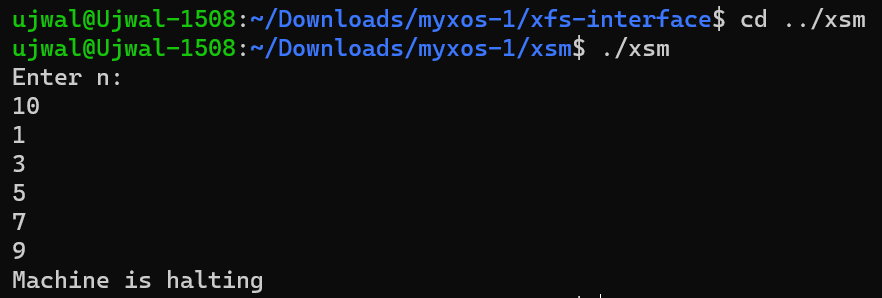
./xfs-interface

load --os ../spl/os\_startup.xsm



Commands used:-

./xsm



## Step 4: Running a User Program

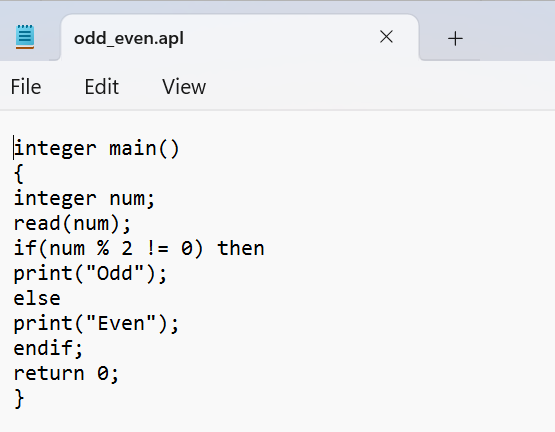
Commands used:-

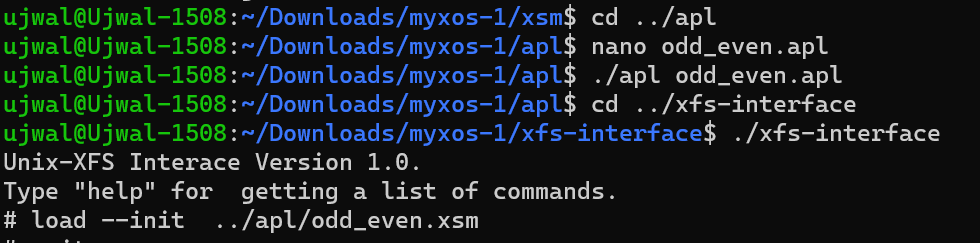
nano odd\_even.apl

./apl odd\_even.apl

./xfs-interface

load --init ../spl/os\_startup.xsm

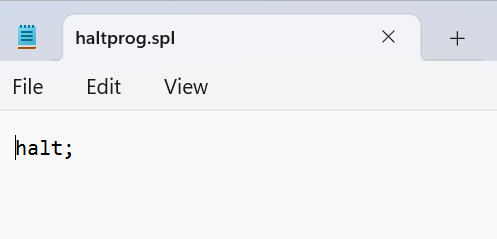




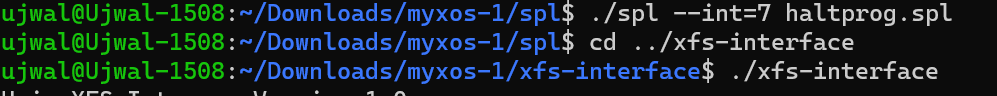
Commands:-

nano haltprog.spl

./spl –int=7 haltprog.spl



