

بسمه تعالی		
گزارش کار آزمایشگاه سیستم عامل		
تاریخ: 1402/11/15		
نام و نام خانوادگی: صبیحه دردآب	شماره دانشجویی: 40073116	استاد درس: استاد لویمی
هدف آزمایش:		<ul style="list-style-type: none"> آشنایی با نحوه نصب یک توزیع لینوکس به صورت مجازی آشنایی با دستورات اولیه سیستم عامل لینوکس و کار با فایل کامپایل و اجرای کد در محیط لینوکس
عناوین آزمایش:		<ul style="list-style-type: none"> نصب یک ماشین مجازی نصب توزیع Ubuntu روی ماشین مجازی نصب شده اجرای دستورات اولیه و پرکاربرد کار با فایل کامپایل و اجرای کد

نصب و آشنایی با ماشین مجازی (Virtual Machine (VM):

ماشین مجازی یک برنامه نرم افزاری است که مشابه یک رایانه مجازی عمل می کند. این برنامه بر روی سیستم عامل فعلی ما (سیستم میزبان) اجرا می شود و سخت افزاری مجازی برای یک سیستم عامل میهمان ایجاد می کند. سیستم عامل میهمان دقیقاً مانند هر برنامه دیگری در یک پنجره بر روی سیستم عامل میزبان اجرا می شود. این امکان را به کاربر می دهد تا سیستم عامل ها و برنامه های مختلف را بدون نیاز به تغییرات سخت افزاری بر روی یک سیستم فیزیکی اجرا کند. به عبارت دیگر، ماشین مجازی ایجاد محیطی مجازی می کند که در آن می توان سیستم عامل های مختلف را به طور همزمان اجرا کرد.

دوتا از ماشین های مجازی معروف عبارتند از:

1. VMware

2. Oracle VM VirtualBox



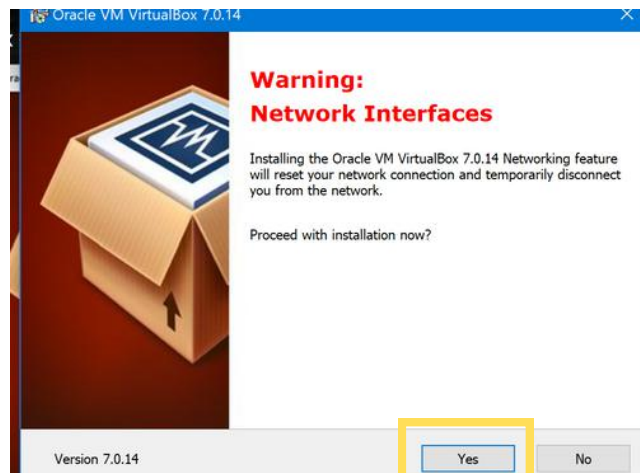
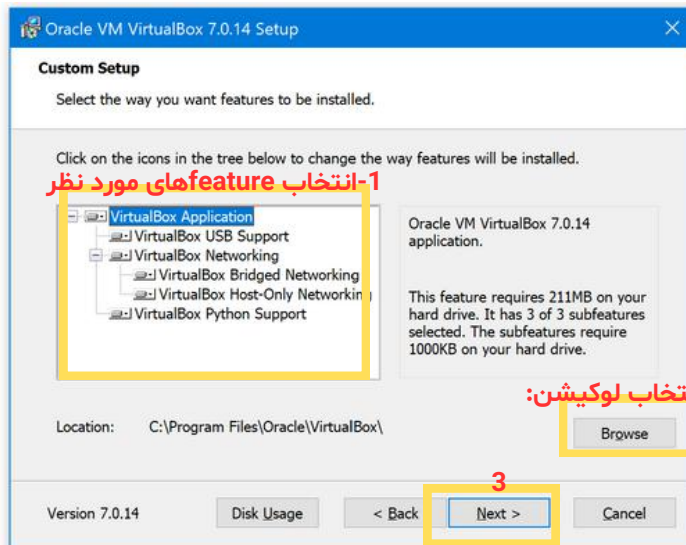
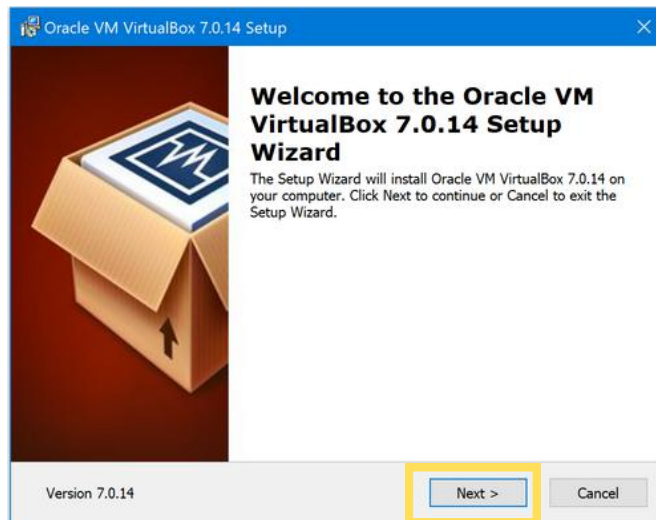
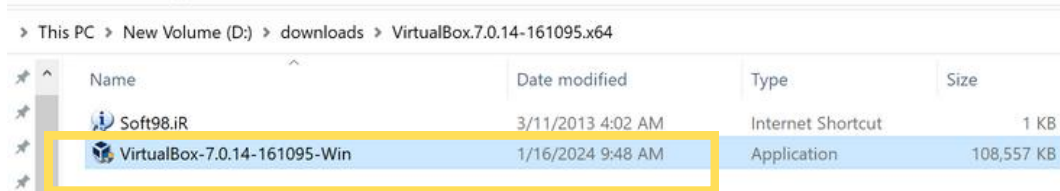
• VMware:

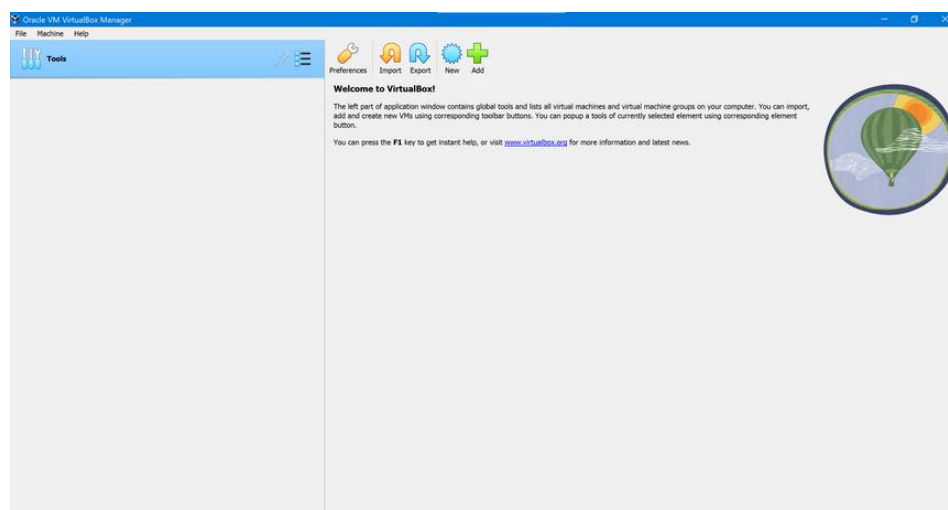
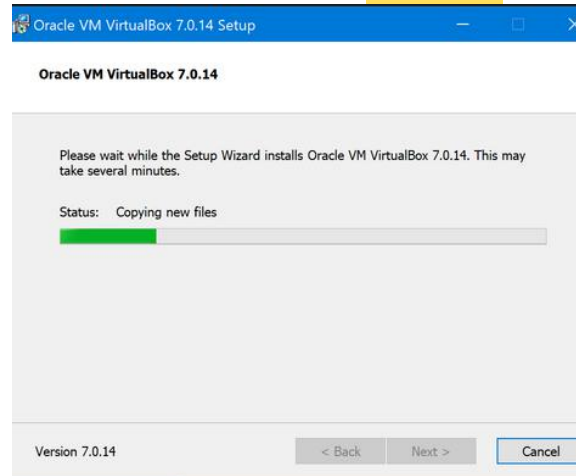
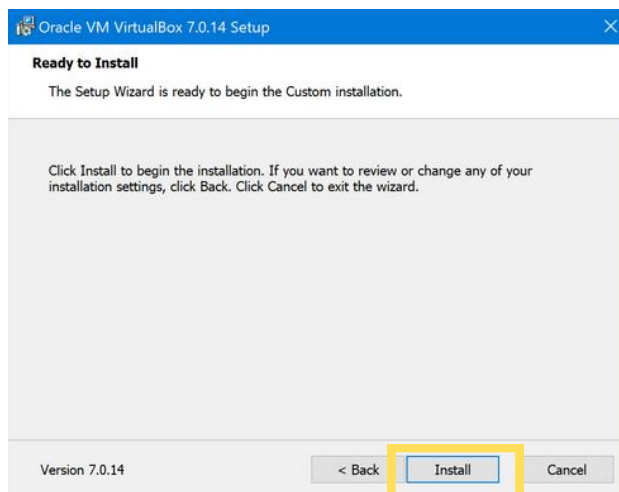
عموماً برای محیط های سروری و حرفه ای ترجیح داده می شود. (نیاز به یک لایسنس دارد ، اما نسخه رایگان آن نیز وجود دارد). دارای قابلیت های پیشرفته تری برای مدیریت و کنترل ماشین های مجازی است. قابل استفاده بر روی بسیاری از سیستم های عامل میزبان مانند ویندوز، مک، و لینوکس است.

• Oracle VM VirtualBox:

برای استفاده شخصی، آموزشی و توسعه نرم افزارها مناسب است. رایگان است و قابل استفاده بر روی بسیاری از سیستم های عامل میزبان مانند ویندوز، مک، و لینوکس است.

سپس طی کردن مراحل زیر:





در خیلی از جوانب، لینوکس با سیستم‌عامل‌های دیگری مانند ویندوز، مک OS و ... شباهت دارد. لینوکس مانند بقیه‌ی سیستم‌عامل‌ها می‌تواند محیط گرافیکی، ویرایشگر متن و تصویر، نمایشگر ویدئو و صدا و خیلی از امکانات دیگر را داشته باشد. در واقع، وجود این شباهت‌ها به این معنی است که اگر با سایر سیستم‌عامل‌ها کار کرده باشید، به راحتی می‌توانید با لینوکس هم کار کنید! اما در کنار این شباهت‌ها، تفاوت‌های مهمی هم وجود دارد. احتمالاً مهمترین تفاوت آن، متن‌باز بودنش است. یعنی منبع کد لینوکس، به صورت عمومی قابل دیدن و ویرایش کردن است. متن‌باز بودن لینوکس به پیشرفت آن کمک بسیاری کرده است. توسعه‌دهندگان با مهارت‌های گوناگون از سراسر دنیا، به بهبود آن کمک کردند و این همکاری سراسری، آن را به یکی از قدرتمندترین و امن‌ترین سیستم‌عامل‌های حال حاضر دنیا تبدیل کرده است. تفاوت مهم دیگر لینوکس، قابلیت بالای آن در شخصی‌سازی است. به راحتی می‌توان ظاهر گرافیکی را عوض کرد، از ادیتورهای مختلف استفاده کرد و قابلیت‌های گوناگون را به آن اضافه یا از آن کم کرد.

لینوکس به دلیل متن‌باز بودن، به صورت رایگان در اختیار همه است و رسماً کسی صاحب آن نیست. هرچند نام تجاری Linux به اسم آقای Linus Torvalds، خالق آن، ثبت شده است. آقای Torvalds لینوکس را اولین بار در سال ۱۹۹۱، وقتی دانشجوی دانشگاه هلسینکی بود، ساخت. سپس آن را به عنوان یک کد متن‌باز منتشر کرد و توانست توجه‌های بسیاری را به سیستم عامل خود جذب کند.

توزیع لینوکس چیست؟

لینوکس دارای چندین نسخه مختلف برای استفاده همه کاربران است. از کاربران تازه‌کار گرفته تا کاربران حرفه‌ای، هرکسی نسخه‌ای از لینوکس متناسب با نیازهای خود خواهد یافت. به این نسخه‌ها توزیع (distribution) یا به اختصار distro می‌گویند. اکثر توزیع‌های لینوکس را می‌توان به صورت رایگان دانلود و نصب کرد. تعدادی از توزیع‌های محبوب لینوکس عبارت‌اند از:

- [Debian](#)
- [Red Hat](#)
- [Arch](#)
- [Fedora](#)
- [Ubuntu](#)

اوبونتو یکی از توزیع‌های لینوکسی است که به طور ویژه برای مبتدیان پیشنهاد می‌شود.

نصب Ubuntu:

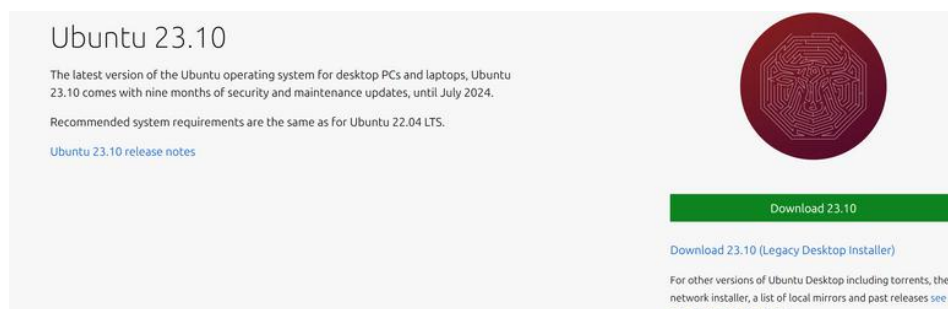
دانلود فایل ISO از <https://ubuntu.com/download/desktop>

1. نسخه LTS (Long-Term Support):

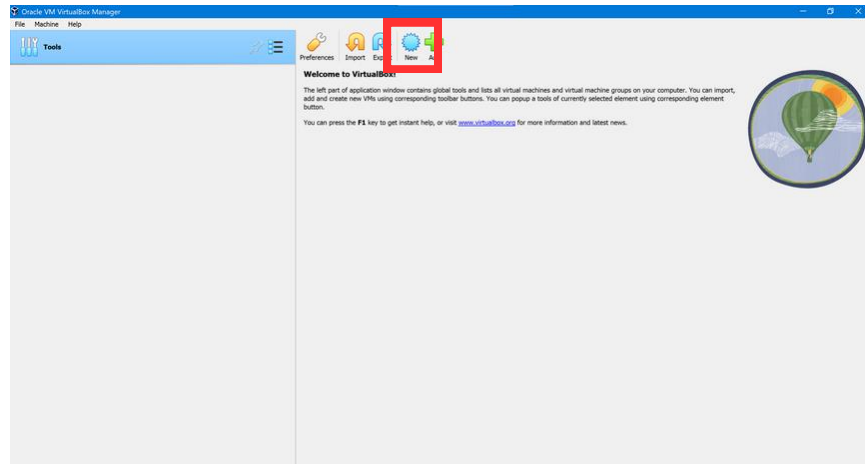
- نسخه LTS به معنی پشتیبانی بلندمدت است. این نسخه‌ها معمولاً هر دو سال یکبار منتشر می‌شوند و برخلاف نسخه‌های معمولی، حداقل 5 سال پشتیبانی از آنها تضمین می‌شود.

2. نسخه‌های معمولی:

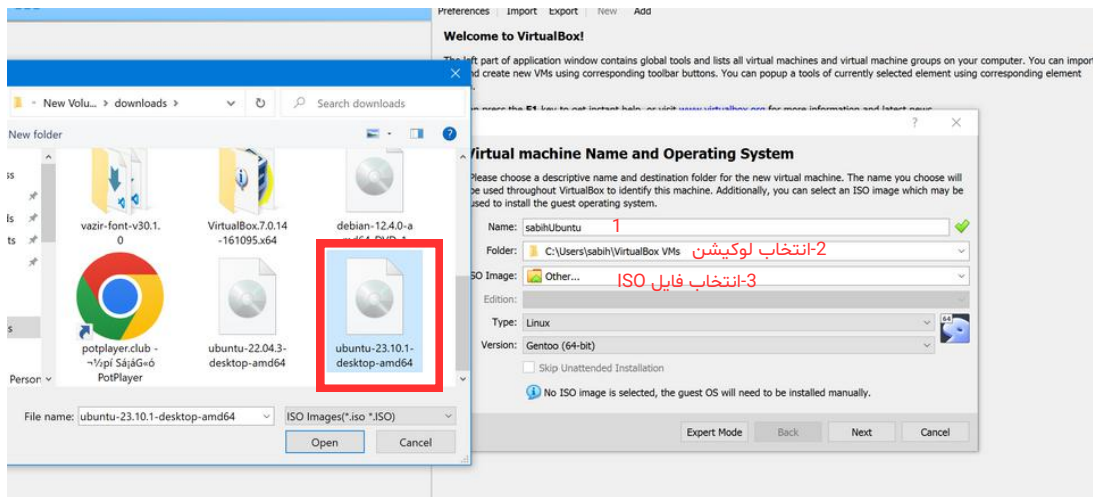
- نسخه‌های معمولی اوبونتو هر 6 ماه یکبار منتشر می‌شوند و عموماً مدت پشتیبانی آنها به مدت 9 ماه یا یک سال است. این نسخه‌ها به دلیل فراوانی بروزرسانی‌ها و تغییرات، ممکن است برای محیط‌های توسعه و آزمایشی مناسب باشند اما برای سرورها و ... بهتر است از نسخه LTS استفاده شود.



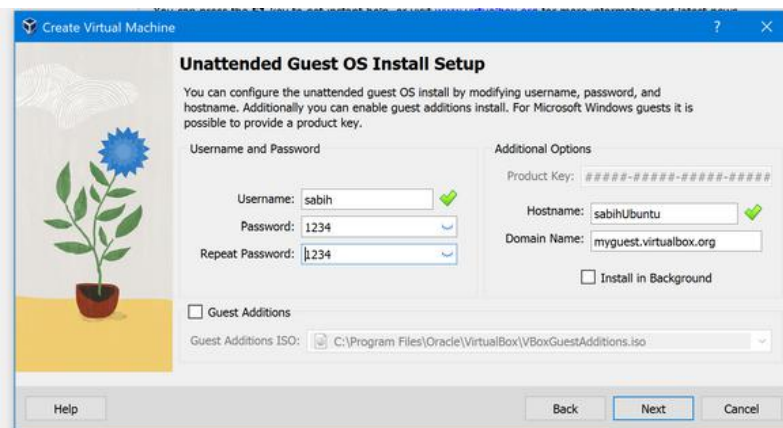
• نرم افزار VirtualBox را باز کرده و بر روی دکمه New کلیک می کنیم:



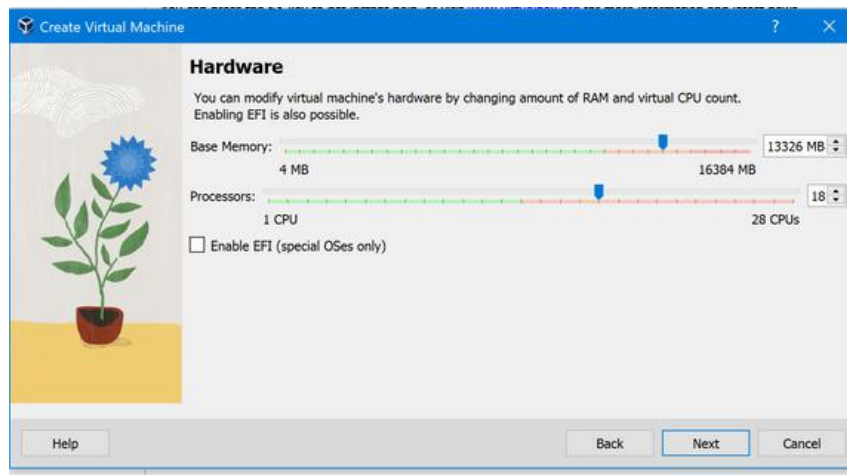
• در این مرحله باید نام سیستم عاملی که می خواهیم نصب کنیم را بنویسیم. سپس لوکیشن را مشخص کنیم و فایل ISO را انتخاب کنیم. سپس باید به طور خودکار Type و Version را تشخیص داده شود. در نهایت روی next کلیک می کنیم:



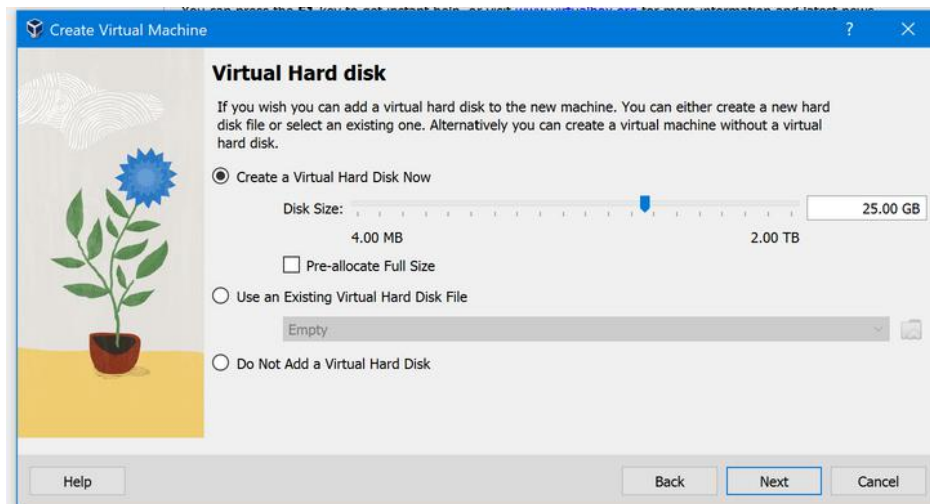
• تعیین یوزرنیم و پسورد:



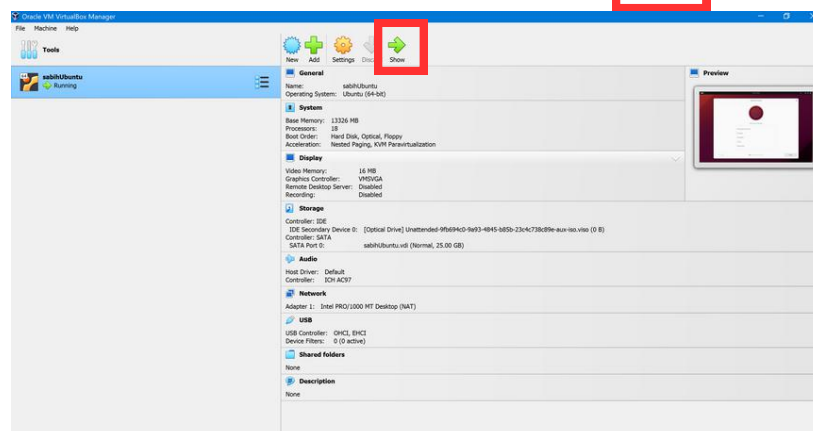
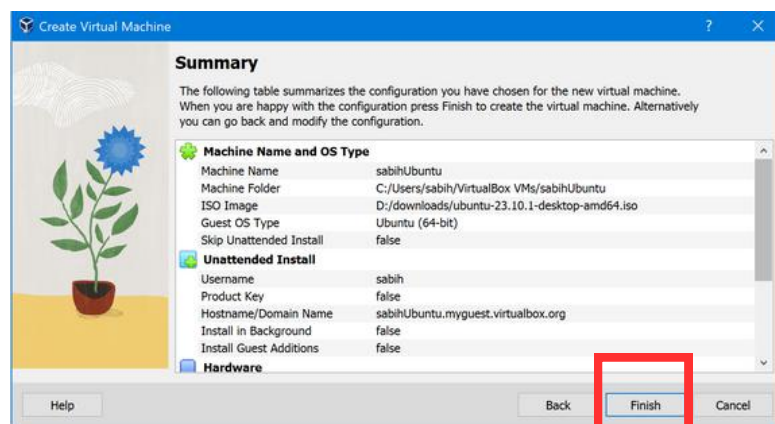
- For good performance it's recommended to provide your VM with around 8GB of RAM (although 4GB will still be usable) and 4 CPUs. Try to remain in the green areas of each slider to prevent issues with your machine running both the VM and the host OS.