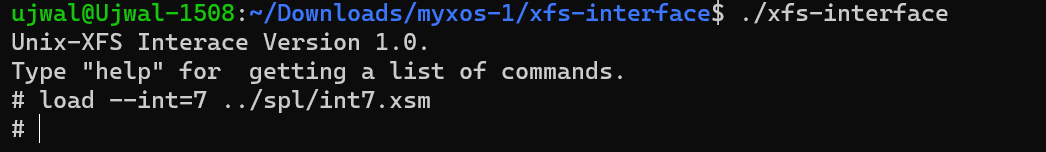




Commands used

./xfs-interface

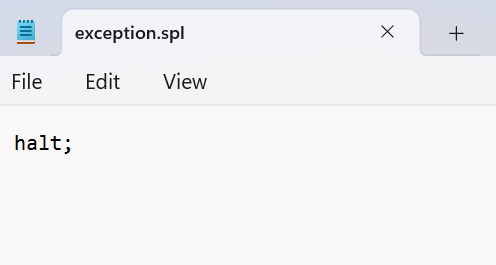
Load --int=7 ../spl/int7.xsm

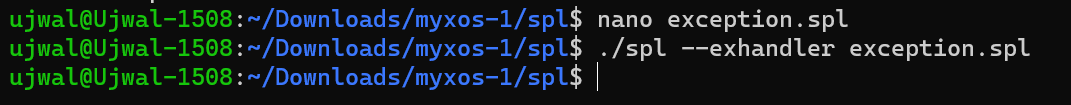


Commands used

nano exception.spl

./spl --exhandler execption.spl

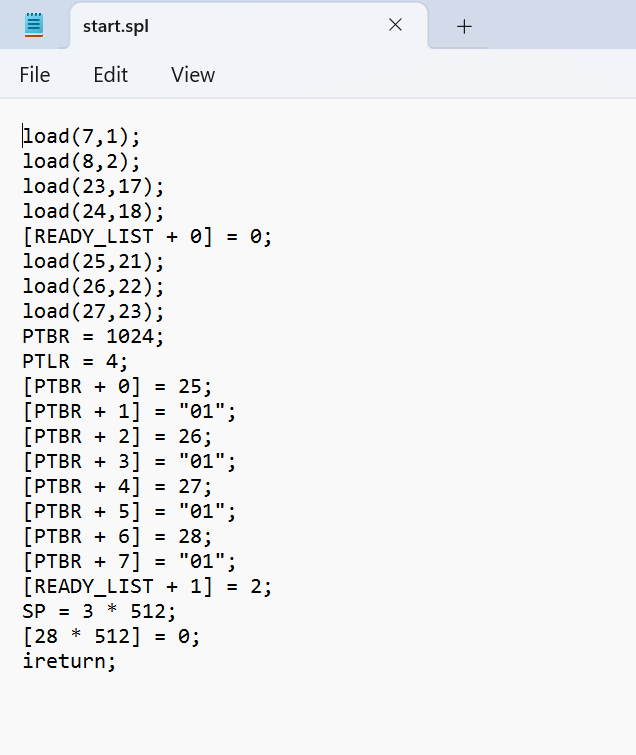


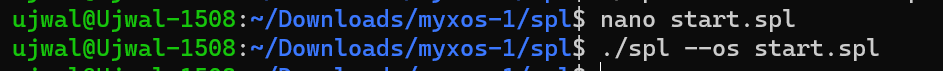


Commands used:-

nano start.spl

./spl --os start.spl

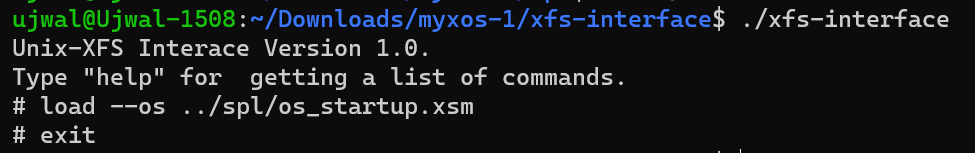




Commands used-

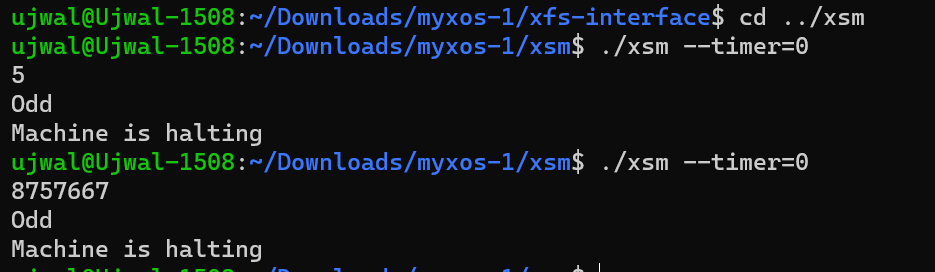
./xfs-interface

load --os ../spl/os\_startup.xsm



Commands used:-

./xsm --timer=0

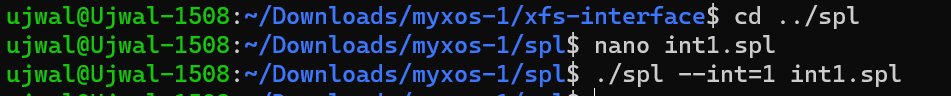


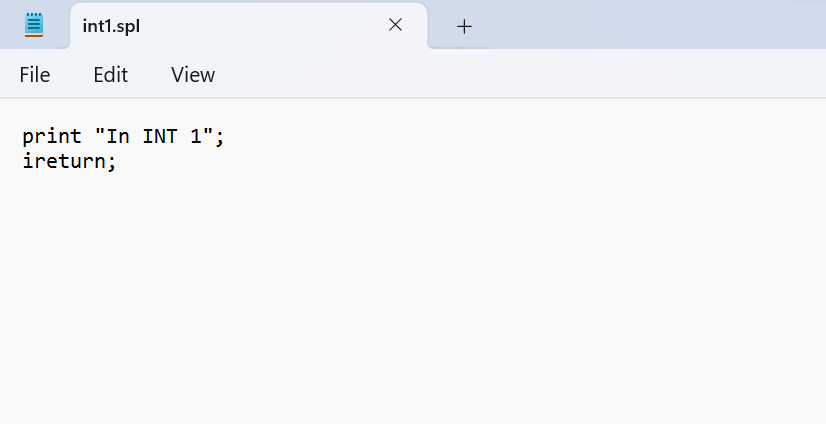
## Step 5: Interrupt routines

Commands used:

nano int1.spl

./spl --int=1 int1.spl

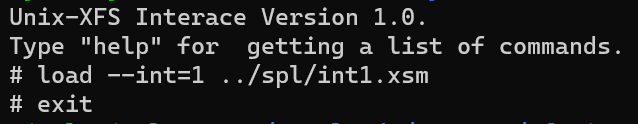




Commands used:-

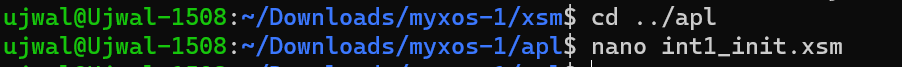
./xfs-interface

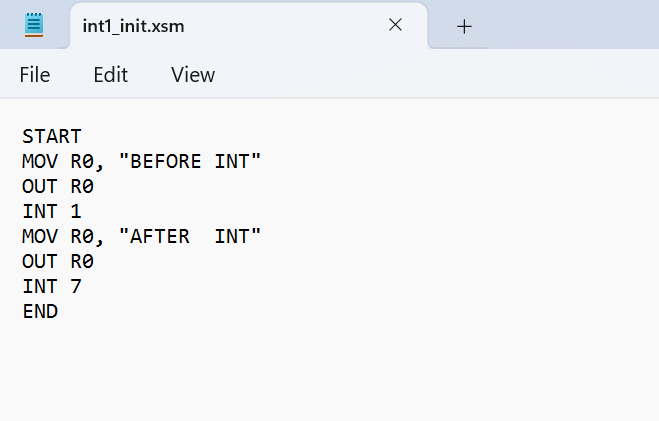
load --int=1 ../spl/int1.xsm



Commands used:

nano int1\_init.xsm

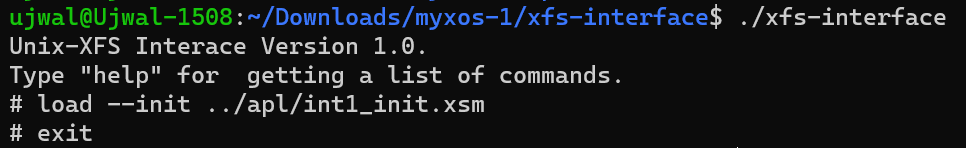




Commands used:

./xfs-interface

load --init ../apl/int1\_init.xsm

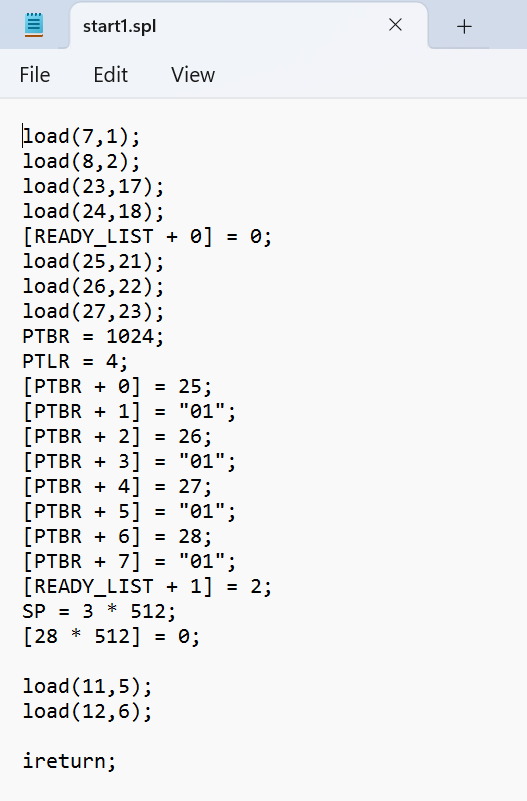


Commands used:-

nano start1.spl

./spl --os start1.spl

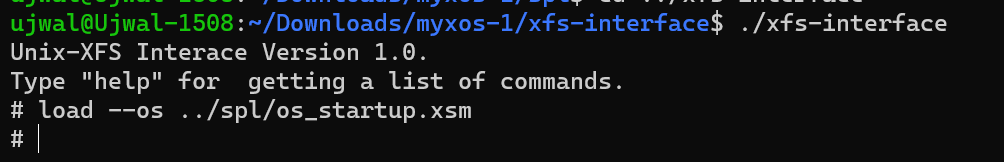




Commands used:

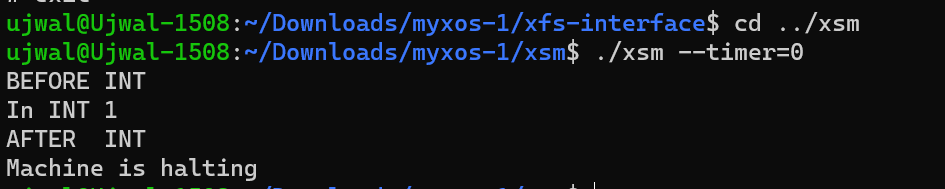
./xfs-interface

load --os ../spl/os\_startup.xsm



Commands used:

./xsm –timer=0



## Step 6: Getting started with multi-programming

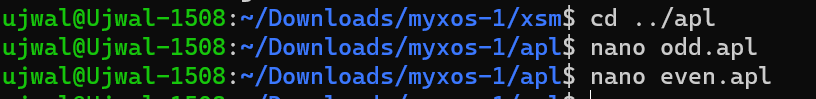
Commands used

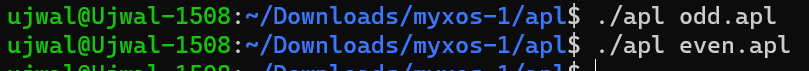
nano odd.spl

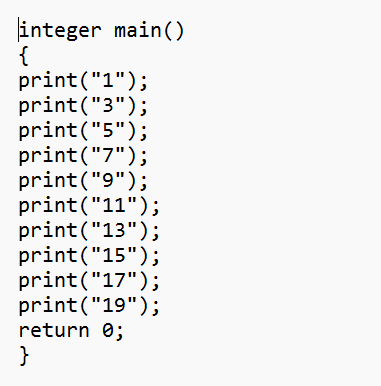
nano even.spl

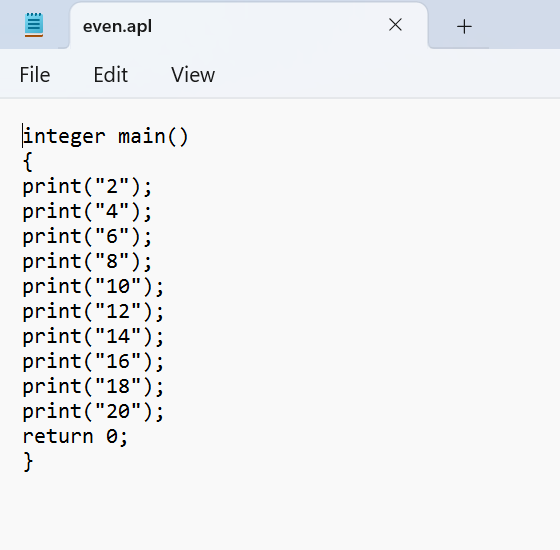
./apl odd.apl

./apl even.apl









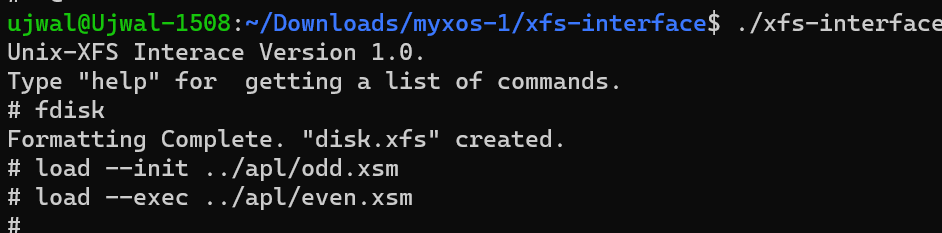
Commands used:-

./xfs-interface

fdisk

load --init ../apl/odd.xsm

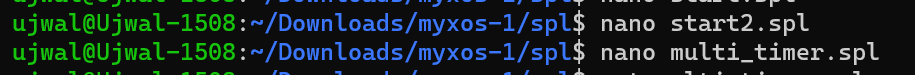
load --exec ../apl/even.xsm

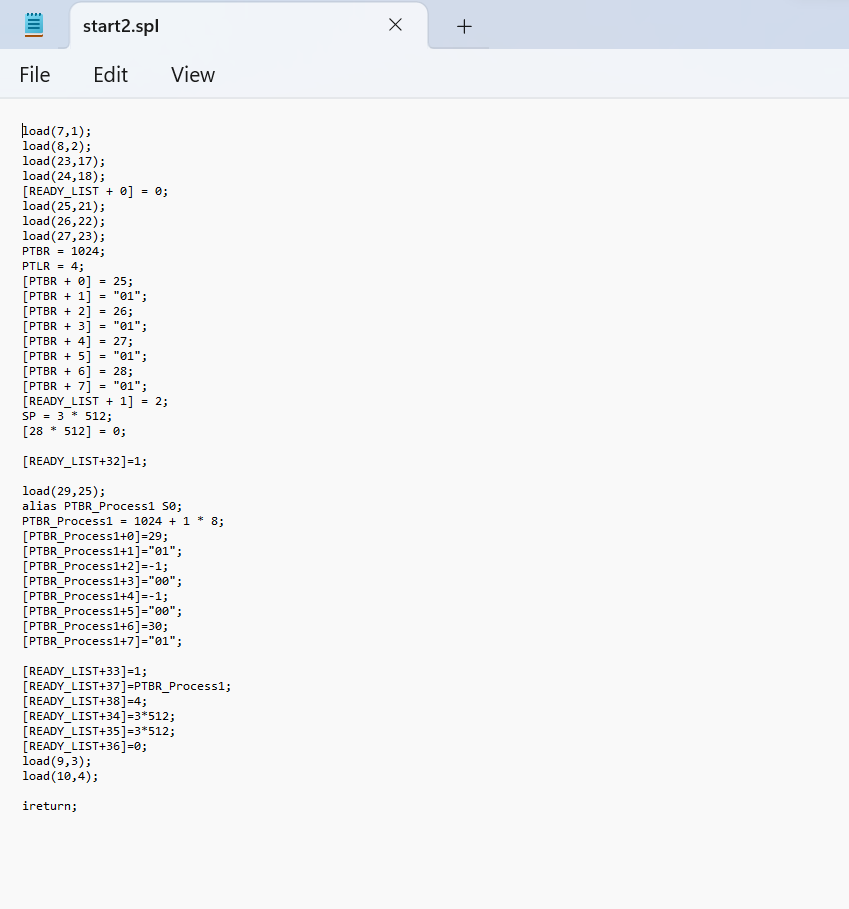


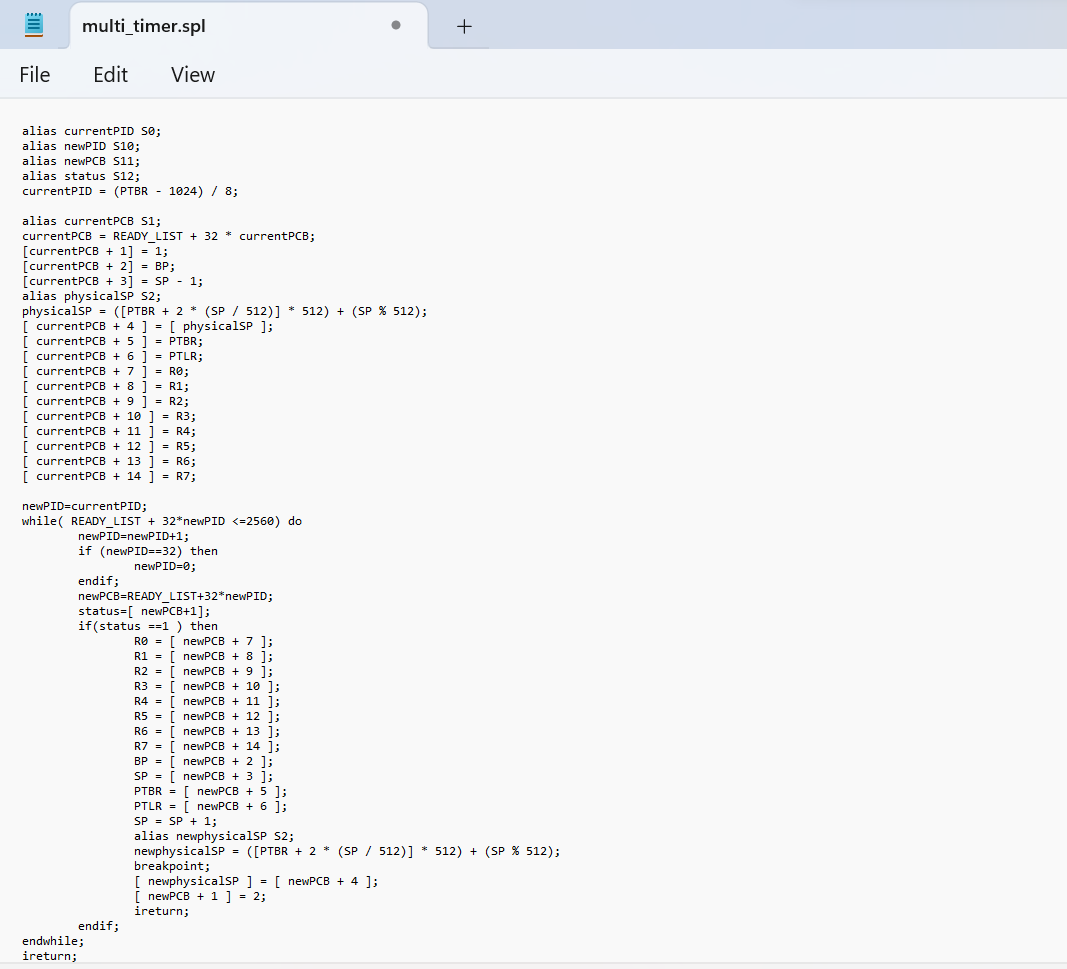
Commands used:-

nano start2.spl

nano multi\_timer.spl

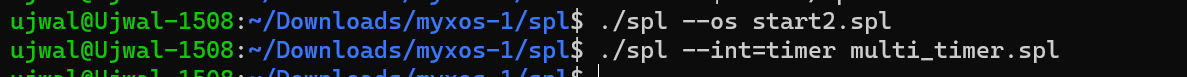






./spl –os start2.spl

./spl –int=timer multi\_timer.spl