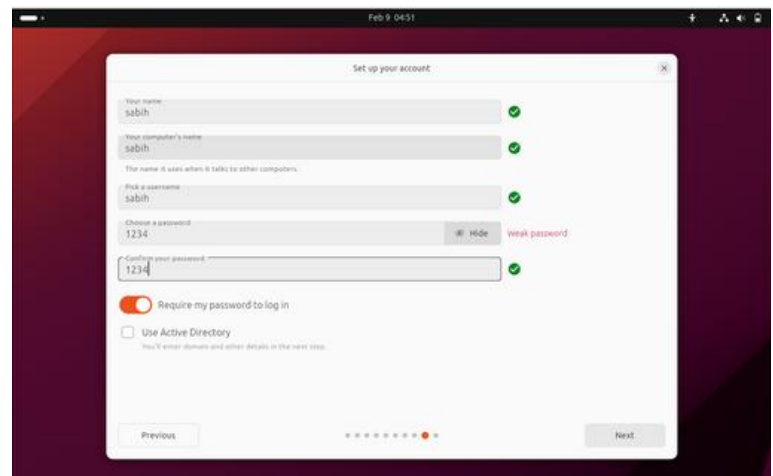
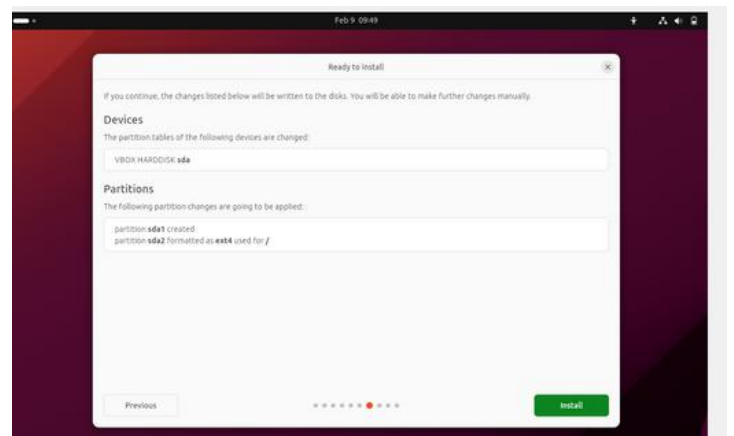
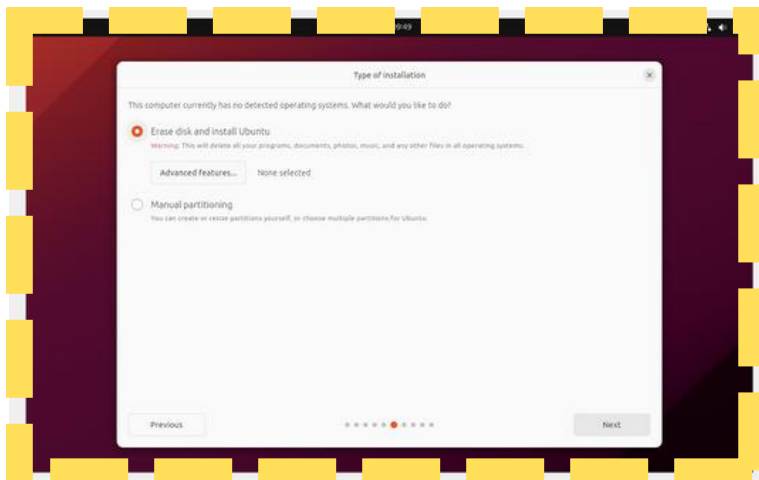
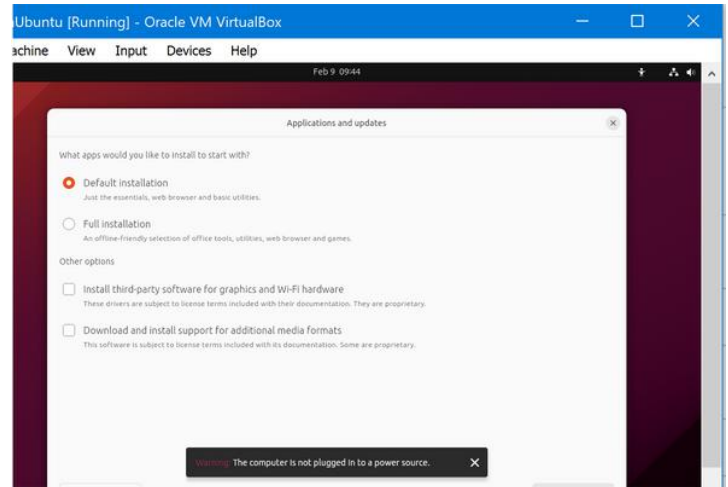
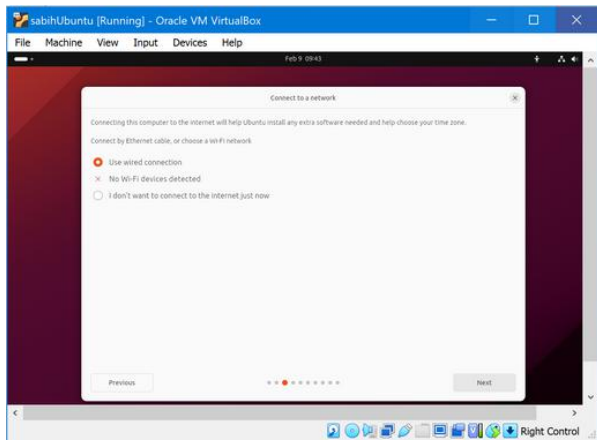
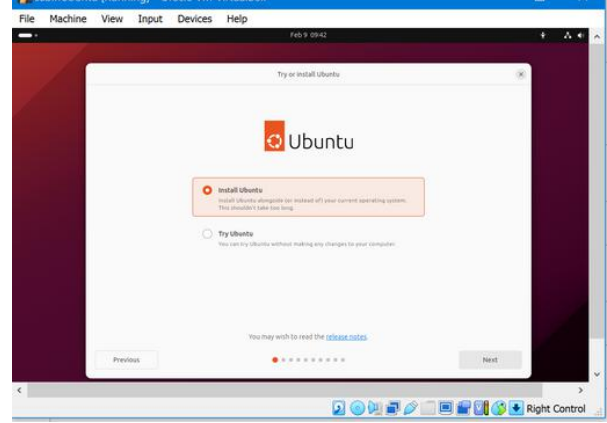
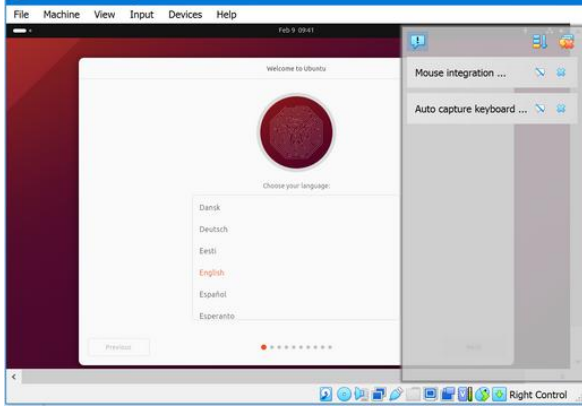


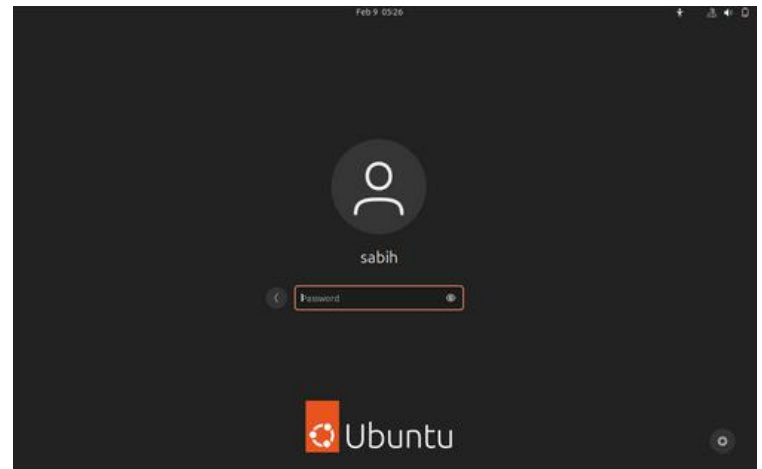
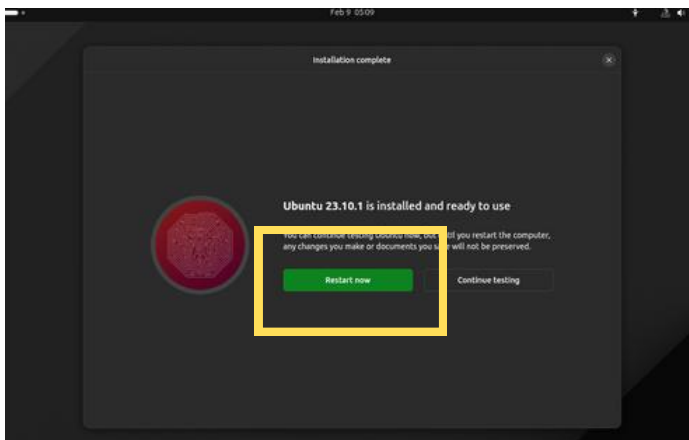
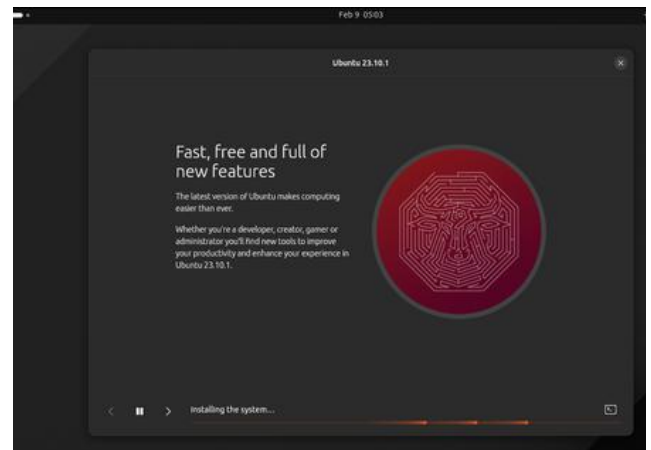
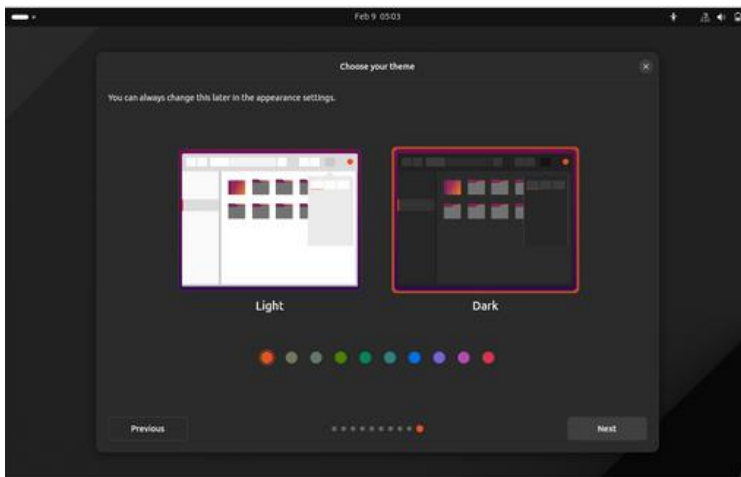
- سپس باید اندازه هارد دیسک را برای ماشین مجازی مشخص کنیم. حداقل 20-25 گیگابایت توصیه می شود.