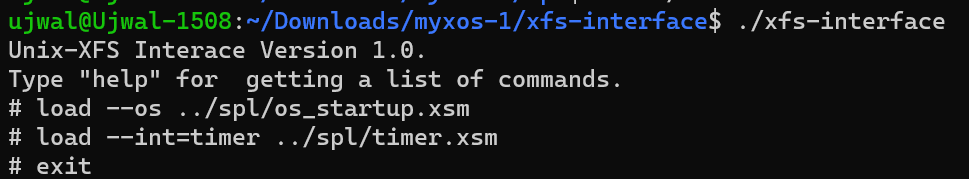


Commands used:-

./xfs-interface

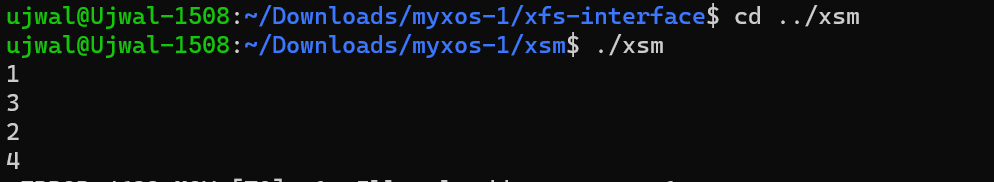
load --os ../spl/os\_startup.xsm

load --int=timer ../spl/timer.xsm



Commands used:

/xsm



# The New Version of XOS – eXpOS:

## Quick Review

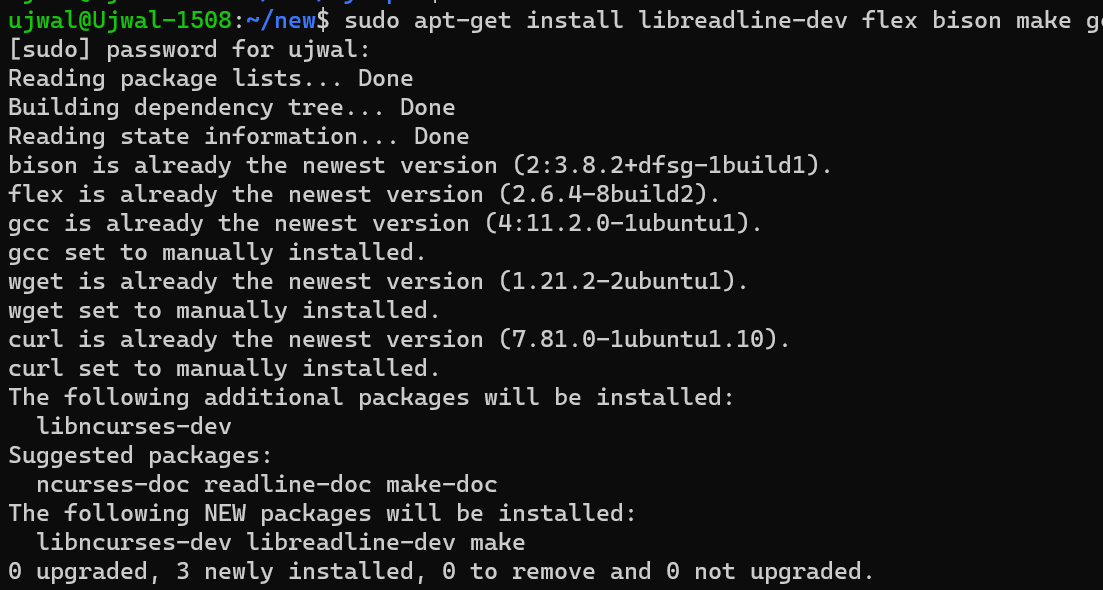
Project eXpOS is an educational initiative created by students as an improved version of XOS, the predecessor operating system. It serves as a valuable tool for students to gain hands-on experience and practical understanding of operating system concepts. The core objective of eXpOS is to facilitate the implementation of essential OS data structures and functionalities by utilizing a simulated machine called the XSM (Experimental String Machine).Developed using the specialized language SPL (System Programmer's Language), eXpOS allows students to explore and build upon the foundations of operating system design. They can delve into various aspects of the OS, including memory management, process scheduling, file systems, and more.Additionally, eXpOS provides students with the opportunity to develop application programs using APL (Application Programmer's Language), enabling them to leverage the OS features and create practical applications that run within the eXpOS environment.By engaging with eXpOS, students can enhance their comprehension of operating system internals while gaining valuable hands-on experience in a controlled educational setting.