- خلاصه ای از تنظیمات :

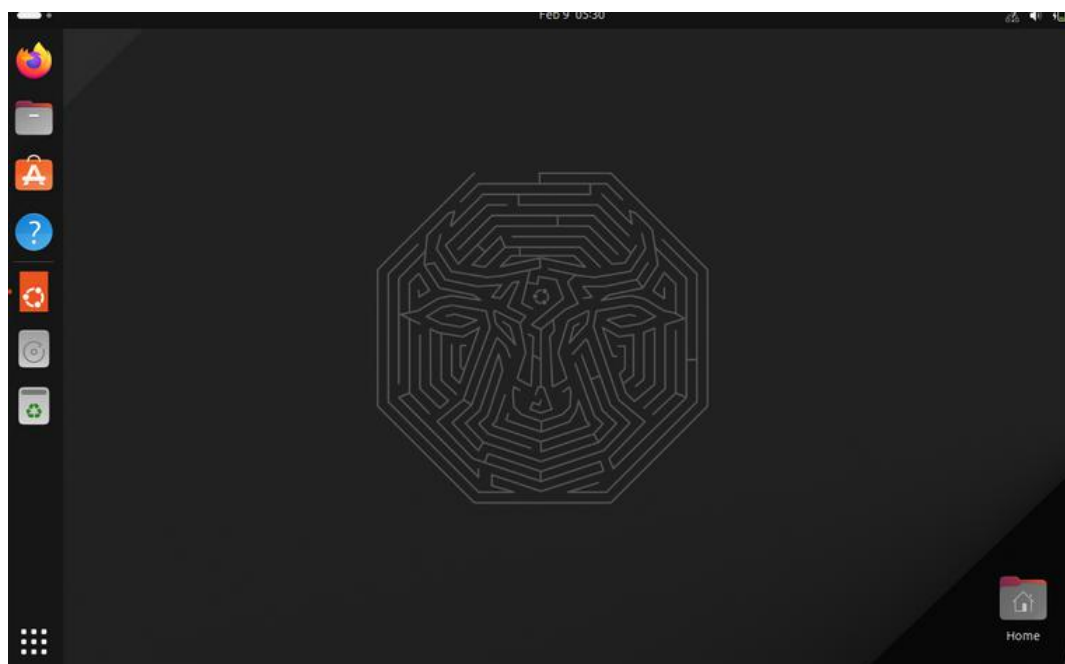






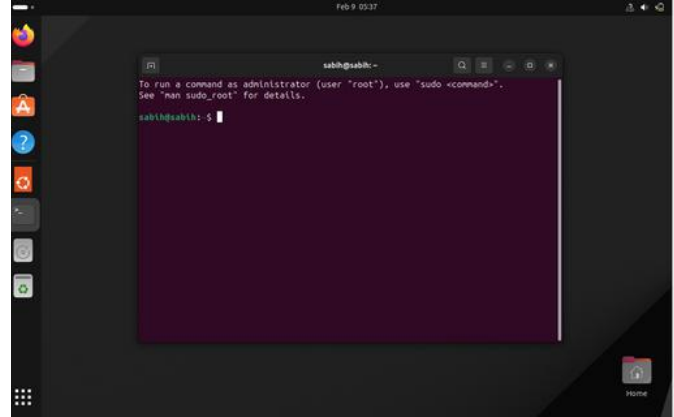
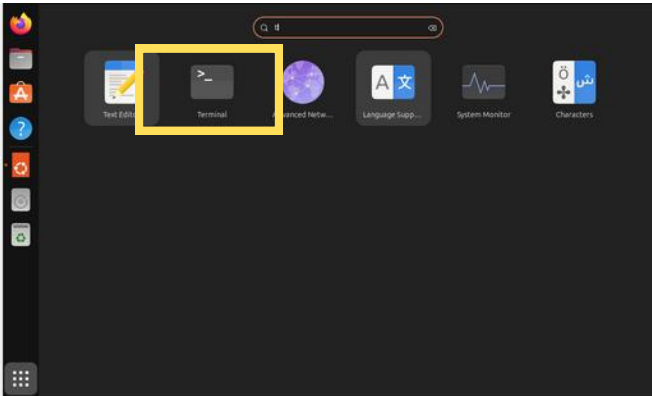
در رابطه با مرحله هایلایت شده:

با دو گزینه مواجه شده بودیم. با گزینه manual partitioning می‌توانیم حافظه‌ای که به ماشین‌مان اختصاص داده بودیم را پارتیشن‌بندی کنیم و گزینه Erase disk and install Ubuntu به سادگی حافظه را پاک می‌کند و اوپونتو را برای‌مان نصب می‌کند. از آنجایی که فرآیند نصب اوپونتو روی VirtualBox (ماشین مجازی) در حال اجراست، ما هیچ پارتیشن‌بندی روی هارد مجازی ایجاد شده نداریم! پس با انتخاب گزینه Erase disk and install Ubuntu تمامی پارتیشن‌های لازم به صورت خودکار ایجاد شده و اوپونتو نصب می‌شود.



ترمینال:

ترمینال ابزاریست که به ما اجازه می‌دهد دستورات مختلف را بر روی آن اجرا کنیم و از این طریق با هسته سیستمعامل (لینوکس) تراکنش داشته باشیم.

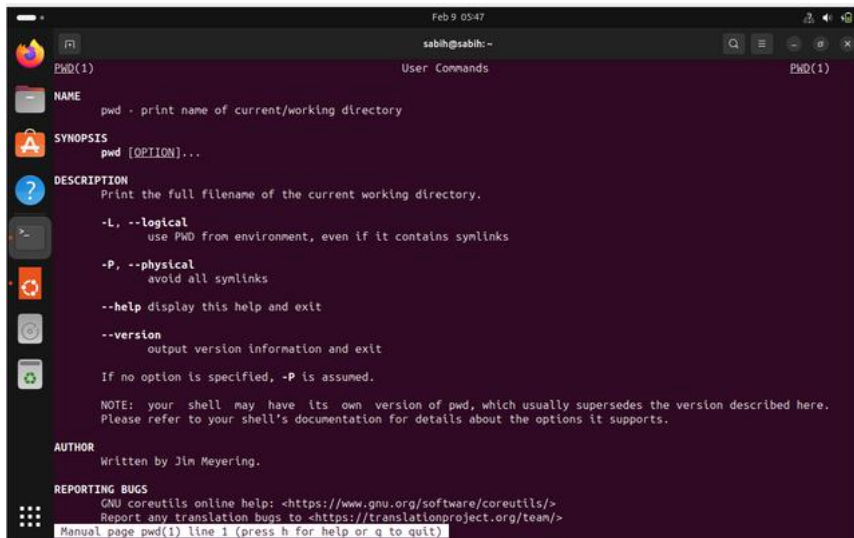
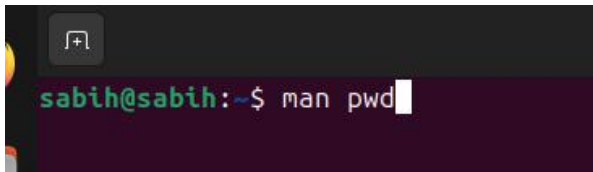


آشنایی با دستورات پایه ای لینوکس:

ساختار کلی دستورات در لینوکس: `COMMAND [OPTIONS] [ARGUMENTS]`

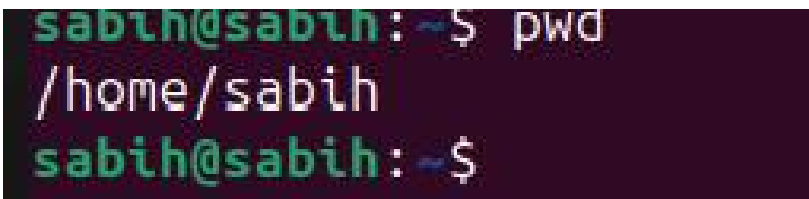
• استفاده از دستور man:

دستور man برای تمامی دستورات لینوکس دارای توضیحات کاملی است. برای مثال اگر از این طریق بخواهیم با دستور pwd آشنا شویم می‌توانیم از دستور زیر استفاده کنیم:



بنابراین متوجه شدیم که با کمک دستور (Print Working Directory) pwd، می‌توانیم دایرکتوری که اکنون با ترمینال درون آن قرار داریم را ببینیم.

• اجرای pwd:



دستور cd

با کمک دستور (Change Directory) cd، می‌توان میان دایرکتوری‌های مختلف جابه‌جا شد. به دو صورت می‌توان دایرکتوری را عوض کرد:

1. وارد کردن آدرس مقصد به صورت کامل.
 2. وارد کردن آدرس مقصد به صورت نسبی بسته به دایرکتوری که در آن قرار داریم.
- در زیر از هر دو روش مثالی آورده شده است:

(Absolute Path) آدرس کامل

در این نحوه آدرس‌دهی، به طور کامل به دایرکتوری که می‌خواهیم به درون آن برویم اشاره می‌کنیم. مثلاً فرض کنید می‌خواهیم با این نحوه آدرس‌دهی، به فولدر Desktop که در آدرس زیر قرار دارد، برویم:

```
home/<USERNAME>/Desktop/
```

کافیست دستور زیر را وارد کنیم:

```
cd /home/<USERNAME>/Desktop
```

سپس دایرکتوری ما به آدرس مورد نظرمان تغییر پیدا می‌کند.

(Relative Path) آدرس نسبی

در این نحوه آدرس‌دهی، با توجه به دایرکتوری که هم‌اکنون درون آن قرار داریم، به دایرکتوری که می‌خواهیم به درون آن برویم، اشاره می‌کنیم. مثلاً فرض کنید می‌خواهیم با این نحوه آدرس‌دهی، به فولدر Desktop خود که در آدرس زیر قرار دارد، برویم:

```
/home/<USERNAME>/Desktop
```

در ابتدا می‌دانیم که ما پس از باز کردن ترمینال درون دایرکتوری `home/<USERNAME>` قرار داریم، کافیست از این‌جایی که هستیم به درون دایرکتوری Desktop برویم. پس برای این‌کار دستور زیر را وارد کنیم :

```
cd Desktop/
```

در این نوع آدرس‌دهی برای رفتن به دایرکتوری پیشین، می‌توانیم از `..` استفاده کنیم.

```
cd ..
```

با دستور cd به دایرکتوری tmp رفته و پوشه ای به نام oslab1 می‌سازیم:

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~$ cd /tmp
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp$ mkdir oslab1
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp$ ls
oslab1
snap-private-tmp
systemd-private-d4957285905746f8a39b96bb89ec43dd-blutetooth.service-izFW6m
systemd-private-d4957285905746f8a39b96bb89ec43dd-bolt.service-KLPRdf
systemd-private-d4957285905746f8a39b96bb89ec43dd-colord.service-jvKKPu
systemd-private-d4957285905746f8a39b96bb89ec43dd-fwupd.service-ToAMse
systemd-private-d4957285905746f8a39b96bb89ec43dd-haveged.service-ot07yj
systemd-private-d4957285905746f8a39b96bb89ec43dd-ModemManager.service-t1fnpa
systemd-private-d4957285905746f8a39b96bb89ec43dd-power-profiles-daemon.service-zXyRX7
systemd-private-d4957285905746f8a39b96bb89ec43dd-switcheroo-control.service-yUnZ43
systemd-private-d4957285905746f8a39b96bb89ec43dd-systemd-logind.service-FGACx7
systemd-private-d4957285905746f8a39b96bb89ec43dd-systemd-oomd.service-fyz5h4
systemd-private-d4957285905746f8a39b96bb89ec43dd-systemd-resolved.service-XFjUrz
systemd-private-d4957285905746f8a39b96bb89ec43dd-systemd-timesyncd.service-tjrCG6
systemd-private-d4957285905746f8a39b96bb89ec43dd-upower.service-y6qqoL
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp$
```

دستورات داخل تصویر:

دستور ls

این دستور برای مشاهده محتویات درون دایرکتوری مورد نظر استفاده می‌شود. در صورت استفاده از این دستور بدون هیچ آپشن و یا آرگومانی، محتویات دایرکتوری فعلی شما را نمایش می‌دهد.

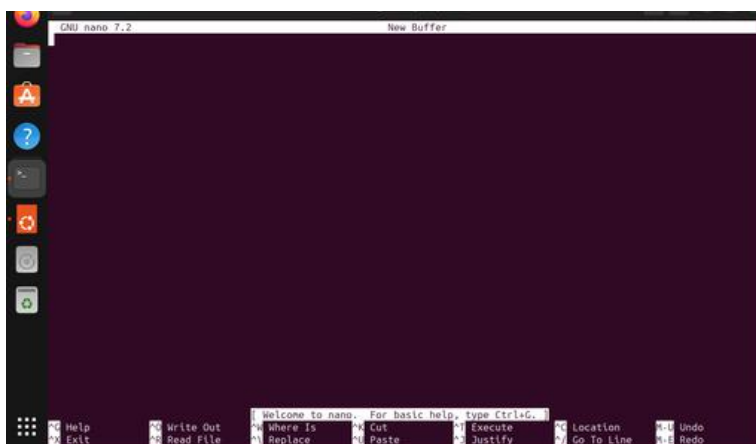
دستور mkdir

همانطور که از نام این دستور پیداست (Make Directory)، با کمک این دستور ما می‌توانیم یک دایرکتوری بسازیم.

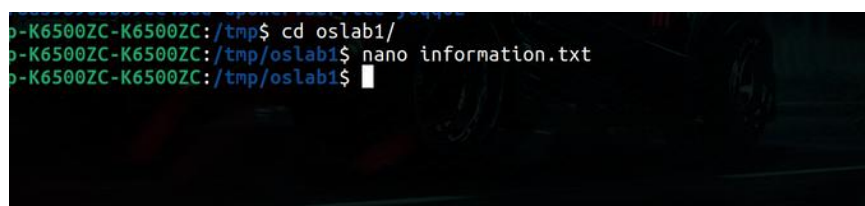
ویرایشگر Nano :

یک ویرایشگر متن ساده و قدرتمند در سیستم عامل های لینوکس است. این ویرایشگر از رابط کاربری خط فرمان (CLI) استفاده می کند و برای ویرایش فایل های متنی مورد استفاده قرار می گیرد.

با اجرای دستور nano:



به کمک ویرایشگر nano یک فایل متنی با محتوای نام و شماره‌ی دانشجویی خود به اسم `information.txt` ایجاد کنید و در نهایت از ویرایشگر خارج شوید.



خروج با `ctrl + x`

دستور cat

از این دستور در جهت مشاهده محتویات درون فایل ها استفاده می شود.



دستور cp

دستور (Copy) cp با گرفتن آدرس یک فایل و آدرس مقصد (آدرسی که ما می‌خواهیم فایل‌مان در آن کپی شود)، فایل را در آدرس مقصد کپی می‌کند.

دستور mv

دستور (Move) mv همانند دستور cp است با این تفاوت که به جای کپی کردن، خود فایل یا دایرکتوری را منتقل می‌کند (به بیانی دیگر کات و پیست می‌کند).

به کمک دستور mv نام فایل را به myinformation.txt تغییر دهید.

در واقع باید آدرس مقصد را نیز برابر با آدرس فعلی قرار داد تا فایل کات شود و با نام جدید paste شود.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ mv ./information.txt ./myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ ls
myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

به کمک دستور cp یک کپی از این فایل به اسم backupinfo.txt را در همان شاخه ایجاد کنید.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ cp ./myinformation.txt ./backupinfo.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ ls
backupinfo.txt  myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

دستورات زیر را اجرا کنید:

```
echo "Hello There!" > myinformation.txt
echo "Hello World!" >> myinformation.txt
```

تفاوت دو خط بالا را شرح دهید.

ریدایرکشن:

ساده‌ترین نوع ریدایرکشن، فرستادن خروجی یک دستور به یک فایل است. در bash می‌توان با کاراکتر > این کار را انجام داد. فرض کنید خروجی استاندارد دستور command را می‌خواهیم در فایل output بریزیم. فرمت کلی این نوع ریدایرکشن به صورت زیر است:

command > output

همچنین دستور echo نیز متن روبرویش را در ترمینال print می‌کند بنابراین دستور اول در دو دستور بالا خروجی echo در فایل myinfotmation قرار می‌گیرد.

```
backupinfo.txt  myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ echo "Hello There!"
Hello There!
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ echo "Hello There!" > myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ cat myinformation.txt
Hello There!
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

محتویات قبلی فایل پاک شد و خروجی دستور در آن نوشته شد.

اما اگر بخواهیم این اتفاق نیفتد یعنی به جای بازنویسی فایل، با هر بار اجرای یک دستور، خروجی آن دستور صرفاً به انتهای محتوای قبلی اضافه شود باید از <> استفاده کنیم. فرمت کلی نیز به صورت زیر خواهد بود:

command >> output

یعنی دستور دوم در دو دستور بالا کاری مشابه با append کردن انجام می‌دهد.

```
Hello There!
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ echo "Hello World!" >> myinformation.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ cat myinformation.txt
Hello There!
Hello World!
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

یک فایل متنی جدید با محتوای دلخواه را به کمک دستور cat بدون استفاده از nano به نام testfile.txt ایجاد کنید.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ cat > testfile.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

ctrl + D

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ cat testfile.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

لیست پردازش‌های در حال اجرا را به کمک دستور ps نمایش دهید.

دستور ps

ps مخفف Process Status و همان‌طور که از نامش پیداست، می‌توانیم از آن برای مشاهده اطلاعات پراسس‌ها استفاده کنیم. اگر این دستور را اجرا کنیم:

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ ps
  PID TTY          TIME CMD
 8847 pts/1    00:00:00 bash
 9202 pts/1    00:00:00 ps
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$
```

پراسس

به‌طور کلی، به اجرای هر برنامه یا دستور، پراسس گفته می‌شود، یعنی زمانی که هر دستور یا برنامه‌ای را اجرا می‌کنیم، لینوکس (حداقل) یک پراسس برای آن ایجاد می‌کند. هر پراسس از منابع سیستم مانند CPU، حافظه و ... استفاده می‌کند. در لینوکس هر پراسس با یک PID (Process Identification Number) مشخص می‌شود.

این دستور آپشن‌هایی دارد که با استفاده از آن‌ها می‌توانیم اطلاعات دقیق‌تری راجع به پراسس‌ها بدست آوریم. این آپشن‌ها عبارت است از:

- آپشن -a: این آپشن تمامی پراسس‌ها در حال اجرا مربوط به همه کاربران را نمایش می‌دهد.
- آپشن -u: این آپشن اطلاعات بیشتری مانند مقدار مصرف منابع سیستم توسط پراسس، مالک پراسس و ... را نمایش می‌دهد.
- آپشن -x: این آپشن پراسس‌هایی که توسط ترمینال اجرا نشده‌اند را نمایش می‌دهد.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/tmp/oslab1$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0 168284 13092 ?        Ss   06:54   0:03 /sbin/init splash
root         2  0.0  0.0     0     0 ?        S    06:54   0:00 [kthreadd]
root         3  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_gp]
root         4  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_par_gp]
root         5  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_tasks_rude_kthread]
root         6  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_tasks_trace_kthread]
root         7  0.0  0.0     0     0 ?        I<   06:54   0:00 [kworker/0:0H-events_highpri]
root        10  0.0  0.0     0     0 ?        I<   06:54   0:00 [mm_percpu_wq]
root        11  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_tasks_kthread]
root        12  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_tasks_rude_kthread]
root        13  0.0  0.0     0     0 ?        I<   06:54   0:00 [rcu_tasks_trace_kthread]
root        14  0.0  0.0     0     0 ?        S    06:54   0:00 [ksoftirqd/0]
root        15  0.0  0.0     0     0 ?        I<   06:54   0:25 [rcu_preempt]
root        16  0.0  0.0     0     0 ?        S    06:54   0:00 [migration/0]
root        17  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/0]
root        18  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/0]
root        19  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/1]
root        20  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/1]
root        21  0.0  0.0     0     0 ?        S    06:54   0:00 [migration/1]
root        22  0.0  0.0     0     0 ?        S    06:54   0:00 [ksoftirqd/1]
root        23  0.0  0.0     0     0 ?        I<   06:54   0:00 [kworker/1:0H-events_highpri]
root        24  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/2]
root        25  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/2]
root        26  0.0  0.0     0     0 ?        S    06:54   0:00 [migration/2]
root        27  0.0  0.0     0     0 ?        S    06:54   0:00 [ksoftirqd/2]
root        28  0.0  0.0     0     0 ?        I<   06:54   0:00 [kworker/2:0H-events_highpri]
root        29  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/3]
root        30  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/3]
root        31  0.0  0.0     0     0 ?        S    06:54   0:00 [migration/3]
root        32  0.0  0.0     0     0 ?        I<   06:54   0:00 [kworker/3:0H-events_highpri]
root        33  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/4]
root        34  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/4]
root        35  0.0  0.0     0     0 ?        S    06:54   0:00 [migration/4]
root        36  0.0  0.0     0     0 ?        I<   06:54   0:00 [kworker/4:0H-events_highpri]
root        37  0.0  0.0     0     0 ?        S    06:54   0:00 [cpuhp/5]
root        38  0.0  0.0     0     0 ?        S    06:54   0:00 [idle_inject/5]
```

نوع دیگری از ریدایرکشن نیز وجود دارد که در دنیای لینوکس پرکاربرد است. این ریدایرکشن که پایپ (Pipe) نام دارد، خروجی استاندارد یک دستور را به ورودی استاندارد دستور دیگر ارسال می‌کند. یعنی ریدایرکشن بین دستور و دستور به جای دستور و فایل! پایپ در فارسی به معنای لوله است و به درستی هم این اسم را روی این نوع ریدایرکشن گذاشته‌اند. در واقع با کمک لوله یا پایپ می‌توان دستورات مختلف لینوکس را مانند قطعات لگو به یکدیگر متصل کرد و کارهای متفاوتی انجام داد.

command1 | command2 | command3 | ... | commandn

به کمک دستور **grep** لیست پردازش‌هایی را نشان دهید که در خط مربوط به آن‌ها (نام پردازش، نام کاربر و...) حرف **a** وجود دارد. دستور **grep**

یکی از دستورات بسیار پرکاربرد در لینوکس دستور **grep** است. این دستور برای گشتن یک عبارت یا الگو در یک فایل استفاده می‌شود. این عبارت یا الگو در قالب عبارت منظم (Regular Expression or Regex) به **grep** داده می‌شود.

grep string file

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~$ ps aux | grep 'a'
root      1  0.0  0.0 168284 13092 ?        Ss   06:55   0:02 /sbin/init splash
root      2  0.0  0.0      0   0 ?        S    06:55   0:00 [kthreadd]
root      4  0.0  0.0      0   0 ?        I<   06:55   0:00 [rcu_par_gp]
root     11  0.0  0.0      0   0 ?        I    06:55   0:00 [rcu_tasks_kthread]
root     12  0.0  0.0      0   0 ?        I    06:55   0:00 [rcu_tasks_rude_kthread]
root     13  0.0  0.0      0   0 ?        I    06:55   0:00 [rcu_tasks_trace_kthread]
root     16  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/0]
root     22  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/1]
root     28  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/2]
root     34  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/3]
root     40  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/4]
root     46  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/5]
root     52  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/6]
root     58  0.0  0.0      0   0 ?        S    06:55   0:01 [migration/7]
root     64  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/8]
root     70  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/9]
root     76  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/10]
root     82  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/11]
root     88  0.0  0.0      0   0 ?        S    06:55   0:01 [migration/12]
root     94  0.0  0.0      0   0 ?        S    06:55   0:01 [migration/13]
root    100  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/14]
root    106  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/15]
root    112  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/16]
root    118  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/17]
root    124  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/18]
root    130  0.0  0.0      0   0 ?        S    06:55   0:00 [migration/19]
root    135  0.0  0.0      0   0 ?        I<   06:55   0:00 [lnet_frag_wq]
root    136  0.0  0.0      0   0 ?        S    06:55   0:00 [kauditd]
root    138  0.0  0.0      0   0 ?        I    06:55   0:00 [kworker/1:1-rcu_par_gp]
root    140  0.0  0.0      0   0 ?        S    06:55   0:00 [khungtdskd]
root    141  0.0  0.0      0   0 ?        S    06:55   0:00 [oom_reaper]
root    143  0.0  0.0      0   0 ?        I<   06:55   0:00 [writeback]
root    145  0.0  0.0      0   0 ?        S    06:55   0:00 [kcompactd0]
root    147  0.0  0.0      0   0 ?        SN   06:55   0:00 [khugepaged]
root    153  0.0  0.0      0   0 ?        I    06:55   0:00 [kworker/3:1-rcu_par_gp]
root    170  0.0  0.0      0   0 ?        I<   06:55   0:00 [ata_sff]
root    172  0.0  0.0      0   0 ?        I<   06:55   0:00 [edac-poller]
root    174  0.0  0.0      0   0 ?        S    06:55   0:00 [watchdogd]
root    176  0.0  0.0      0   0 ?        S    06:55   0:00 [ksm]d0]
```

خروجی **ps** با **pipe** به عنوان ورودی **grep** استفاده شده است.

به کمک دستور **cd** به داخل شاخه‌ی **usr/bin/** رفته و به کمک دستور **ls** لیست فایل‌های موجود در آن را نمایش دهید. فایل‌های موجود در این پوشه بخشی از دستورات قابل اجرا در سیستم هستند. (توضیحات **ls** در بالا گفته شده است.)