## Installation

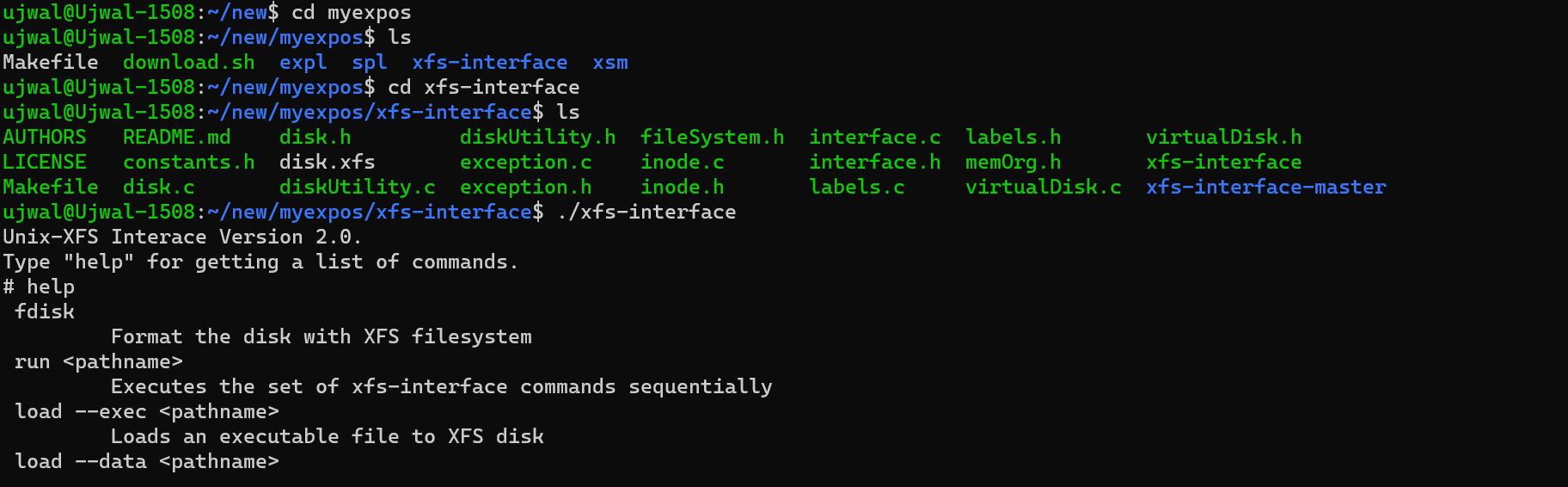
First, we need to download the eXpOS files from their official link given below

<http://exposnitc.github.io/support_tools-files/setting-up.html>

same like XOS we can should even install libreadline-dev, flex, bison, make and gcc to make eXpOS work properly



Proof that the new xos is installed properly



Here you can see the execution interface of eXpOS above.