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~$ cd /usr/bin
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/bin$ ls
['
4iitopppn      gdnigrator      nsgattrib       resizecons
7z             gdtppn          nsgcat          resizepart
7za            glntall-info    nsgcmp         resolvectl
7zr            gio             nsgconv        resolvelp
aa-enabled     gipddcode       nsgen          rev
aa-exec        git             nsgeexec       rfcomm
aa-features-abi gitdiff          nsgrfilter     rgb3topppn
accountwizard  gitdiffview     nsgrng         rgrep
aconect        git-receive-pack nsgrng         rhythmbox
acpi_listen   git-shell        nsgrng         rhythmbox-client
add-apt-repository git-upload-archive nsgrng         rletopppn
addpare        git-upload-pack  nsgrng         rlogin
addr2line      gjs             nsgrng         rmdir
afrscan-discover gjs-console     nt             rnano
akonadi_agent_launcher gkb-d-keyboard-display nt-gnu         routef
akonadi_agent_server glib-compile-schemas ntr            routel
akonadi_akonotes_resource glxdemo         ntraces       rpgen
akonadi_archivemail_agent glxinfo         ntr-packet     rsync
akonadi_birthdays_resource glxgears        ntrtopppn      rsh
akonadi_contacts_resource glxgears.x86_64-linux-gnu nuon           rstart
akonadi_control glxheads        nv             rstartd
akonadi_ctl    glxheads.x86_64-linux-gnu ny_print_defaults rsync
akonadi_davgroupware_resource glxinfo         nysql          rsync-ssl
akonadi_ewsnta_resource glxinfo.x86_64-linux-gnu nysql_install_db rtstat
akonadi_ews_resource gmake           nysql_upgrade  runcom
akonadi_followupreminder_agent gmenudbusnexusproxy namei          run-nalicap
akonadi_google_resource gnone-calculator nano           run-parts
akonadi_lcaldir_resource gnone-calendar  nautilus       run-with-aspell
akonadi_lcal_resource gnone-characters nautilus-autorun-software rvim
akonadi_lmap_resource gnone-control-center nautilus-sendto rvm
akonadi_indexing_agent gnone-disk-image-mounter nawk          rygel
akonadi_kalarn_dir_resource gnone-disks     nc            sane-find-scanner
akonadi_kalarn_resource gnone-extensions ncspenbsd     sane-log
akonadi_kolab_resource gnone-extensions-app neotoppn      sbattach
akonadi_nalldir_resource gnone-font-viewer neqn          sbttopppn
```


به کمک دستور ls و استفاده از پارامترهای مناسب، علاوه بر نام فایل‌ها، حجم آن‌ها را نمایش دهید.

آپشن -l

در صورت استفاده از دستور ls با این آپشن، نتایج دقیق‌تر نمایش داده می‌شوند. در این نوع نمایش جزئیاتی از قبیل یوزر سازنده فایل‌ها، دسترسی و حجم فایل‌ها، زمان آخرین تغییرات روی فایل‌ها و ... نمایش داده می‌شود:

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/usr/bin$ ls -l
total 336044
-rwxr-xr-x 1 root root 51632 Feb 7 2022 '['
-rwxr-xr-x 1 root root 18456 Feb 7 2021 411toppm
-rwxr-xr-x 1 root root 39 Aug 15 2020 7z
-rwxr-xr-x 1 root root 40 Aug 15 2020 7za
-rwxr-xr-x 1 root root 40 Aug 15 2020 7zr
-rwxr-xr-x 1 root root 35344 Jun 5 2023 aa-enabled
-rwxr-xr-x 1 root root 35344 Jun 5 2023 aa-exec
-rwxr-xr-x 1 root root 31248 Jun 5 2023 aa-features-abi
-rwxr-xr-x 1 root root 876800 Mar 3 2022 accountwizard
-rwxr-xr-x 1 root root 22912 Jan 12 2022 aconnect
-rwxr-xr-x 1 root root 19016 Jan 25 2022 acpi_listen
-rwxr-xr-x 1 root root 14478 Aug 10 2023 add-apt-repository
-rwxr-xr-x 1 root root 14712 Feb 20 2022 addpart
lrwxrwxrwx 1 root root 26 Jan 3 06:16 addr2line -> x86_64-linux-gnu-addr2line
-rwxr-xr-x 1 root root 150376 Mar 25 2022 aircan-discover
-rwxr-xr-x 1 root root 22848 Mar 17 2022 akonadi_agent_launcher
-rwxr-xr-x 1 root root 92560 Mar 17 2022 akonadi_agent_server
-rwxr-xr-x 1 root root 149912 Mar 3 2022 akonadi_akonotes_resource
-rwxr-xr-x 1 root root 100744 Mar 3 2022 akonadi_archivemail_agent
-rwxr-xr-x 1 root root 104848 Mar 3 2022 akonadi_birthdays_resource
-rwxr-xr-x 1 root root 104848 Mar 3 2022 akonadi_contacts_resource
-rwxr-xr-x 1 root root 334232 Mar 17 2022 akonadi_control
-rwxr-xr-x 1 root root 104840 Mar 17 2022 akonadi_ctl
-rwxr-xr-x 1 root root 432528 Mar 3 2022 akonadi_davgroupware_resource
-rwxr-xr-x 1 root root 80272 Mar 3 2022 akonadiewsmta_resource
```

در سطر اول خروجی که در تصویر مشاهده می‌کنید، مجموع حجم بلاک‌های حافظه برای فایل‌هاست.

در ادامه جدولی می‌بینیم که ستون اول آن، مربوط به دسترسی فایل‌ها و دایرکتوری‌هاست.

ستون دوم تعداد هاردلینک‌ها به این فایل را به ما نشان می‌دهد.

ستون سوم و چهارم به ترتیب نام یوزر و نام گروه آن یوزری که فایل یا دایرکتوری را ساخته نمایش می‌دهد.

ستون پنجم حجم فایل‌ها را به بایت نمایش می‌دهد.

ستون ششم و هفتم و هشتم به ترتیب ماه و روز و ساعت آخرین تغییرات روی فایل‌ها را نمایش می‌دهد.

در نهایت ستون نهم نام فایل را به ما نمایش می‌دهد.

فهمیدن حجم فایل از روی ستون پنجم کار سختی‌ست و ممکن است خطایی در تبدیل آن به واحدهای قابل فهمی مانند کیلوبایت و مگابایت و گیگابایت و ... داشته باشیم. برای فهمیدن راحت‌تر حجم فایل‌ها و دایرکتوری‌ها نیاز به آپشن -lh داریم.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/usr/bin$ ls -lh
total 329M
-rwxr-xr-x 1 root root 51K Feb 7 2022 '['
-rwxr-xr-x 1 root root 19K Feb 7 2021 411toppm
-rwxr-xr-x 1 root root 39 Aug 15 2020 7z
-rwxr-xr-x 1 root root 40 Aug 15 2020 7za
-rwxr-xr-x 1 root root 40 Aug 15 2020 7zr
-rwxr-xr-x 1 root root 35K Jun 5 2023 aa-enabled
-rwxr-xr-x 1 root root 35K Jun 5 2023 aa-exec
-rwxr-xr-x 1 root root 31K Jun 5 2023 aa-features-abi
-rwxr-xr-x 1 root root 851K Mar 3 2022 accountwizard
-rwxr-xr-x 1 root root 23K Jan 12 2022 aconnect
-rwxr-xr-x 1 root root 19K Jan 25 2022 acpi_listen
-rwxr-xr-x 1 root root 15K Aug 10 2023 add-apt-repository
-rwxr-xr-x 1 root root 15K Feb 20 2022 addpart
lrwxrwxrwx 1 root root 26 Jan 3 06:16 addr2line -> x86_64-linux-gnu-addr2line
-rwxr-xr-x 1 root root 147K Mar 25 2022 aircan-discover
-rwxr-xr-x 1 root root 23K Mar 17 2022 akonadi_agent_launcher
-rwxr-xr-x 1 root root 91K Mar 17 2022 akonadi_agent_server
-rwxr-xr-x 1 root root 147K Mar 3 2022 akonadi_akonotes_resource
-rwxr-xr-x 1 root root 99K Mar 3 2022 akonadi_archivemail_agent
-rwxr-xr-x 1 root root 103K Mar 3 2022 akonadi_birthdays_resource
-rwxr-xr-x 1 root root 103K Mar 3 2022 akonadi_contacts_resource
-rwxr-xr-x 1 root root 327K Mar 17 2022 akonadi_control
-rwxr-xr-x 1 root root 103K Mar 17 2022 akonadi_ctl
-rwxr-xr-x 1 root root 423K Mar 3 2022 akonadi_davgroupware_resource
-rwxr-xr-x 1 root root 79K Mar 3 2022 akonadiewsmta_resource
```

به کمک دستور grep لیست فایل‌های در این پوشه را نشان دهید که در آن‌ها کلمه fs یا ld وجود دارد.

آپشن -E برای استفاده از رجکس است. خروجی ls با استفاده از pipe به عنوان ورودی grep استفاده شده است.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/usr/bin$ ls | grep -E 'fs|ld'
akonadi_icaldir_resource
akonadi_maildir_resource
akonadi_maildispatcher_agent
akonadi_mixedmaildir_resource
dehtmldiff
diffstat
dpkg-buildflags
dpkg-buildpackage
dpkg-checkbuilddeps
dpkg-genbuildinfo
fold
fstopgm
fsview
gnome-shell-extension-prefs
gold
gouldtoppm
grub-fstest
grub-ntldr-img
gtk4-builder-tool
gtk-builder-tool
hbpldecode
kbuildsysco5
ld
ld.bfd
ldd
```

نوشتن یک برنامه به زبان C

(برای فراخوانی دستورهای پوسته در یک برنامه به زبان سی میتوان از تابع `system()` بهره برد که در کتابخانه ی `stdlib.h` است.)

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1
GNU nano 6.2 test.c *
#include <stdio.h>
#include <stdlib.h>

void main(){
    system("ls");
}
```

compile کردن برنامه:

لیست compiler های نصب شده روی سیستم:

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$ dpkg --get-architecture | grep compiler
ii g++ 4:11.2.0-1ubuntu1 amd64 GNU C++ compiler
ii g++-11 11.4.0-1ubuntu1~22.04 amd64 GNU C++ compiler
ii gcc 4:11.2.0-1ubuntu1 amd64 GNU C compiler
ii gcc-11 11.4.0-1ubuntu1~22.04 amd64 GNU C compiler
ii libllvm13:amd64 1:13.0.1-2ubuntu2.2 amd64 Modular compiler and toolchain technologies, runtime library
ii libllvm15:amd64 1:15.0.7-0ubuntu0.22.04.3 amd64 Modular compiler and toolchain technologies, runtime library
ii libxkbcommon0:amd64 1.4.0-1 amd64 library interface to the XKB compiler - shared
ii rpcsvc-proto 1.4.2-0ubuntu6 amd64 RPC protocol compiler and definitions
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$
```

مشاهده می شود که GCC بر روی سیستم نصب می باشد و می توان برنامه های به زبان c را compile کرد:

`gcc -o output_filename source_code.c`

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$ ls
backupinfo.txt myinformation.txt test.c testfile.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$ gcc -o test test.c
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$ ls
backupinfo.txt myinformation.txt test test.c testfile.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$ ./test
backupinfo.txt myinformation.txt test test.c testfile.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: /tmp/oslab1$
```

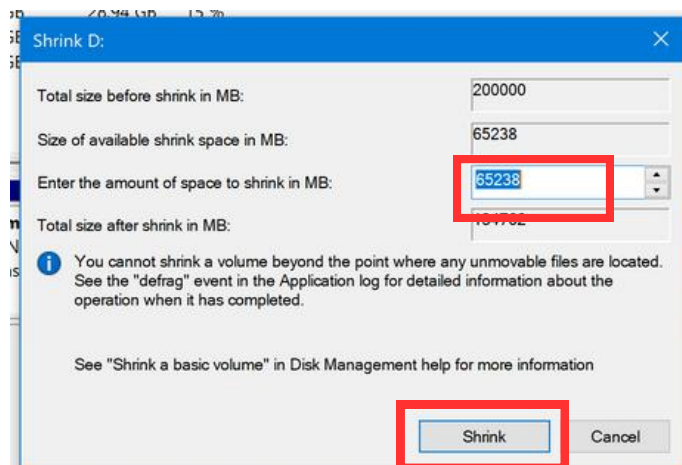
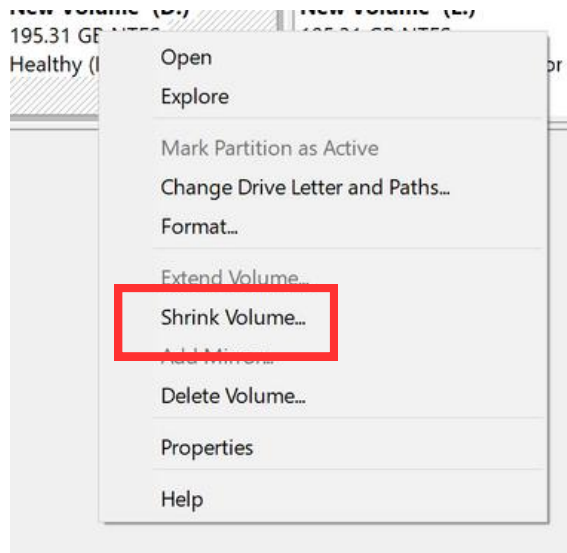
با اجرای `gcc -o test test.c` یک فایل اجرایی به نام `test` ایجاد شده و بعد با `./test` اجرا شده است که مطابق کد بالا دستور `ls` را اجرا کرده است.

پارتیشن چیست؟

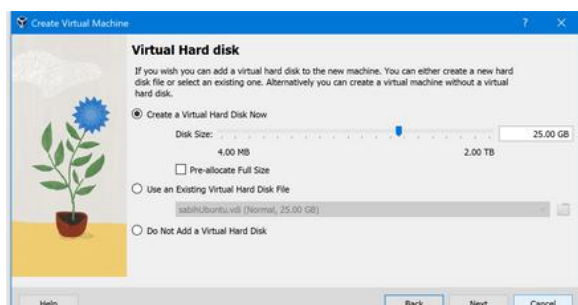
Disk Management									
File Action View Help									
Volume Layout Type File System Status Capacity Free Sp. % Free									
(C:)	Simple	Basic	NTFS	Healthy (B...)	194.66 GB	83.94 GB	43 %		
(Disk 0 partition 1)	Simple	Basic		Healthy (E...)	100 MB	100 MB	100 %		
(Disk 0 partition 4)	Simple	Basic		Healthy (F...)	547 MB	547 MB	100 %		
(Disk 0 partition 9)	Simple	Basic		Healthy (P...)	977 MB	977 MB	100 %		
(Disk 0 partition 10)	Simple	Basic		Healthy (P...)	18.63 GB	18.63 GB	100 %		
(Disk 0 partition 11)	Simple	Basic		Healthy (P...)	49.73 GB	49.73 GB	100 %		
New Volume (D:)	Simple	Basic	NTFS	Healthy (B...)	195.31 GB	63.76 GB	33 %		
New Volume (E:)	Simple	Basic	NTFS	Healthy (B...)	195.31 GB	28.94 GB	15 %		
New Volume (G:)	Simple	Basic	NTFS	Healthy (B...)	195.31 GB	84.49 GB	43 %		
New Volume (H:)	Simple	Basic	NTFS	Healthy (B...)	103.30 GB	35.63 GB	34 %		
Disk 0									
Basic	10	194.66 GB	547	New Volum	New Volum	New Volum	New Volum		
953.85 GB	He	Healthy (B)	Heal	Healthy (Ba	Healthy (Ba	Healthy (Ba	Healthy (B	Healthy (B	Healthy (P
Online									
Unallocated Primary partition									

در disk management می توان لیست دیسک ها و پارتیشن ها را مشاهده کرد.

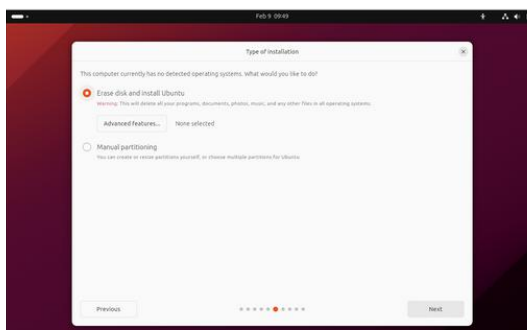
برای ساخت پارتیشن جدید از یک دیسک باید روی دیسک مورد نظر کلیک راست کرده و سپس روی گزینه shrink کلیک کنیم و حجم مورد نیاز برای پارتیشن جدید را مشخص کنیم و سپس پس از تایید این پارتیشن به صورت free space خواهد بود.



در بحث نصب به صورت دوال بوت معمولا از گزینه erase disk استفاده نمی کنیم و قبل از نصب، پارتیشنی برای نصب ابونتو روی آن تعیین می کنیم(مانند بالا).

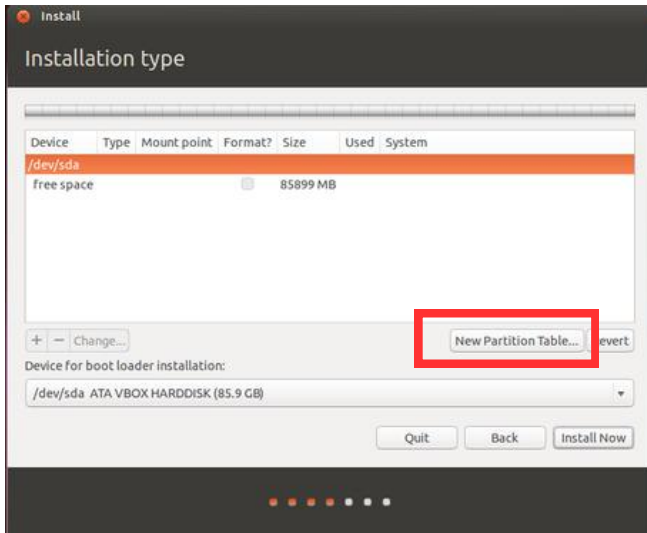


در مراحل نصب ما ابونتو را روی یک هارد مجازی نصب کردیم:



اما در بخش هایلایت شده از مراحل نصب گزینه پارتیشن بندی را انتخاب نکردیم:

اما اگر گزینه دوم را انتخاب می کردیم با چنین صفحه ای مواجه میشدیم:



پارتیشن /boot - این پارتیشن شامل فایل های لازم برای راه اندازی سیستم است. از فایل سیستم ext2 یا ext4 استفاده می شود. حداقل 100 مگابایت فضا نیاز است.

پارتیشن / - این پارتیشن ریشه فایل سیستم است و شامل فایل ها و دایرکتوری های مهم سیستم و برنامه ها می شود. از فایل سیستم ext3 یا ext4 استفاده می شود. حداقل 10 گیگابایت فضا پیشنهاد می شود. (بالاترین دایرکتوری ممکن در ساختار فایل لینوکس است و تمامی فایل ها و سایر دایرکتوری های کل سیستم عامل تان درون این دایرکتوری قرار دارند.)

پارتیشن /home - این پارتیشن شامل فایل های شخصی کاربران است. از فایل سیستم ext3 یا ext4 استفاده می شود. اندازه آن بستگی به نیاز کاربران دارد.

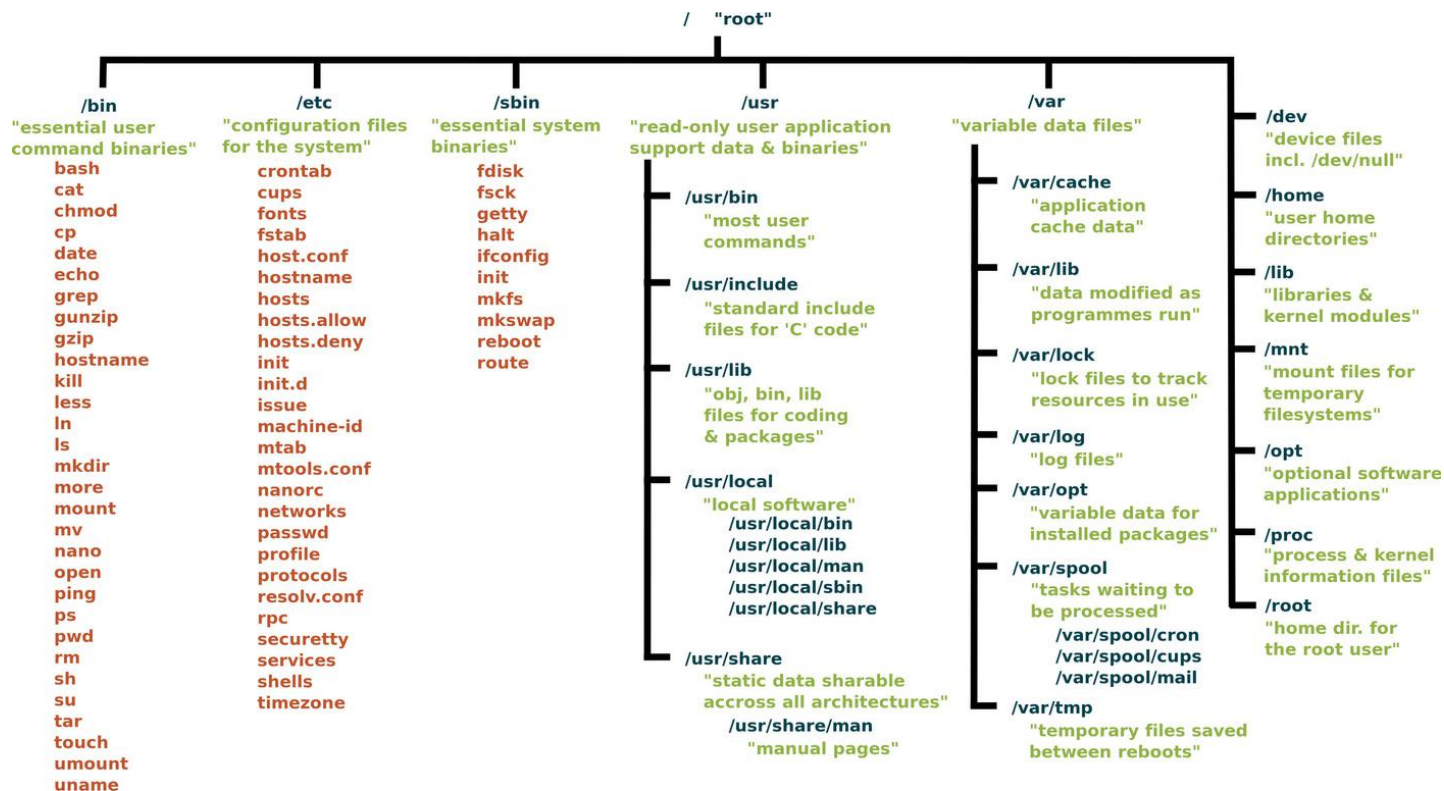
پارتیشن /var - این پارتیشن شامل داده های متغیر مانند لاگ ها است. از فایل سیستم ext3 یا ext4 استفاده می شود. حداقل 2 گیگابایت فضا پیشنهاد می شود.

پارتیشن /tmp - این پارتیشن شامل فایل های موقت است. از فایل سیستم ext3 یا ext4 استفاده می شود. حداقل 500 مگابایت فضا پیشنهاد می شود.

پارتیشن /usr - این پارتیشن شامل برنامه های کاربری است. از فایل سیستم ext3 یا ext4 استفاده می شود. حداقل 5 گیگابایت فضا پیشنهاد می شود.

پارتیشن /opt - این پارتیشن معمولاً برای نصب بسته های نرم افزاری اضافی استفاده می شود. از فایل سیستم ext3 یا ext4 استفاده می شود و اندازه آن بستگی به نیاز دارد.

در مجموع پارتیشن بندی باعث مدیریت بهتر فضای دیسک، افزایش امنیت و بهبود عملکرد می شود.



چندتا از انواع فایل سیستم مورد استفاده در پارتیشن بندی لینوکس:

ext2 - یک فایل سیستم قدیمی در لینوکس است که ویژگی های پایه ای مانند ساختار دایرکتوری و مدیریت فضا را فراهم می کند.

ext3 - نسخه بهبود یافته ext2 است.

ext4 - آخرین نسخه از خانواده ext است که ویژگی های پیشرفته تری مانند حداکثر اندازه بزرگ فایل و سرعت بهتر را ارائه می دهد.

XFS - فایل سیستمی با کارایی بالا برای سیستم های فایل بزرگ. سرعت خوبی در مدیریت فایل های بزرگ دارد.

Btrfs - فایل سیستمی نوین که ویژگی های پیشرفته ای مانند snapshots، فشردن سازی و دارد.

کاربرد دستورات زیر به اختصار:

- cut
- find
- head
- tail
- touch
- wc
- kill

دستور cut

این دستور به خودی خود هیچ کاربردی ندارد ولی استفاده آن در کنار آپشن های متنوعش برای مان فوق العاده کاربردی خواهد بود. با کمک آپشن های متنوع این دستور می توانیم خطوط موردنظرمان را براساس یک کاراکتر خاص دسته بندی کنیم و یا بر اساس تعداد بایت متن را بپریم! مثال:

معرفی آپشن -b

این آپشن برای به دست آوردن تعداد مشخصی بایت از هر سطر است. (هر کاراکتر معادل یک بایت است، اسپیس نیز یک بایت به حساب می آید). برای مثال، فرض کنید می خواهیم سه کاراکتر اول هر سطر را ببینیم. سه کاراکتر اول یعنی نیاز به دیدن سه بایت اول از هر سطر داریم. می توانیم مانند زیر با کاما (,) بایت های مورد نظرم را مشخص کنیم:

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: ~/Desktop
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ touch hi.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ cat hi.txt
hi
bye
sala
hello
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ cut -b 1,2,3 hi.txt
hi
bye
sal
hel
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC: ~/Desktop
With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

-b, --bytes=LIST
    select only these bytes

-c, --characters=LIST
    select only these characters

-d, --delimiter=DELIM
    use DELIM instead of TAB for field delimiter

-f, --fields=LIST
    select only these fields; also print any line that contains no
    delimiter character, unless the -s option is specified

-n
    (ignored)

--complement
    complement the set of selected bytes, characters or fields

Manual page cut(1) line 12 (press h for help or q to quit)
```

دستور find

این دستور به ما کمک می کند تا به راحتی دنبال فایل ها و یا دایرکتوری های مورد نظرم بگردیم و نتیجه جست و جو را با توجه به نیازمان شخصی سازی کنیم و یا حتی روی هر نتیجه تغییراتی را اعمال کنیم. ساختار کلی دستور، به این صورت است:

find [where to start searching from][expression determines what to find] [-options] [what to find]

مثال با آپشن -type و -name

برای یافتن دایرکتوری (d type) هایی با نام hi :

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ find . -type d -name "hi"
./hi
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ls
hi hi.c hi.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

دستور head

این دستور برای ما چند خط ابتدای محتوا را نشان می دهد (به طور پیش فرض ۱۰ خط اول فایل مورد نظرم را برایمان نشان خواهد داد).

مثال:


```
Open  hi.txt
~/Desktop
1 salam
2 bye
3 sabih
4 sara
5 farnaz
6 hi
7 hello hello hello
8 1234
9 5678
10 9999
11 bla bla bla
12 lorem?
13
```

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ head hi.txt
salam
bye
sabih
sara
farnaz
hi
hello hello hello
1234
5678
9999
```

آپشن `-n`
اگر نمی‌خواستیم ۱۰ خط اول را ببینیم و به جای آن قصد داشتیم تعداد خط دلخواهی از ابتدای فایل را ببینیم، اینجاست که این آپشن به کمک ما می‌آید. با استفاده از این آپشن می‌توانیم هر تعداد خطی که دلمان خواست را از ابتدای محتوای فایل ببینیم.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ head -n 11 hi.txt
salam
bye
sabih
sara
farnaz
hi
hello hello hello
1234
5678
9999
bla bla bla
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ head -n 3 hi.txt
salam
bye
sabih
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

دستور tail
این دستور برای ما چند خط آخر محتوا را نشان می‌دهد (پیش فرض ۱۰ خط).

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ tail hi.txt
sara
farnaz
hi
hello hello hello
1234
5678
9999
bla bla bla
lorem?
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ tail -n 3 hi.txt
bla bla bla
lorem?
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ tail -n 3 hi.txt
```

دستور touch
به کمک این دستور می‌توانیم فایل جدیدی را ایجاد کنیم.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ touch bye.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ls
bye.txt hi.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

این دستور یک فایل یا رشته را به عنوان ورودی گرفته و اطلاعاتی راجع به آن به ما می‌دهد. این اطلاعات مانند تعداد کاراکتر، کلمه و تعداد خط‌های آن فایل است.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ wc hi.txt
13 16 84 hi.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ wc bye.txt
0 0 0 bye.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

این اعداد به ترتیب از چپ به راست تعداد خط‌ها، تعداد لغات و تعداد کاراکترها می‌باشند.

آپشن -l

با آپشن -l تنها تعداد خط‌های فایل خروجی داده می‌شود.

آپشن -w

آپشن -w تنها تعداد کلمات فایل را خروجی می‌دهد.

معرفی آپشن -c

آپشن -c تعداد بایت‌های اشغالی فایل را خروجی می‌دهد.

سیگنال‌ها

به‌طور کلی سیگنال‌ها اعلانات یک‌طرفه‌ای هستند که با کمک آن‌ها می‌توان رویدادهایی را به پراسس‌ها اعلام کرد. سیگنال می‌تواند از کرنل به پراسس، از یک پراسس به پراسس دیگر و از یک پراسس به خودش ارسال شود. برای مثال هر بار که از میانبر Ctrl+C برای متوقف کردن یک پراسس استفاده می‌کنیم، درحال فرستادن یک سیگنال با نام SIGINT برای پراسس موردنظر هستیم.

کرنل لینوکس حدود ۳۰ سیگنال را پیاده‌سازی کرده که مهم‌ترین سیگنال‌ها به شرح زیر هستند:

- سیگنال SIGINT: عملیات پیشفرض این سیگنال این است که اجرا پراسس را خاتمه دهد، این سیگنال را می‌توان توسط کلید میانبر Ctrl+C به پراسس فرستاد.
- سیگنال SIGKILL: عملیات پیشفرض این سیگنال این است که اجرا پراسس را خاتمه دهد و فرق آن با SIGINT این است که سیگنال SIGKILL را نمی‌توان نادیده گرفت و یا رفتار آن را تغییر داد.
- سیگنال SIGSTOP: عملیات پیشفرض این سیگنال این است که اجرا پراسس را متوقف کند، این سیگنال را می‌توان توسط کلید میانبر Ctrl+Z به پراسس فرستاد.

لیست کامل سیگنال‌ها را می‌توان با استفاده از -l مشاهده کرد.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ kill -l
1) SIGHUP      2) SIGINT      3) SIGQUIT     4) SIGILL      5) SIGTRAP
6) SIGABRT     7) SIGBUS     8) SIGFPE     9) SIGKILL    10) SIGUSR1
11) SIGSEGV    12) SIGUSR2    13) SIGPIPE    14) SIGALRM    15) SIGTERM
16) SIGSTKFLT  17) SIGCHLD    18) SIGCONT    19) SIGSTOP    20) SIGTSTP
21) SIGTTIN    22) SIGTTOU    23) SIGURG     24) SIGXCPU    25) SIGXFSZ
26) SIGVTALRM  27) SIGPROF    28) SIGWINCH   29) SIGIO      30) SIGPWR
31) SIGSYS     34) SIGRTMIN   35) SIGRTMIN+1 36) SIGRTMIN+2 37) SIGRTMIN+3
38) SIGRTMIN+4 39) SIGRTMIN+5 40) SIGRTMIN+6 41) SIGRTMIN+7 42) SIGRTMIN+8
43) SIGRTMIN+9 44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5 60) SIGRTMAX-4 61) SIGRTMAX-3 62) SIGRTMAX-2
63) SIGRTMAX-1 64) SIGRTMAX
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

ما می‌توانیم با استفاده از دستور kill به پراسس‌ها طبق قالب زیر سیگنال ارسال کنیم:

kill -NAME <PID>

یا

kill -NUMBER <PID>

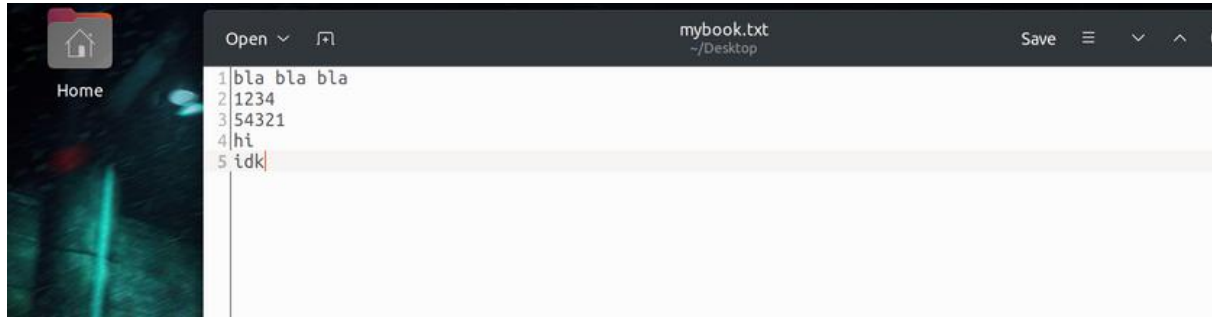
با کمک دستوراتی که فراگرفته اید، فرمان هایی برای اعمال زیر بنویسید:

0 پیدا کردن تعداد خطوط در یک فایل متنی به نام mybook.txt

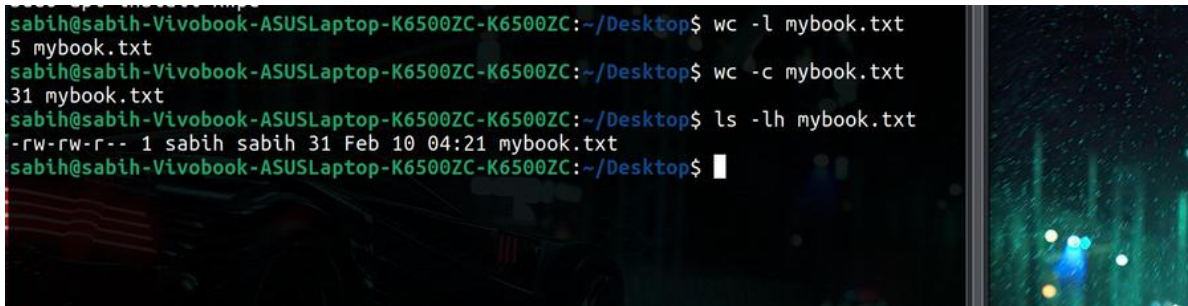
0 پیدا کردن تعداد فایل هایی که با حرف A شروع می شوند.

0 پیدا کردن حجم فایل mybook.txt

• با کمک نمونه ی برنامه داده شده در بخش قبل، برنامه ای به زبان C بنویسید که نشانی یک پوشه را از کاربر بگیرد و فهرست پرونده ها و پوشه های درون آن پوشه را نمایش دهد.

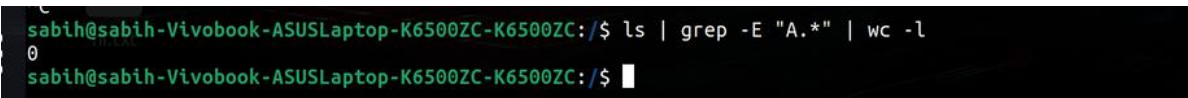


```
1 bla bla bla
2 1234
3 54321
4 hi
5 idk
```



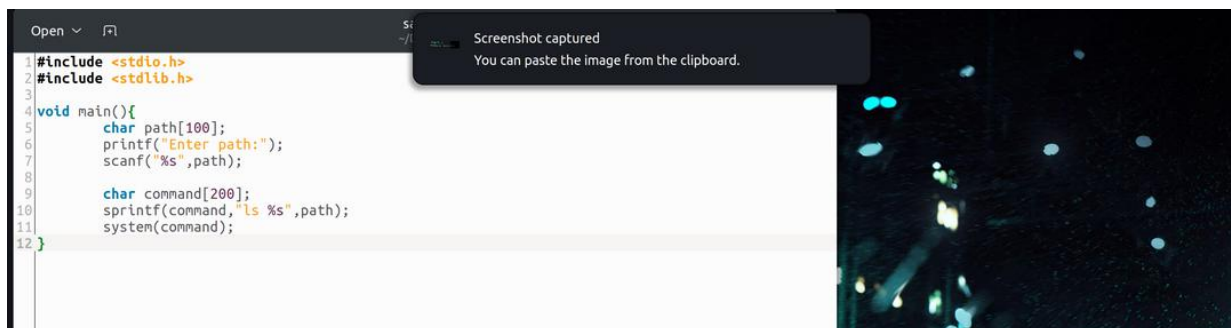
```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ wc -l mybook.txt
5 mybook.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ wc -c mybook.txt
31 mybook.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ls -lh mybook.txt
-rw-rw-r-- 1 sabih sabih 31 Feb 10 04:21 mybook.txt
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

پیدا کردن تعداد خطوط با `wc -l` و پیدا کردن حجم با `wc -c` یا `ls -lh`



```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/$ ls | grep -E "A.*" | wc -l
0
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/$
```

تعداد فایل هایی که با حرف A شروع می شوند:ارسال خروجی ls با پایپ به عنوان ورودی grep و ارسال خروجی آن(یعنی فایل هایی که با A شروع می شوند) به عنوان ورودی wc -l برای یافتن تعداد



```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 void main(){
5     char path[100];
6     printf("Enter path:");
7     scanf("%s",path);
8
9     char command[200];
10    sprintf(command,"ls %s",path);
11    system(command);
12 }
```

برنامه ای که یک ورودی از کاربر بگیرد و لیست فایل ها و دایرکتوری های آن ورودی(مسیر) را به کاربر نشان دهد.

```
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/$ cd ~
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:/$ cd Desktop/
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ gcc -o sabih sabih.c
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ls
bye.txt hi.txt mybook.txt sabih sabih.c
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ sabih
sabih: command not found
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ./sabih
Enter path:/
bin boot cdrom dev etc home lib lib32 lib64 libx32 lost+found media mnt opt proc root run sbin snap srv sys tmp usr var
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$ ./sabih
Enter path:~
Desktop Documents Downloads google-chrome-stable_current_amd64.deb Music Pictures Public snap Templates Videos
sabih@sabih-Vivobook-ASUSLaptop-K6500ZC-K6500ZC:~/Desktop$
```

- آشنایی با مفهوم ماشین مجازی
- آشنایی با لینوکس و توزیع های آن
- آشنایی با نحوه نصب یک توزیع لینوکس به صورت مجازی
- آشنایی با دستورات اولیه سیستم عامل لینوکس و کار با فایل
- کامپایل و اجرای کد در محیط لینوکس
- آشنایی با پارتیشن بندی و پارتیشن های مهم لینوکس
- ...

نتایج: