



## Trabajo Práctico Integrador

## Arquitectura y Sistemas Operativos

### Alumnos

Zampar Facundo, correo: facuzamparutn@gmail.com

Zampieri Pamela, correo: pamezampieri@gmail.com

**Tecnicatura Universitaria en Programación - Universidad Tecnológica Nacional.**

**Arquitectura y Sistemas Operativos**

### Docente Titular

Osvaldo Falabella

### Docente Tutor

Patricio Costello

## Índice

<b>Introducción.....</b>	<b>3</b>
<b>Marco teórico.....</b>	<b>4</b>
Caso Práctico 1 – Instalación de VirtualBox y creación de máquina virtual con Ubuntu Desktop 24.04.....	7
Caso Práctico 2 – Configuración de idioma y teclado en Ubuntu Desktop 24.04.....	9
Caso Práctico 3 – Problema y solución: Copiar y pegar entre host y máquina virtual.....	11
Caso Práctico 4 – Desarrollo en Python con Visual Studio Code.....	13
<b>Conclusiones.....</b>	<b>15</b>
<b>Bibliografía.....</b>	<b>16</b>
<b>Anexos.....</b>	<b>17</b>
Caso Práctico 1.....	17
Caso Práctico 2.....	21
Caso Práctico 3.....	24
Caso Práctico 4.....	26
Link video demostrativo del entorno y ejecución del código.....	34

## Introducción

La virtualización es una tecnología que permite crear una versión simulada (o virtual) de un recurso físico, como una computadora, un sistema operativo, un servidor, un almacenamiento o una red. A través de software especializado, se emulan entornos que actúan como si fueran físicos, permitiendo correr varios sistemas operativos o aplicaciones sobre una única máquina física.

Esta tecnología ha cobrado gran relevancia en los últimos años debido a su capacidad para optimizar recursos, reducir costos y facilitar entornos de prueba y desarrollo. En ámbitos educativos, profesionales y empresariales, la virtualización permite experimentar con distintos sistemas y configuraciones sin comprometer el entorno real.

En este trabajo se abordará el uso de VirtualBox, un hipervisor de tipo 2 ampliamente utilizado por su facilidad de uso y compatibilidad con múltiples sistemas operativos. A través de distintos casos prácticos, se mostrará cómo instalar y configurar una máquina virtual con Ubuntu, resolver problemas comunes de entorno, e implementar un entorno de desarrollo con herramientas como Python y Visual Studio Code.

## Marco teórico

La virtualización nace como una solución para optimizar el uso del hardware, reducir costos y mejorar la escalabilidad y seguridad. En lugar de tener múltiples computadoras físicas, se puede tener una sola máquina poderosa que ejecute varias máquinas virtuales (VMs), cada una con su propio sistema operativo y aplicaciones.

El componente fundamental de la virtualización es el hipervisor, un software que se encarga de gestionar las máquinas virtuales, permitiendo que comparten los recursos físicos de manera eficiente.

### Tipos de Virtualización

#### 1. Virtualización de Hardware

Simula un conjunto completo de hardware, permitiendo ejecutar sistemas operativos sin modificación. Ejemplo: VirtualBox, VMware Workstation.

#### 2. Virtualización de Software

Simula solo ciertas partes del sistema (como bibliotecas o entornos). Permite ejecutar programas en entornos distintos al original. Ejemplo: Wine o contenedores como Docker.

#### 3. Virtualización Completa

El sistema operativo invitado no necesita ser modificado, ya que el hipervisor simula todo el hardware. Esto ofrece compatibilidad total pero con algo más de sobrecarga. Ejemplo: VirtualBox, VMware, KVM.

#### 4. Paravirtualización

El sistema operativo huésped está modificado para ser consciente de que se ejecuta en un entorno virtual. Requiere menos recursos y tiene mejor rendimiento, pero menor compatibilidad.

Ejemplo: Xen, algunas configuraciones de KVM.

### Tipos de Hipervisores

Los hipervisores son el software que gestiona y ejecuta máquinas virtuales. Se dividen en dos tipos principales:

#### Hipervisor Tipo 1 (Bare Metal)

## TRABAJO PRÁCTICO INTEGRADOR

Se instala directamente sobre el hardware físico, sin necesidad de un sistema operativo anfitrión. Son más eficientes y se usan en servidores.

Ejemplos: VMware ESXi, Microsoft Hyper-V, Xen.

### Hipervisor Tipo 2 (Hosted)

Se ejecutan sobre un sistema operativo anfitrión. Son más fáciles de usar, ideales para uso personal o pruebas.

Ejemplos: VirtualBox, VMware Workstation, Parallels.

## Relevancia

La virtualización es una tecnología clave en el desarrollo y funcionamiento de la informática moderna. Su importancia radica en varios aspectos fundamentales:

- Optimización de recursos: Permite ejecutar múltiples sistemas operativos en un mismo hardware, aprovechando mejor la capacidad de procesamiento, memoria y almacenamiento.
- Reducción de costos: Disminuye la necesidad de comprar y mantener múltiples equipos físicos.
- Facilidad de pruebas y desarrollo: Los entornos virtuales son ideales para realizar pruebas sin riesgo para el sistema principal, lo que beneficia a desarrolladores, técnicos y estudiantes.
- Escalabilidad y flexibilidad: En entornos empresariales y de nube, facilita la implementación de sistemas según la demanda, con rápida adaptación a cambios.
- Mayor seguridad: Las máquinas virtuales pueden aislarse entre sí, reduciendo el riesgo de propagación de fallos o amenazas.
- Base de tecnologías modernas: Es la base de la computación en la nube, contenedores (como Docker) y DevOps.

## Herramienta utilizada: VirtualBox

Para llevar a cabo los casos prácticos del trabajo, se eligió utilizar **VirtualBox**, un hipervisor de tipo 2 desarrollado por Oracle. Es ampliamente utilizado en contextos educativos y de desarrollo debido a su facilidad de instalación, interfaz amigable, compatibilidad con diversos sistemas operativos (Windows, Linux, macOS) y soporte activo.

## TRABAJO PRÁCTICO INTEGRADOR

VirtualBox permite crear y gestionar máquinas virtuales de forma sencilla, lo que lo convierte en una herramienta ideal para aprender y aplicar conceptos de virtualización. En este trabajo se utilizará para instalar y ejecutar una máquina virtual con Ubuntu Desktop 24.04 LTS.

## Caso Práctico 1 – Instalación de VirtualBox y creación de máquina virtual con Ubuntu Desktop 24.04

Se utilizó **Oracle VirtualBox** para crear e instalar una máquina virtual con el sistema operativo **Ubuntu 24.04 Desktop LTS**, permitiendo así experimentar con Linux en un entorno aislado sin modificar el sistema operativo principal del equipo.

### Metodología

#### 1. Creación de la máquina virtual:

- Se abrió VirtualBox y se eligió la opción "Nueva".
- Se asignó el nombre UBUNTU-AYSO, con las siguientes especificaciones:
  - **Memoria RAM:** 4096 MB (4 GB)
  - **Procesadores:** 1 núcleo
  - **Disco duro virtual:** 40 GB, tipo VDI (almacenamiento dinámico)

#### 2. Montaje de imagen ISO y configuración básica:

- Se descargó la imagen ISO oficial desde el sitio de Ubuntu: [ubuntu-24.04.2-desktop-amd64.iso](https://releases.ubuntu.com/24.04/ubuntu-24.04.2-desktop-amd64.iso).
- Se montó la ISO como unidad de arranque.
- Se inició la máquina virtual y se siguió el asistente de instalación de Ubuntu, seleccionando una **instalación mínima**.
- Se completó la instalación con la configuración regional por defecto en inglés.

### Resultados Obtenidos

- La máquina virtual Ubuntu se instaló exitosamente y quedó lista para su uso.
- Se validó que el sistema operativo arranca correctamente desde VirtualBox.

## TRABAJO PRÁCTICO INTEGRADOR

- Se establecieron las bases para probar funcionalidades como configuración de red, instalación de software y uso del entorno Linux.
- Se adquirió experiencia con la herramienta de virtualización VirtualBox y con el proceso de instalación de un sistema operativo Linux.

## Caso Práctico 2 – Configuración de idioma y teclado en Ubuntu Desktop 24.04

Luego de instalar Ubuntu en la máquina virtual, se presentaron problemas con la distribución del teclado y el idioma del sistema, ya que por defecto estaba en inglés (US) y no reconocía correctamente símbolos como /, @, etc.

### Metodología:

#### 1. Abrir la Terminal:

Como el atajo Ctrl + Alt + T no funcionaba en la Mac virtualizada, se accedió desde el buscador del sistema usando el ícono de actividades y escribiendo “Terminal”.

#### 2. Cambiar el idioma del sistema:

- ◆ Se accedió a *Settings > Region & Language*
- ◆ Se agregó el idioma Español (Latin American) y se estableció como predeterminado.
- ◆ Se reinició la sesión para aplicar los cambios.

#### 3. Configurar el teclado correctamente:

- ◆ Desde la misma sección se eliminó el teclado “English (US)” y se agregó “Spanish Latin American”.
- ◆ Se probó la escritura de símbolos comunes como /, -, @, ñ, etc.
- ◆ Se seleccionó la opción “**Spanish Latin American**” entre todas las variantes disponibles.

#### 4. Verificación final:

- ◆ Se ejecutaron comandos en la terminal como:

```
sudo apt update
```

```
cd /home
```

Se confirmó que todos los caracteres necesarios podían escribirse sin problemas.

### Resultados Obtenidos

## TRABAJO PRÁCTICO INTEGRADOR

- Se soluciona el problema de la disposición del teclado.
- El sistema quedó completamente configurado en **idioma español**.
- La experiencia de uso mejoró al contar con un entorno familiar para el usuario.
- Se aprendió a identificar y ajustar configuraciones regionales en sistemas Linux.

## Caso Práctico 3 – Problema y solución: Copiar y pegar entre host y máquina virtual

### Problema

Al inicio, no era posible copiar texto desde el sistema host (Mac) hacia la máquina virtual (Ubuntu Desktop 24.04), ni viceversa. Esto dificultaba la instalación y configuración porque no se podían pegar comandos, rutas ni texto de forma cómoda dentro de la VM.

### Solución implementada: Instalación de Guest Additions

Para habilitar la funcionalidad de copiar y pegar entre el host y la VM, se procedió a instalar las **Guest Additions**, un conjunto de herramientas que mejoran la integración entre ambos sistemas, incluyendo soporte para portapapeles compartido y mejor rendimiento gráfico.

### Metodología

#### 1. Montar la imagen de Guest Additions:

Desde la barra de menú de VirtualBox, se seleccionó:

Dispositivos > Insertar imagen de CD de las Guest Additions...

#### 2. Instalar los paquetes necesarios para la compilación:

En la terminal de la VM se ejecutaron los siguientes comandos:

```
sudo apt update  
sudo apt install build-essential dkms linux-headers-$(uname -r)
```

#### 3. Ejecutar el instalador de Guest Additions:

Se identificó la ruta donde se montó el CD (puede variar según el usuario), y se ejecutó:

```
sudo sh /media/pamezampieri/VBox_GAs_7.1.8/VBoxLinuxAdditions.run
```

#### 4. Reiniciar la máquina virtual:

```
sudo reboot
```

**5. Configurar el portapapeles compartido:**

Desde la ventana de VirtualBox, en el menú:

Dispositivos > Portapapeles compartido > Bidireccional

**Resultados obtenidos**

Tras completar estos pasos, fue posible copiar y pegar texto de forma fluida entre el sistema host y la máquina virtual, lo que facilitó la interacción y mejoró la experiencia de uso al permitir copiar comandos largos, rutas y textos sin errores ni limitaciones.

## Caso Práctico 4 – Desarrollo en Python con Visual Studio Code

En lugar de instalar un servidor web, se optó por configurar la máquina virtual como entorno de desarrollo. Para ello, se instaló **Python 3.12.3** y el editor de código **Visual Studio Code**, y se verificó su correcto funcionamiento ejecutando un archivo de prueba.

### Metodología:

1. Se creó una máquina virtual Ubuntu utilizando VirtualBox con 2 núcleos, 2 GB de RAM y 20 GB de disco.

2. Se verificó que **Python 3** estaba preinstalado.

3. Se instaló **pip3** con:

```
sudo apt update
```

```
sudo apt install python3-pip -y
```

4. Se descargó e instaló **Visual Studio Code** desde el sitio oficial (<https://code.visualstudio.com/>) con el paquete .deb.

5. Durante la instalación, se aceptó la incorporación del repositorio oficial de Microsoft para permitir actualizaciones automáticas.

6. Se ejecutó Visual Studio Code con el comando:

```
code
```

7. Se creó una carpeta en el Escritorio llamada **python** y dentro de ella un archivo **hola.py**.

8. Se instaló la extensión oficial de **Python** para VS Code.

9. Se escribió el siguiente código de prueba:

```
print("¡Hola desde Ubuntu en una máquina virtual!")
```

10. Se ejecutó el archivo desde el propio editor utilizando la opción **Run > Run Without Debugging**, mostrando la salida correctamente en la terminal integrada.

## Desarrollo adicional: simulación en Python

Como práctica adicional y aplicación concreta, se reimplementó en Python un ejercicio del libro "**Cómo Programar en Java**" de **Deitel**, específicamente el ejercicio **7.28 - “Simulación: La tortuga y la liebre”**.

El juego consiste en una carrera simulada entre dos personajes, con movimiento aleatorio y visualización en consola, utilizando estructuras de control, funciones y manipulación de listas.

Esta adaptación permitió:

- Aplicar conocimientos de lógica de programación adquiridos previamente en Java.
- Reforzar el uso de Python como lenguaje de scripting en entornos Linux.
- Validar el entorno de desarrollo configurado en la VM para actividades educativas y de práctica.

## Resultados Obtenidos:

- Se logró configurar y utilizar una máquina virtual Ubuntu como entorno de desarrollo completo.
- Se comprendió cómo instalar software dentro de una VM y cómo integrar herramientas de desarrollo modernas.
- Se ejecutó exitosamente una adaptación de un ejercicio clásico de programación dentro de un entorno completamente virtualizado.

## Conclusiones

A lo largo del desarrollo de este trabajo integrador se logró comprender y aplicar de manera práctica los conceptos fundamentales de virtualización. Mediante la utilización de VirtualBox y una imagen de Ubuntu 24.04 Desktop LTS, se pudo simular un entorno real de trabajo sin necesidad de modificar el sistema operativo del host.

Durante el proceso, se enfrentaron y resolvieron diversos desafíos que enriquecieron el aprendizaje: desde la instalación de la máquina virtual hasta la correcta configuración del idioma y del teclado, la habilitación del portapapeles compartido para facilitar la interacción con el host, y la instalación de herramientas de desarrollo como Python y Visual Studio Code.

Cada una de estas acciones no sólo permitió configurar una máquina virtual funcional y adaptable, sino también profundizar en la comprensión de la relación entre sistema anfitrión e invitado, el uso de herramientas de virtualización, y la capacidad de preparar entornos específicos para distintos fines (como desarrollo de software).

En conclusión, la virtualización se presentó como una tecnología clave para la administración de sistemas, el aprendizaje, el desarrollo de software y la experimentación segura. El trabajo realizado no solo demostró competencias técnicas, sino también la capacidad de investigar, resolver problemas y documentar un proceso técnico de forma estructurada.

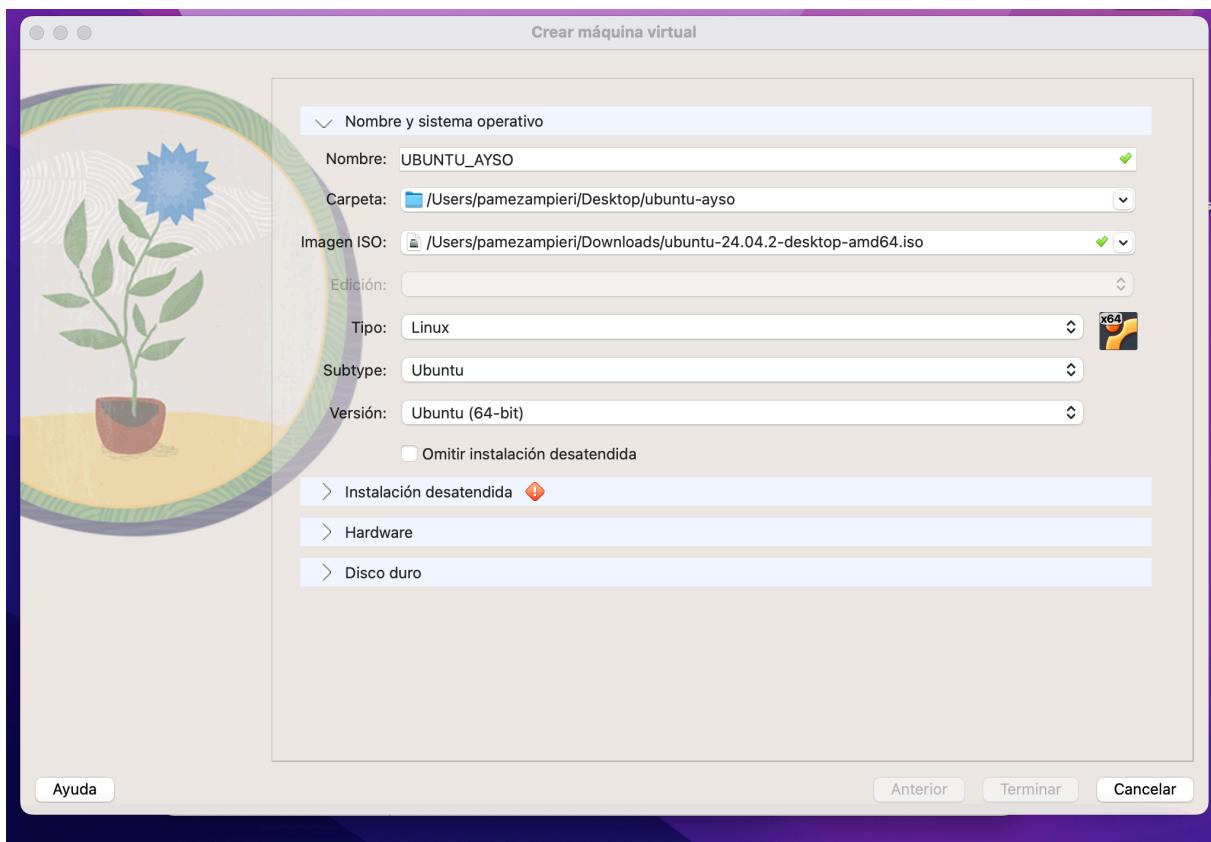
## Bibliografía

- **Oracle VirtualBox Manual**  
<https://www.virtualbox.org/manual/>
- **Ubuntu Desktop Documentation**  
<https://ubuntu.com/tutorials>
- **Guía de Redes Virtuales con VirtualBox – Arch Wiki**  
<https://wiki.archlinux.org/title/VirtualBox>
- **Python 3 Documentation**  
<https://docs.python.org/3/>
- **Visual Studio Code – Documentation**  
<https://code.visualstudio.com/docs>
- **Guía de instalación de VS Code en Ubuntu**  
<https://code.visualstudio.com/docs/setup/linux>

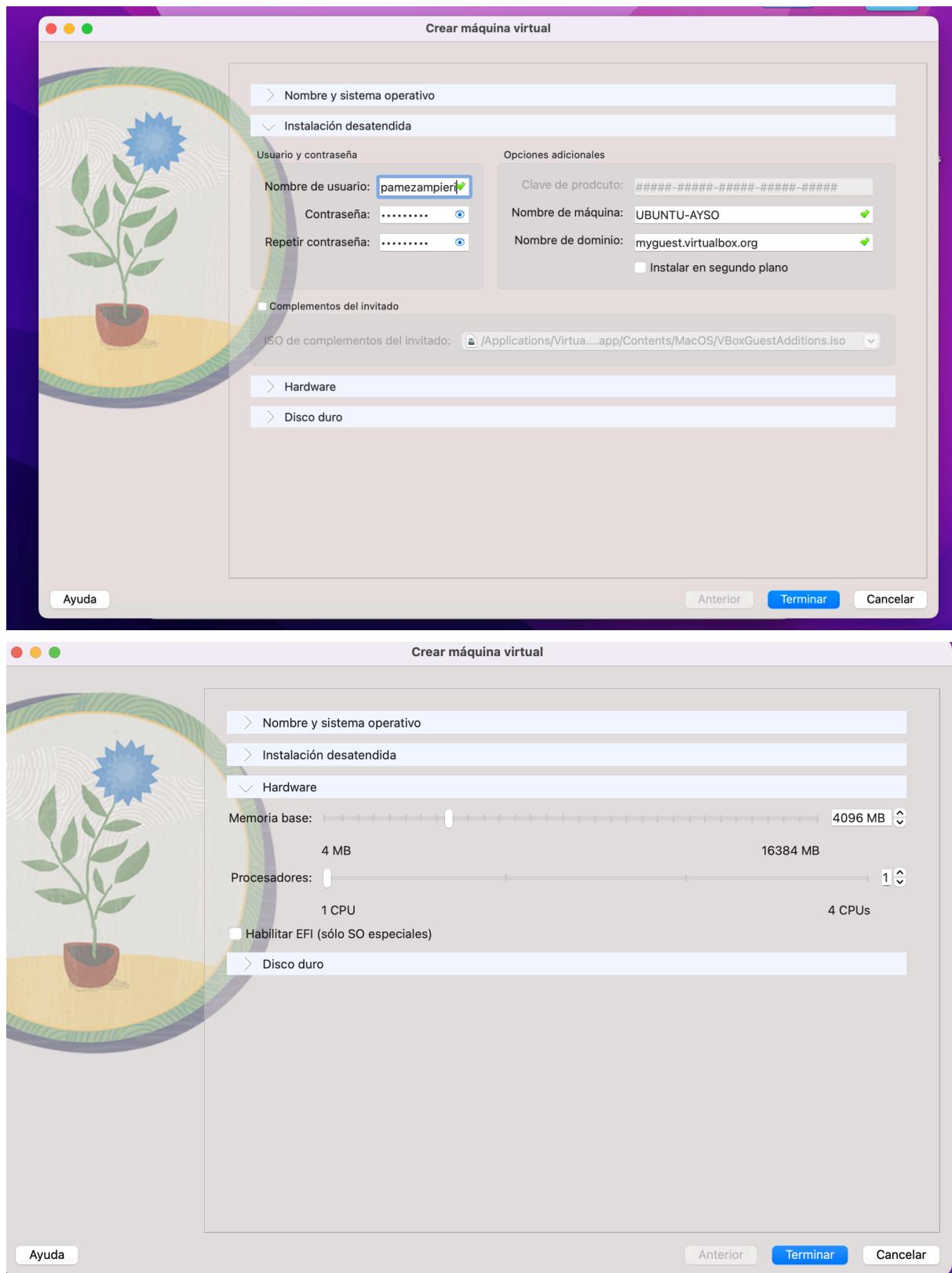
## Anexos

### Caso Práctico 1 – Instalación de VirtualBox y creación de máquina virtual con Ubuntu Desktop 24.04

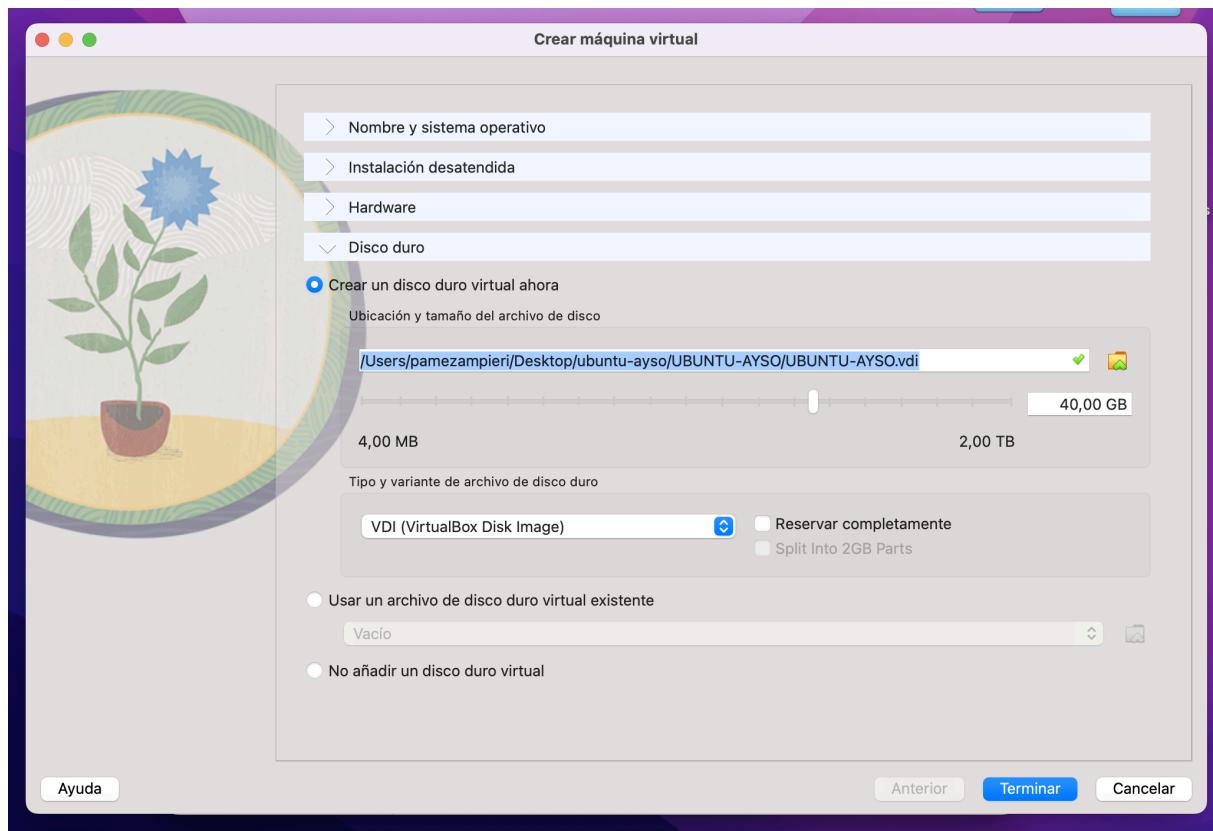
#### 1. Creación de máquina virtual:



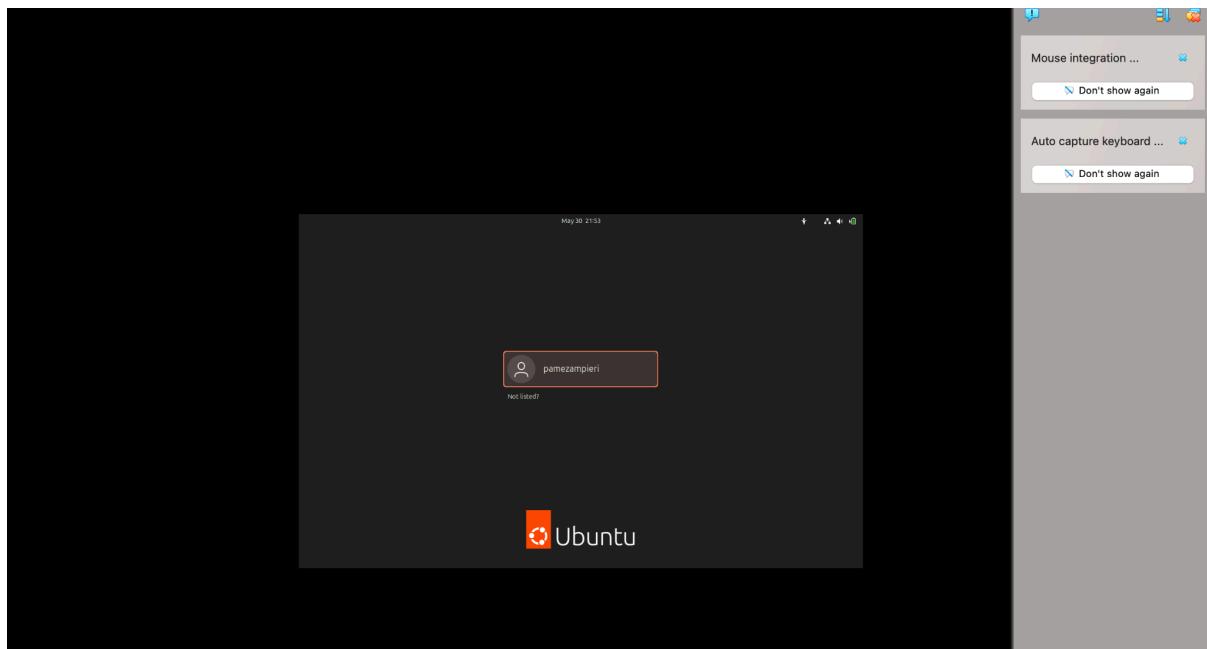
## TRABAJO PRÁCTICO INTEGRADOR



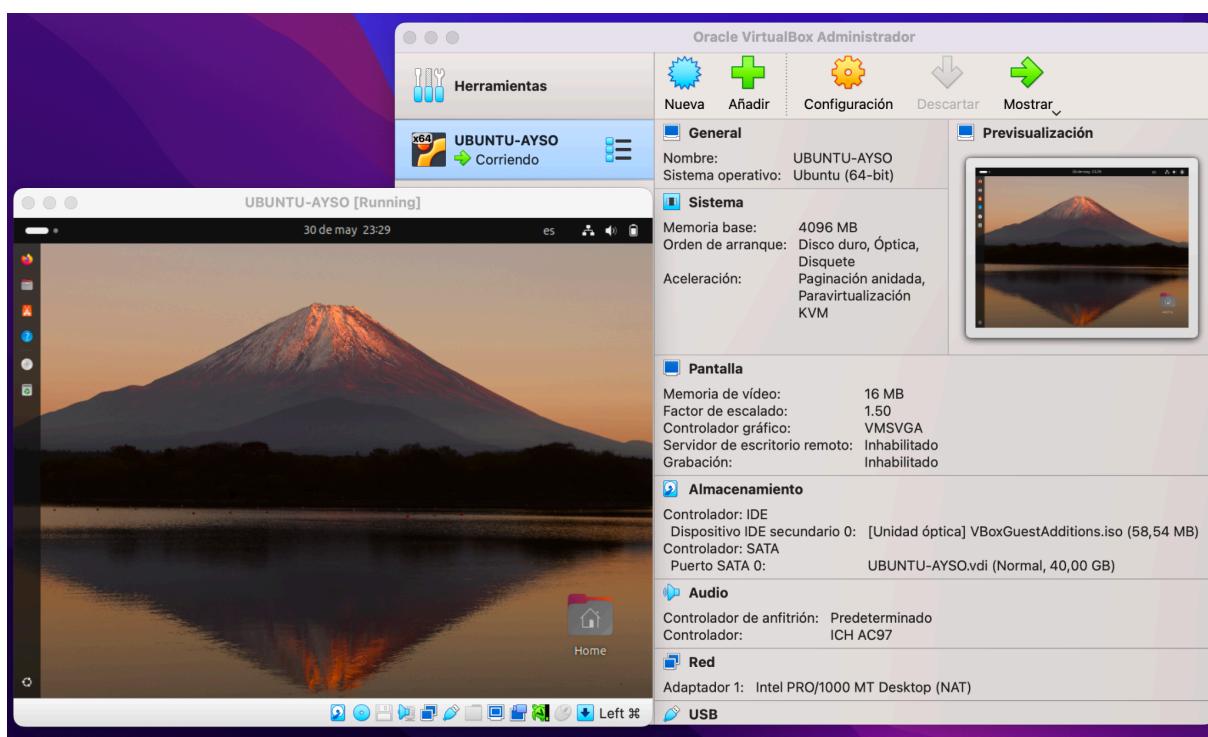
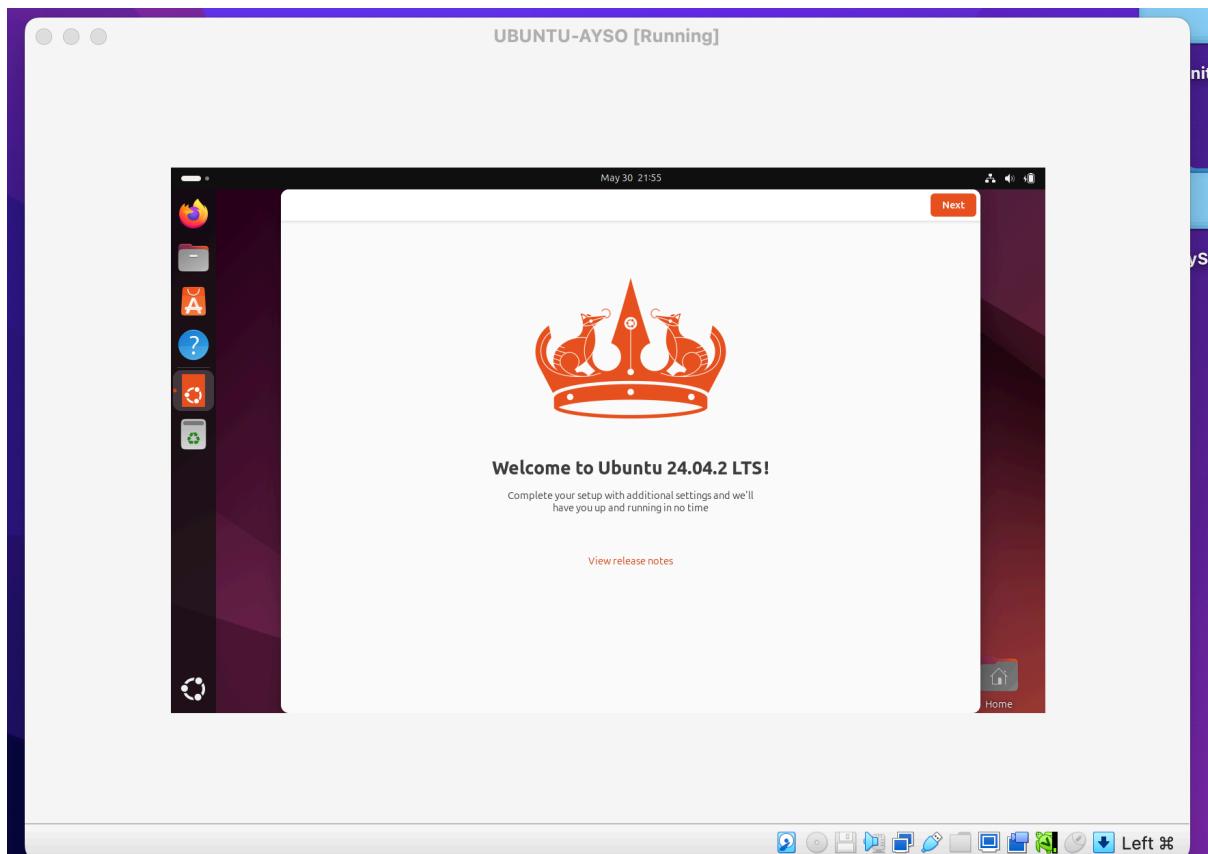
## TRABAJO PRÁCTICO INTEGRADOR



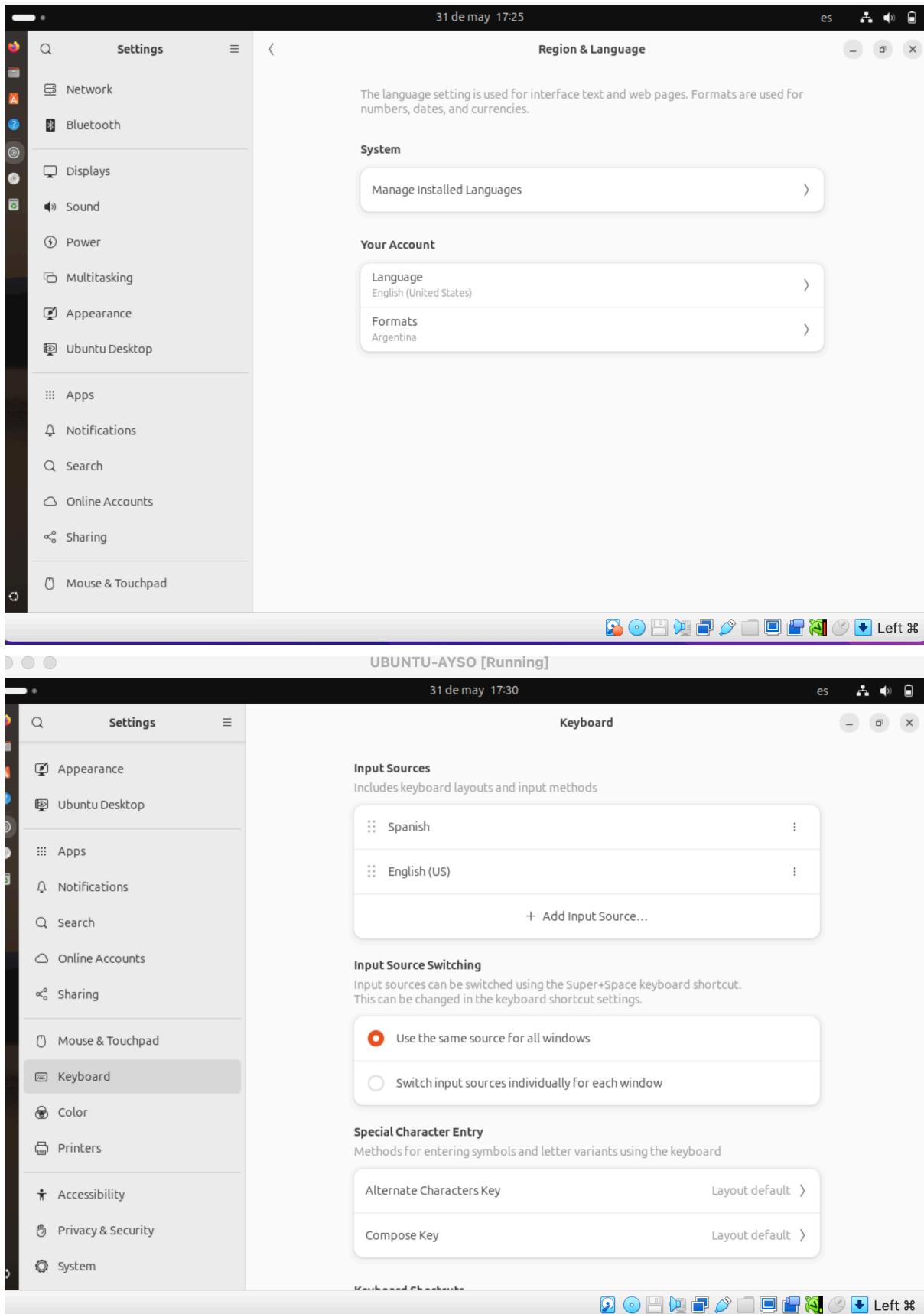
### 2. Inicialización de la máquina virtual:



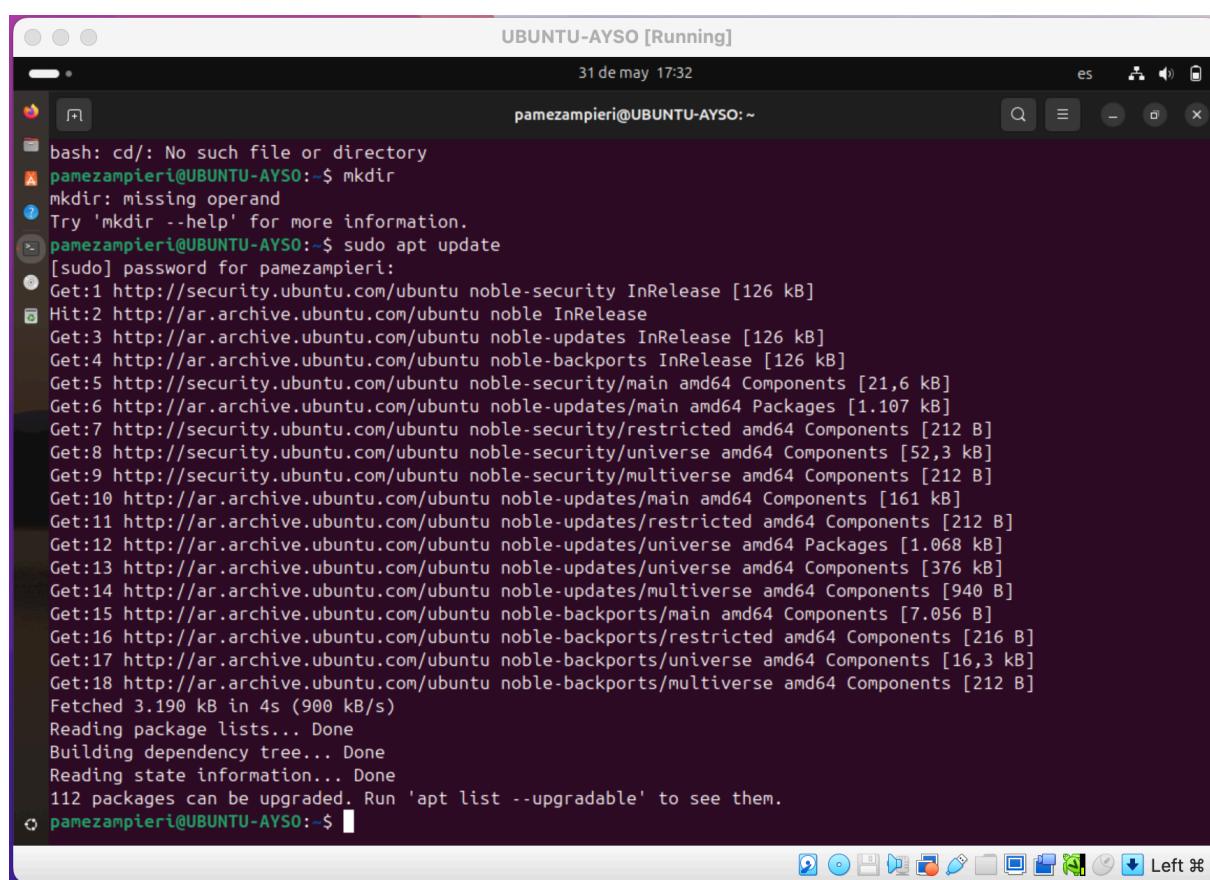
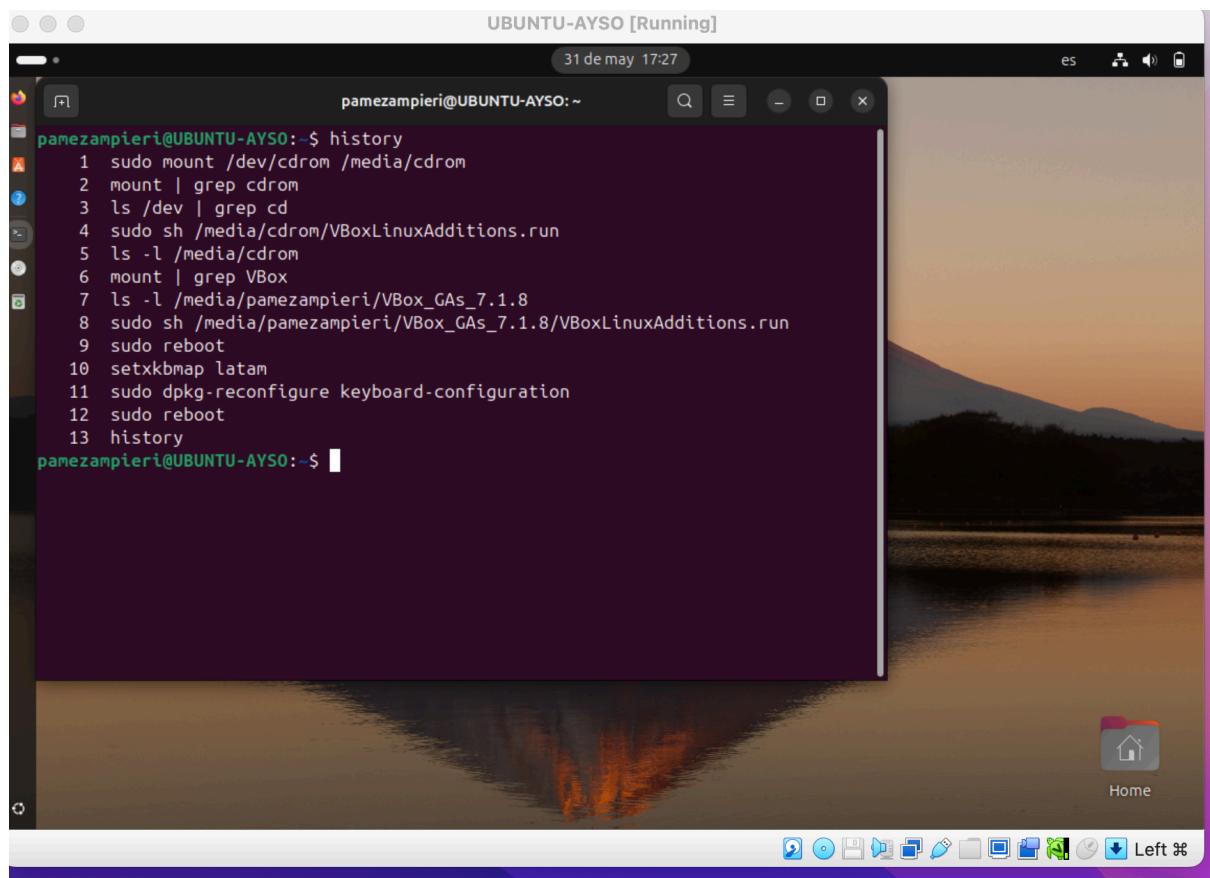
## TRABAJO PRÁCTICO INTEGRADOR



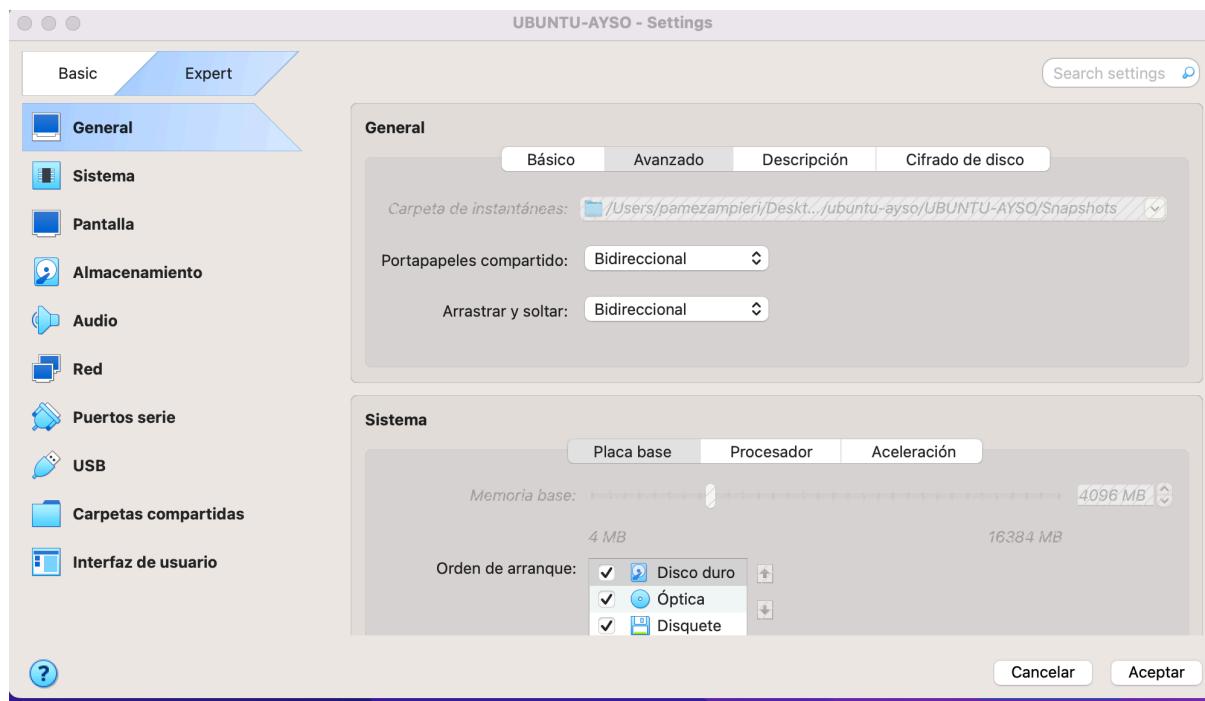
## Caso Práctico 2 – Configuración de idioma y teclado en Ubuntu Desktop 24.04



## TRABAJO PRÁCTICO INTEGRADOR



## TRABAJO PRÁCTICO INTEGRADOR



## Caso Práctico 3 – Problema y solución: Copiar y pegar entre host y máquina virtual

The screenshot shows a terminal window titled "UBUNTU-AYSO [Running]" with the command line "pamezampieri@UBUNTU-AYSO:~\$". The terminal displays a series of "Get" and "Hit" messages from an apt update process. A right-click context menu is open over the terminal output, with the "Copy" option highlighted. The menu also includes "Copy as HTML", "Paste", "Read-Only", "Preferences", "New Window", "New Tab", and "Show Menubar". The status bar at the bottom shows various icons and the text "adable' to see them."

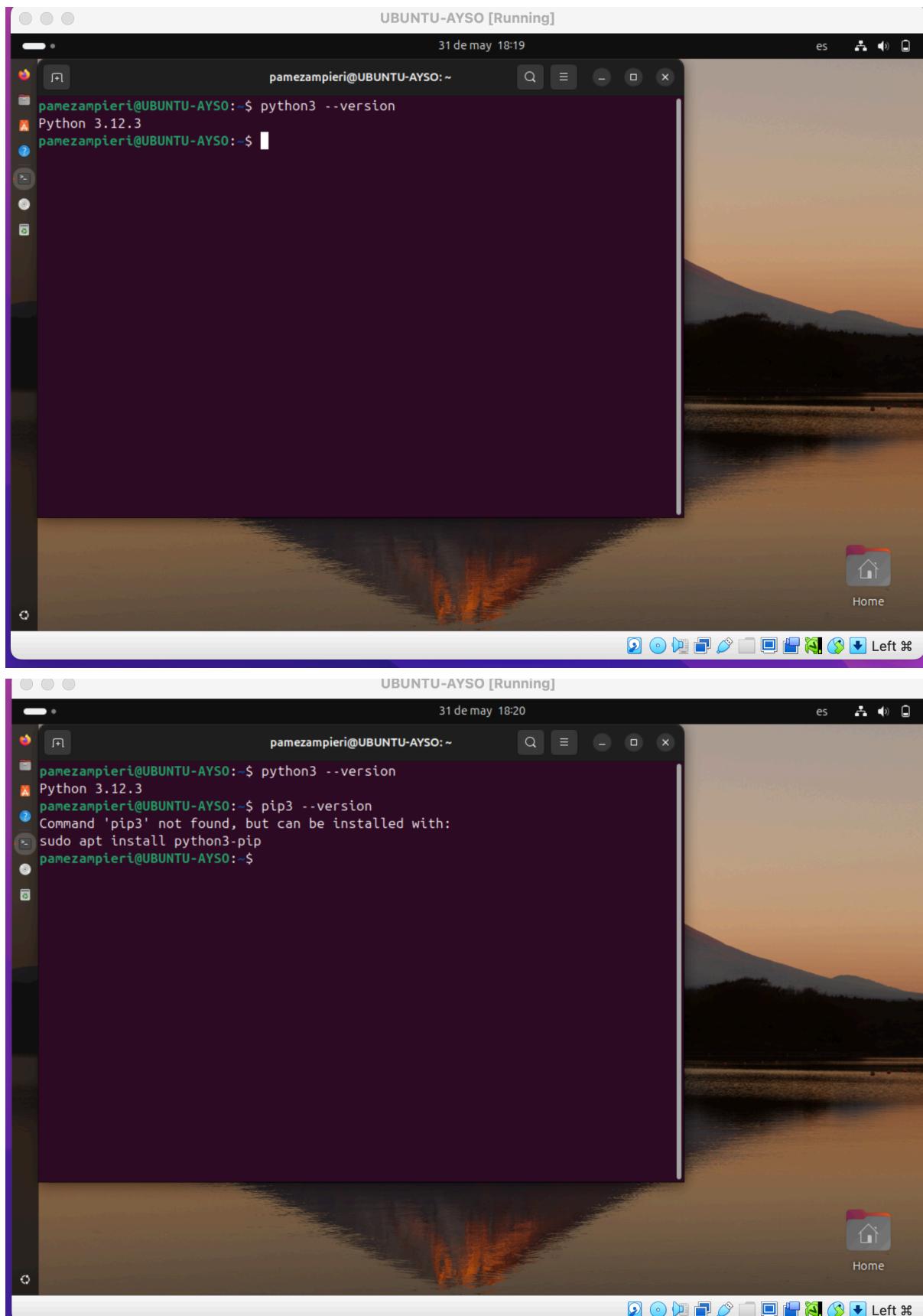
```
bash: cd/: No such file or directory
pamezampieri@UBUNTU-AYSO:~$ mkdir
mkdir: missing operand
Try 'mkdir --help' for more information.
pamezampieri@UBUNTU-AYSO:~$ sudo apt update
[sudo] password for pamezampieri:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://ar.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://ar.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://ar.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21,6 kB]
Get:6 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1.107 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52,3 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 kB]
Get:10 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [161 kB]
Get:11 http://ar.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 kB]
Get:12 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1.068 kB]
Get:13 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [376 kB]
Get:14 http://ar.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 kB]
Get:15 http://ar.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7.056 kB]
Get:16 http://ar.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 kB]
Get:17 http://ar.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [16,3 kB]
Get:18 http://ar.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 kB]
Fetched 3.190 kB in 4s (900 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
112 packages can be upgraded. Run 'sudo apt upgrade' to see them.
```

## TRABAJO PRÁCTICO INTEGRADOR

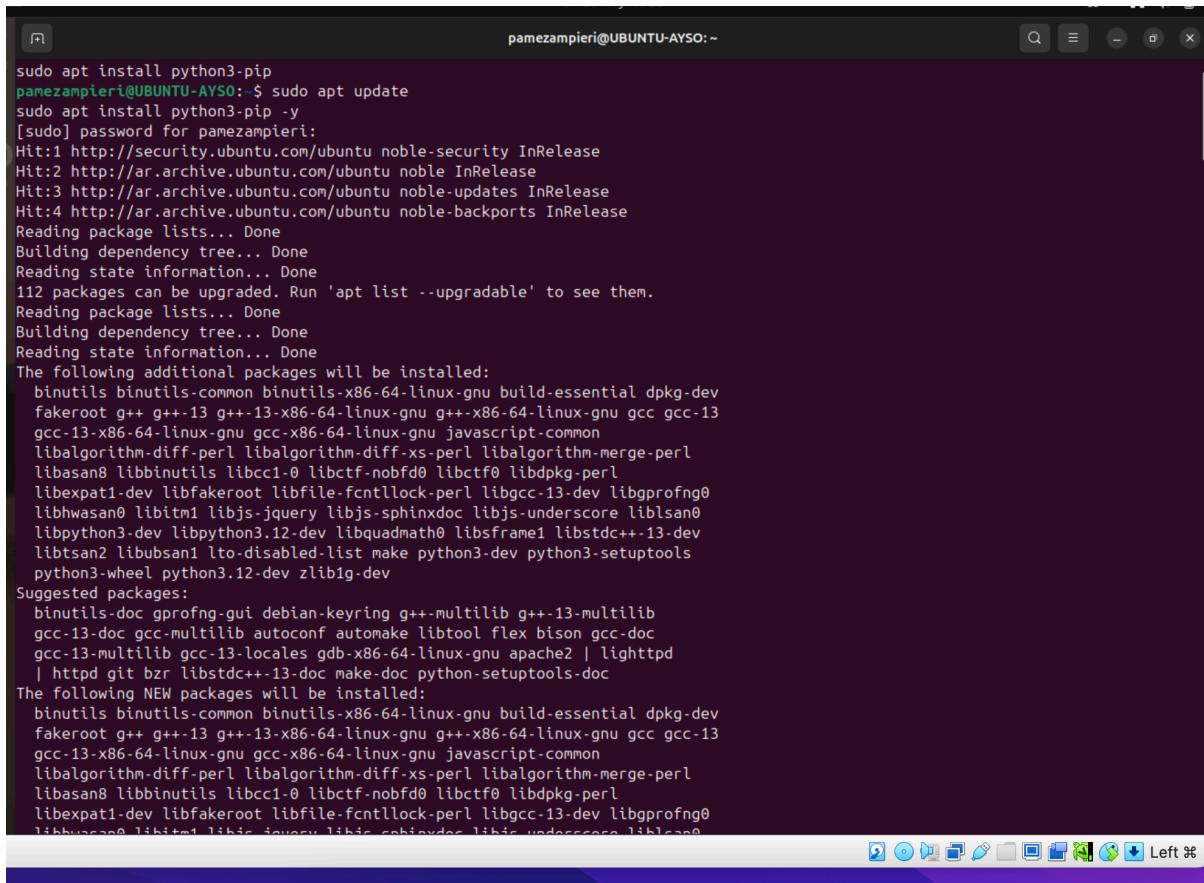
```
UBUNTU-AYSO [Running]
31 de may 17:33
pamezampieri@UBUNTU-AYSO:~ 
bash: cd/: No such file or directory
pamezampieri@UBUNTU-AYSO:~ $ mkdir
mkdir: missing operand
Try 'mkdir --help' for more information.
pamezampieri@UBUNTU-AYSO:~ $ sudo apt update
[sudo] password for pamezampieri:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://ar.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://ar.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://ar.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21,6 kB]
Get:6 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1.107 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52,3 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:10 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [161 kB]
Get:11 http://ar.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:12 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1.068 kB]
Get:13 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [376 kB]
Get:14 http://ar.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:15 http://ar.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7.056 B]
Get:16 http://ar.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:17 http://ar.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [16,3 kB]
Get:18 http://ar.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 3.190 kB in 4s (900 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
112 packages can be upgraded. Run 'apt list --upgradable' to see them.
pamezampieri@UBUNTU-AYSO:~ $ echo ";Hola, este teclado ya funciona bien!"
;Hola, este teclado ya funciona bien!
```

```
UBUNTU-AYSO [Running]
31 de may 17:34
pamezampieri@UBUNTU-AYSO:~ 
mkdir: missing operand
Try 'mkdir --help' for more information.
pamezampieri@UBUNTU-AYSO:~ $ sudo apt update
[sudo] password for pamezampieri:
Get:1 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:2 http://ar.archive.ubuntu.com/ubuntu noble InRelease
Get:3 http://ar.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:4 http://ar.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:5 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21,6 kB]
Get:6 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1.107 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:8 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52,3 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:10 http://ar.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [161 kB]
Get:11 http://ar.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:12 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1.068 kB]
Get:13 http://ar.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [376 kB]
Get:14 http://ar.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:15 http://ar.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7.056 B]
Get:16 http://ar.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:17 http://ar.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [16,3 kB]
Get:18 http://ar.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 3.190 kB in 4s (900 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
112 packages can be upgraded. Run 'apt list --upgradable' to see them.
pamezampieri@UBUNTU-AYSO:~ $ echo ";Hola, este teclado ya funciona bien!"
;Hola, este teclado ya funciona bien!
```

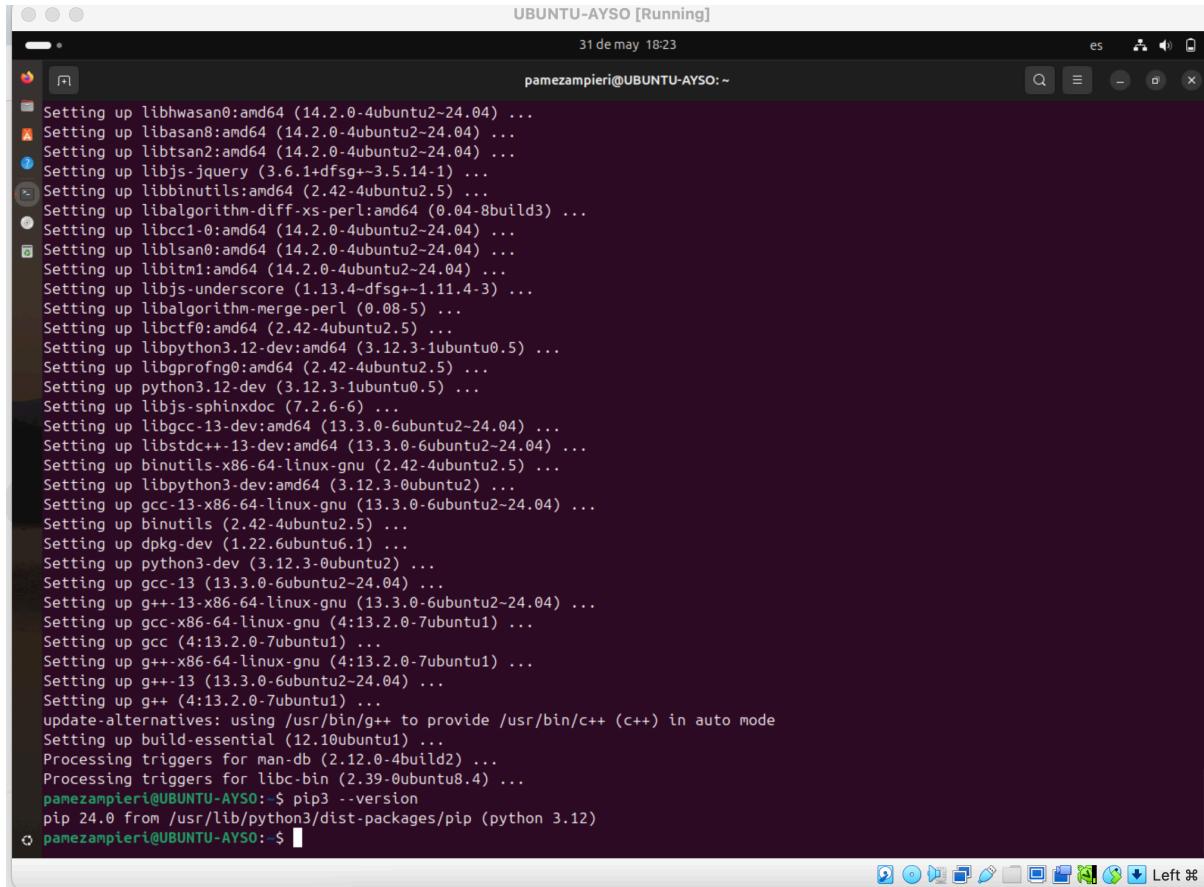
## Caso Práctico 4 – Desarrollo en Python con Visual Studio Code



## TRABAJO PRÁCTICO INTEGRADOR

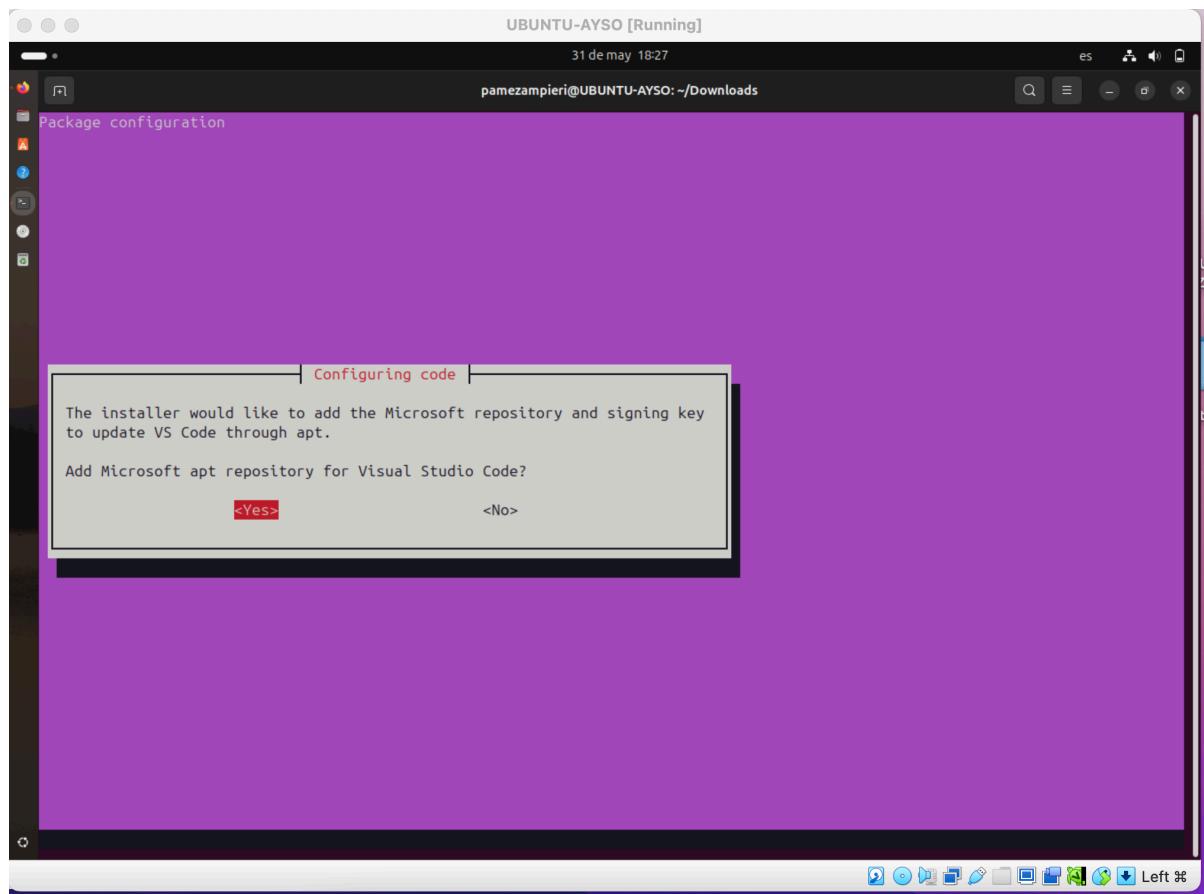
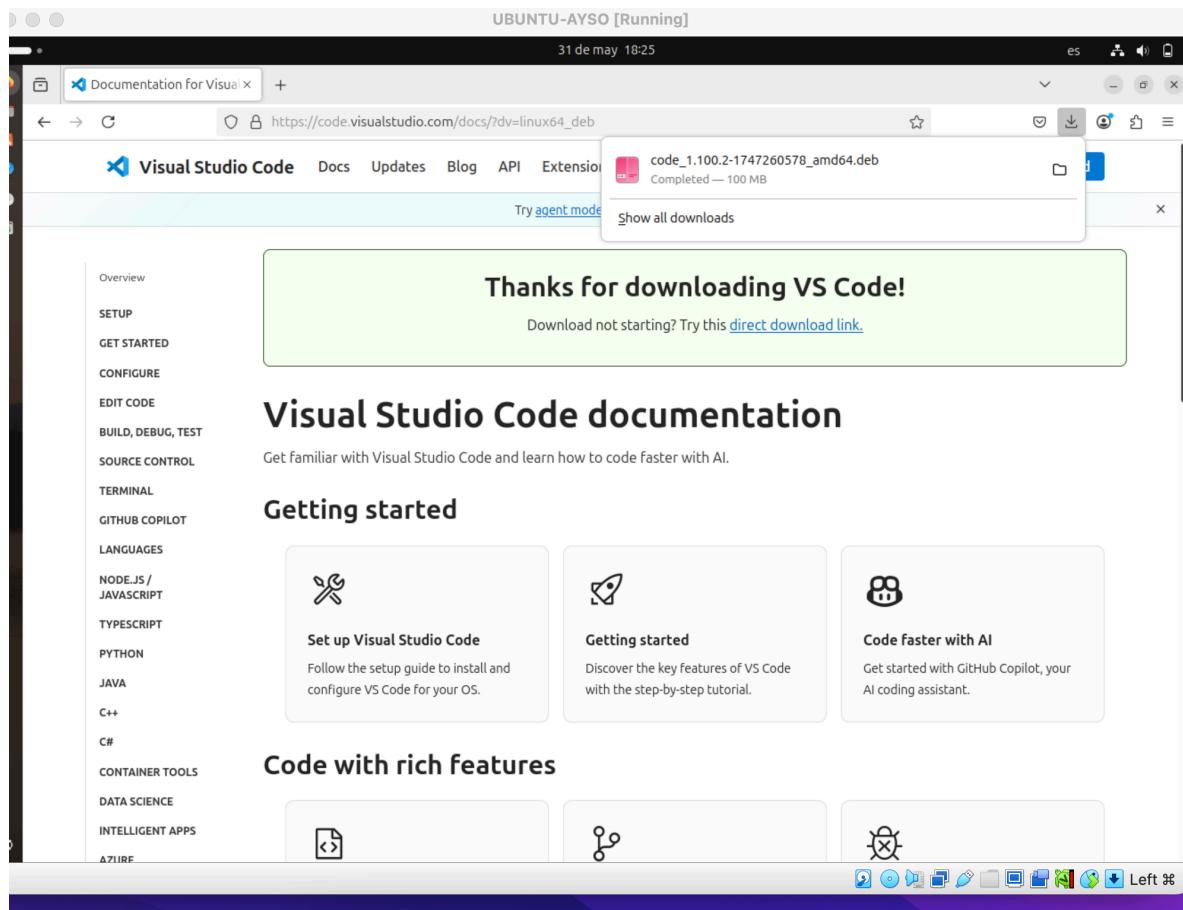


```
pamezampieri@UBUNTU-AYSO:~$ sudo apt update
[sudo] password for pamezampieri:
Hit:1 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:2 http://ar.archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://ar.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://ar.archive.ubuntu.com/ubuntu noble-backports InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
112 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev
  fakeroot g++-13 g++-13-x86-64-linux-gnu g++-x86-64-linux-gnu gcc gcc-13
  gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu javascript-common
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libasan8 libbinutils libgcc1-0 libctf-nobfd0 libctf0 libdpkg-perl
  libexpat1-dev libfakeroot libfile-fcnllock-perl libgcc-13-dev libgprofng0
  libhwasan0 libitm1 libjs-jquery libjs-sphinxdoc libjs-underscore liblsan0
  libpython3-dev libpython3.12-dev libquadmath0 libsframe1 libstdc++-13-dev
  libtsan2 libubsan1 lto-disabled-list make python3-dev python3-setuptools
  python3-wheel python3.12-dev zlib1g-dev
Suggested packages:
  binutils-doc gprofng-gui debian-keyring g++-multilib g++-13-multilib
  gcc-13-doc gcc-multilib autoconf automake libtool flex bison gcc-doc
  gcc-13-multilib gcc-13-locales gdb-x86-64-linux-gnu apache2 | lighttpd
  | httpd git bzip2 libstdc++-13-doc make-doc python-setuptools-doc
The following NEW packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev
  fakeroot g++-13 g++-13-x86-64-linux-gnu g++-x86-64-linux-gnu gcc gcc-13
  gcc-13-x86-64-linux-gnu gcc-x86-64-linux-gnu javascript-common
  libalgorithm-diff-perl libalgorithm-diff-xs-perl libalgorithm-merge-perl
  libasan8 libbinutils libgcc1-0 libctf-nobfd0 libctf0 libdpkg-perl
  libexpat1-dev libfakeroot libfile-fcnllock-perl libgcc-13-dev libgprofng0
  libhwasan0 libitm1 libjs-jquery libjs-sphinxdoc libjs-underscore liblsan0
```

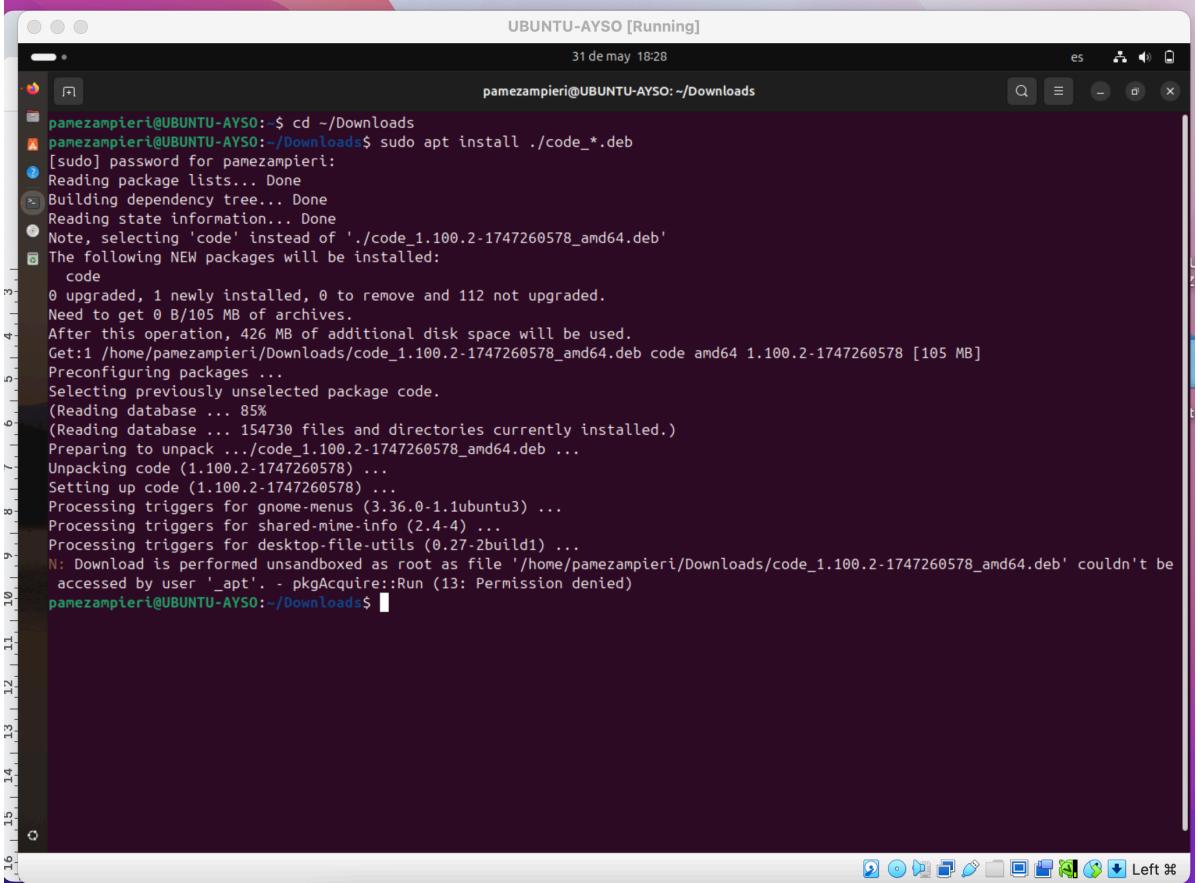


```
UBUNTU-AYSO [Running]
31 de may 18:23
pamezampieri@UBUNTU-AYSO:~$ 
Setting up libhwasan0:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up libasan8:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up libtsan2:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up libjs-jquery (3.6.1+dfsg+-3.5.14-1) ...
Setting up libbinutils:amd64 (2.42-4ubuntu2.5) ...
Setting up libalgorithm-diff-xs-perl:amd64 (0.04-8build3) ...
Setting up libbcc1-0:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up liblsan0:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up libitm1:amd64 (14.2.0-4ubuntu2-24.04) ...
Setting up libjs-underscore (1.13.4+dfsg+-1.11.4-3) ...
Setting up libalgorithm-merge-perl (0.08-5) ...
Setting up libctf0:amd64 (2.42-4ubuntu2.5) ...
Setting up libpython3.12-dev:amd64 (3.12.3-1ubuntu0.5) ...
Setting up libgprofng0:amd64 (2.42-4ubuntu2.5) ...
Setting up python3.12-dev (3.12.3-1ubuntu0.5) ...
Setting up libjs-sphinxdoc (7.2.6-6) ...
Setting up libgcc-13-dev:amd64 (13.3.0-6ubuntu2-24.04) ...
Setting up libstdc++-13-dev:amd64 (13.3.0-6ubuntu2-24.04) ...
Setting up binutils-x86-64-linux-gnu (2.42-4ubuntu2.5) ...
Setting up libpython3-dev:amd64 (3.12.3-0ubuntu2) ...
Setting up gcc-13-x86-64-linux-gnu (13.3.0-6ubuntu2-24.04) ...
Setting up binutils (2.42-4ubuntu2.5) ...
Setting up dpkg-dev (1.22.6ubuntu6.1) ...
Setting up python3-dev (3.12.3-0ubuntu2) ...
Setting up gcc-13 (13.3.0-6ubuntu2-24.04) ...
Setting up g++-13-x86-64-linux-gnu (13.3.0-6ubuntu2-24.04) ...
Setting up gcc-x86-64-linux-gnu (4:13.2.0-7ubuntu1) ...
Setting up gcc (4:13.2.0-7ubuntu1) ...
Setting up g++-x86-64-linux-gnu (4:13.2.0-7ubuntu1) ...
Setting up g++-13 (13.3.0-6ubuntu2-24.04) ...
Setting up g++ (4:13.2.0-7ubuntu1) ...
update-alternatives: using /usr/bin/g++ to provide /usr/bin/c++ (c++) in auto mode
Setting up build-essential (12.10ubuntu1) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.4) ...
pamezampieri@UBUNTU-AYSO:~$ pip3 --version
pip 24.0 from /usr/lib/python3/dist-packages/pip (python 3.12)
pamezampieri@UBUNTU-AYSO:~$
```

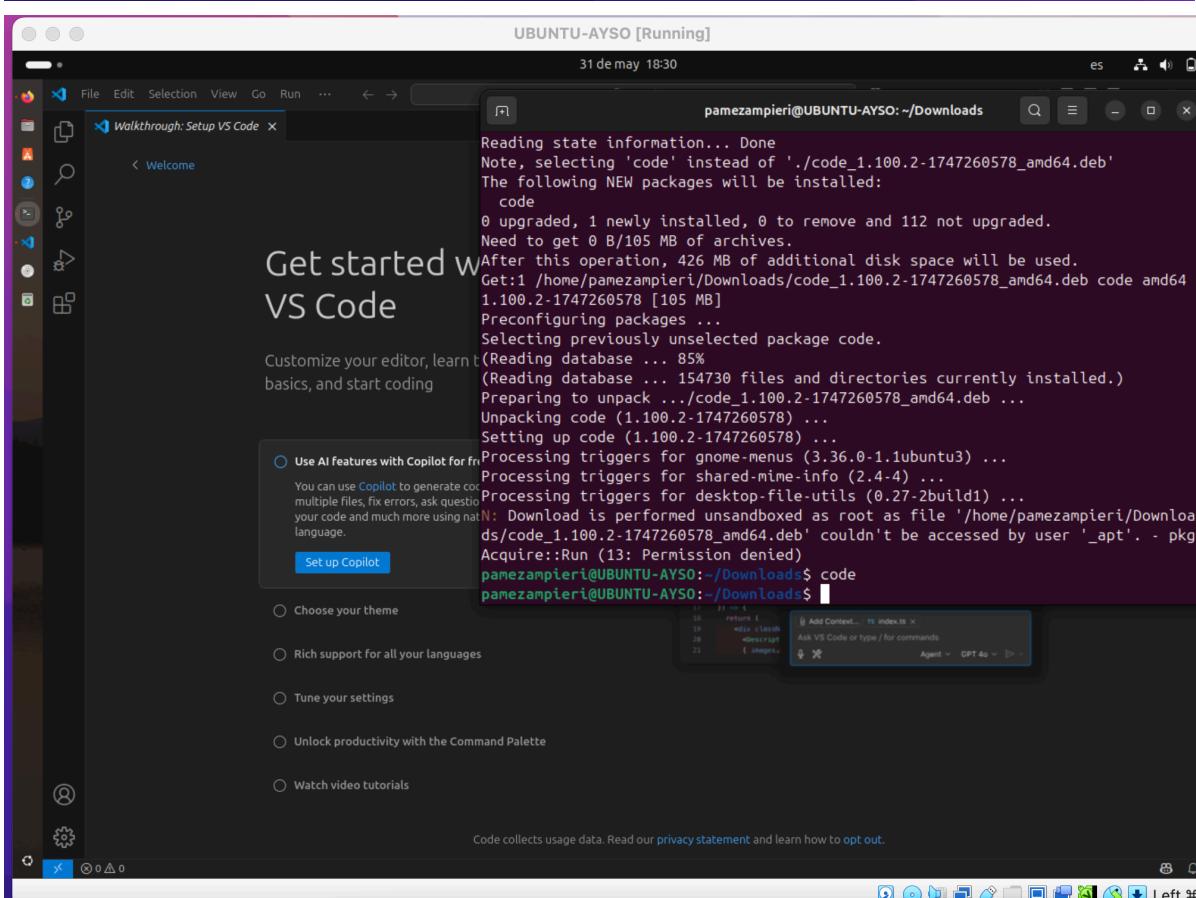
## TRABAJO PRÁCTICO INTEGRADOR



## TRABAJO PRÁCTICO INTEGRADOR

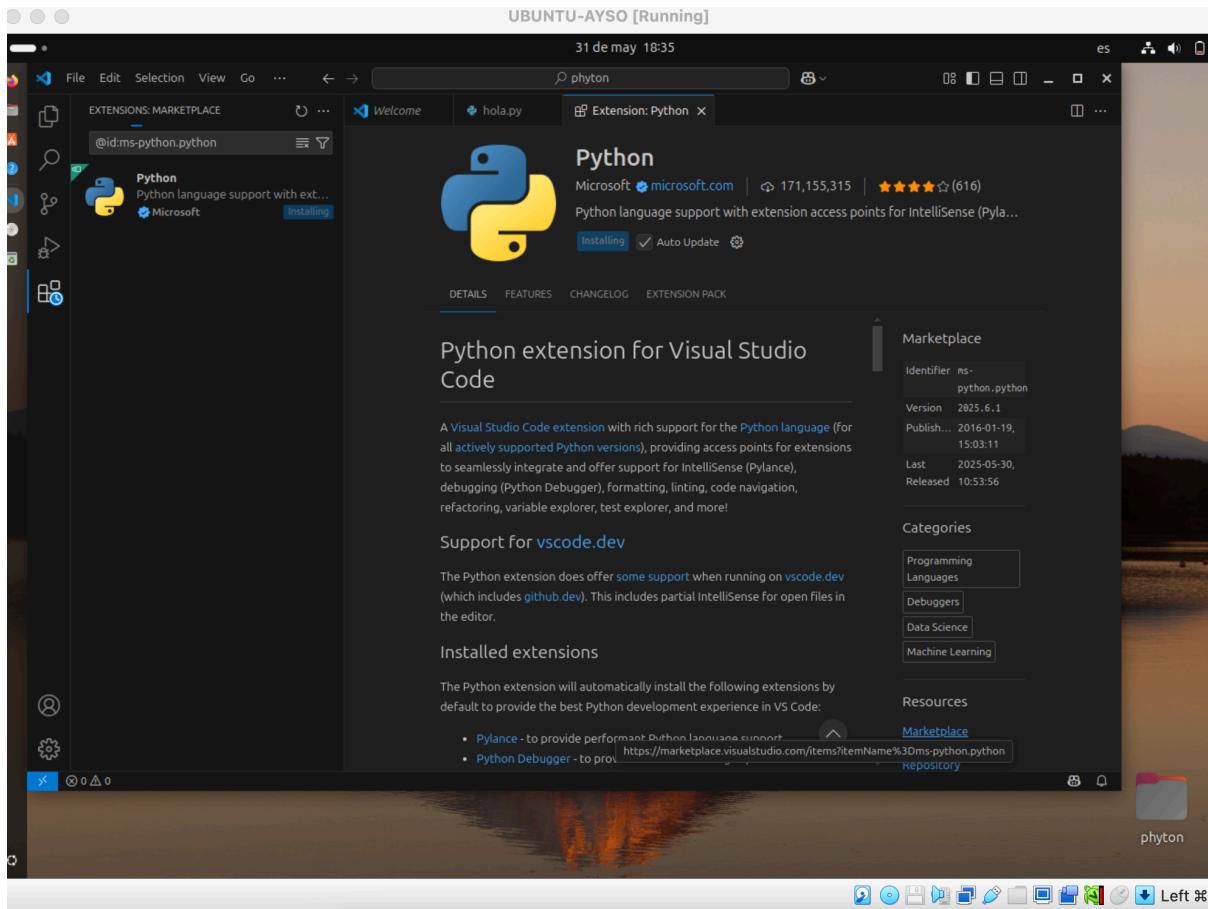
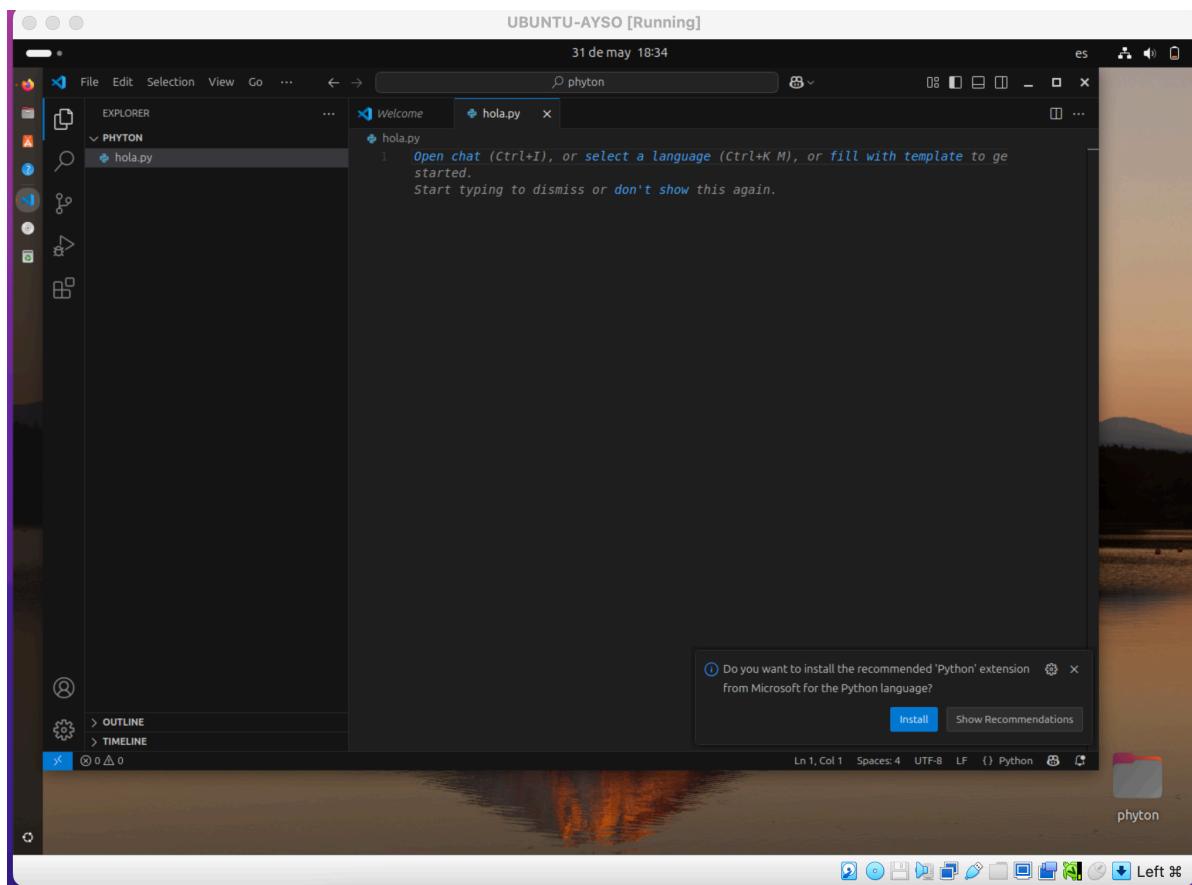


```
pamezampieri@UBUNTU-AYSO:~$ cd ~/Downloads
pamezampieri@UBUNTU-AYSO:~/Downloads$ sudo apt install ./code_* .deb
[sudo] password for pamezampieri:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Note, selecting 'code' instead of './code_1.100.2-1747260578_amd64.deb'
The following NEW packages will be installed:
  code
0 upgraded, 1 newly installed, 0 to remove and 112 not upgraded.
Need to get 0 B/105 MB of archives.
After this operation, 426 MB of additional disk space will be used.
Get:1 /home/pamezampieri/Downloads/code_1.100.2-1747260578_amd64.deb amd64 1.100.2-1747260578 [105 MB]
Preconfiguring packages ...
Selecting previously unselected package code.
(Reading database ... 85%
(Reading database ... 154730 files and directories currently installed.)
Preparing to unpack .../code_1.100.2-1747260578_amd64.deb ...
Unpacking code (1.100.2-1747260578) ...
Setting up code (1.100.2-1747260578) ...
Processing triggers for gnome-menus (3.36.0-1.1ubuntu3) ...
Processing triggers for shared-mime-info (2.4-4) ...
Processing triggers for desktop-file-utils (0.27-2build1) ...
N: Download is performed unsandboxed as root as file '/home/pamezampieri/Downloads/code_1.100.2-1747260578_amd64.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)
pamezampieri@UBUNTU-AYSO:~/Downloads$
```

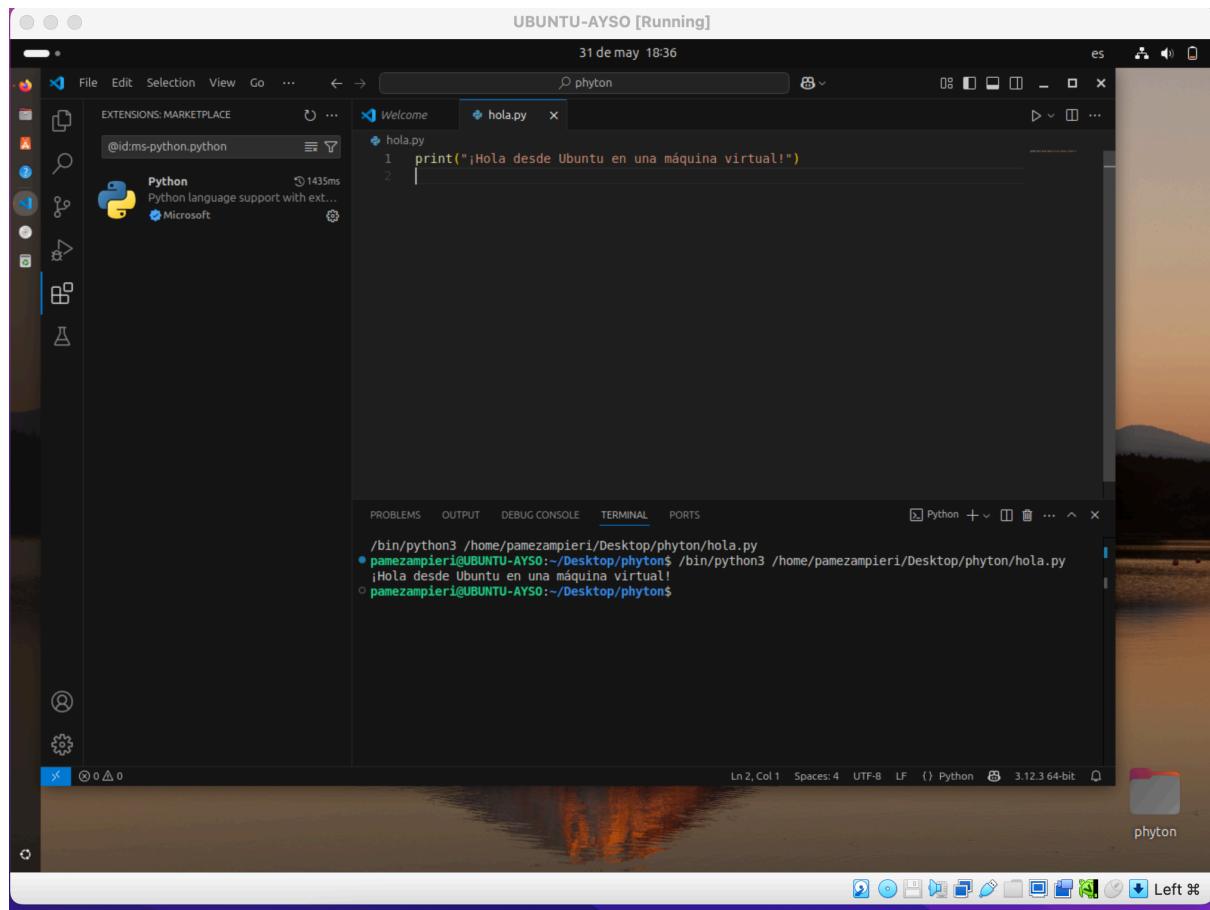
  


```
File Edit Selection View Go Run ... ← →
Walkthrough: Setup VS Code x
< Welcome
Reading state information... Done
Note, selecting 'code' instead of './code_1.100.2-1747260578_amd64.deb'
The following NEW packages will be installed:
  code
0 upgraded, 1 newly installed, 0 to remove and 112 not upgraded.
Need to get 0 B/105 MB of archives.
After this operation, 426 MB of additional disk space will be used.
Get:1 /home/pamezampieri/Downloads/code_1.100.2-1747260578_amd64.deb amd64 1.100.2-1747260578 [105 MB]
Preconfiguring packages ...
Selecting previously unselected package code.
(Reading database ... 85%
(Reading database ... 154730 files and directories currently installed.)
Preparing to unpack .../code_1.100.2-1747260578_amd64.deb ...
Unpacking code (1.100.2-1747260578) ...
Setting up code (1.100.2-1747260578) ...
Processing triggers for gnome-menus (3.36.0-1.1ubuntu3) ...
Processing triggers for shared-mime-info (2.4-4) ...
Processing triggers for desktop-file-utils (0.27-2build1) ...
N: Download is performed unsandboxed as root as file '/home/pamezampieri/Downloads/code_1.100.2-1747260578_amd64.deb' couldn't be accessed by user '_apt'. - pkgAcquire::Run (13: Permission denied)
pamezampieri@UBUNTU-AYSO:~/Downloads$ code
pamezampieri@UBUNTU-AYSO:~/Downloads$
```

## TRABAJO PRÁCTICO INTEGRADOR



## TRABAJO PRÁCTICO INTEGRADOR



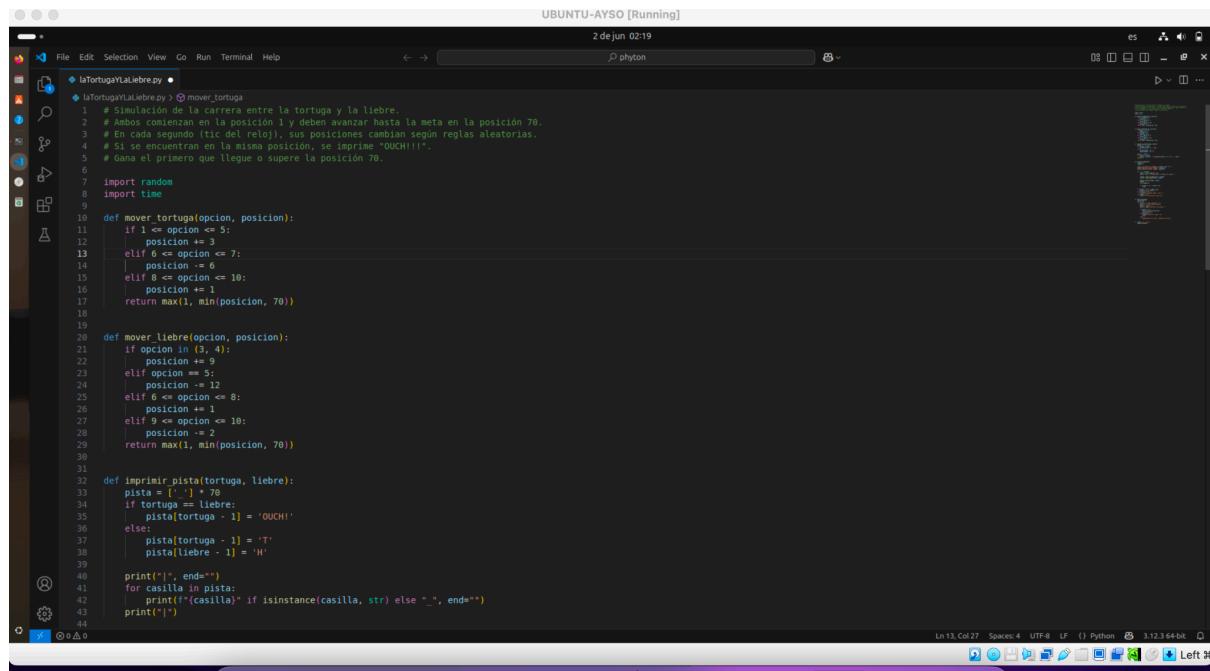
The screenshot shows the Visual Studio Code interface running on an Ubuntu system. The code editor has a file named `hola.py` open with the following content:

```
1 print("¡Hola desde Ubuntu en una máquina virtual!")
```

The terminal below shows the output of running the script:

```
/bin/python3 /home/pamezampieri/Desktop/phyton/hola.py
● pamezampieri@UBUNTU-AYSO:~/Desktop/phyton$ /bin/python3 /home/pamezampieri/Desktop/phyton/hola.py
¡Hola desde Ubuntu en una máquina virtual!
○ pamezampieri@UBUNTU-AYSO:~/Desktop/phyton$
```

## Desarrollo adicional: simulación en Python



The screenshot shows the Visual Studio Code interface running on an Ubuntu system. The code editor has a file named `laTortugaLaLiebre.py` open with the following content:

```
1 # Simulación de la carrera entre la tortuga y la liebre.
2 # Ambos comienzan en la posición 1 y deben avanzar hasta la meta en la posición 70.
3 # En cada segundo (tic del reloj), sus posiciones cambian según reglas aleatorias.
4 # Si se encuentran en la misma posición, se imprime "OUCH!".
5 # Gana el primero que llegue o supere la posición 70.
6
7 import random
8 import time
9
10 def mover_tortuga(opcion, posicion):
11     if 1 <= opcion <= 5:
12         posicion += 3
13     elif 6 <= opcion <= 7:
14         posicion -= 6
15     elif 8 <= opcion <= 10:
16         posicion += 1
17     return max(1, min(posicion, 70))
18
19 def mover_liebre(opcion, posicion):
20     if opcion in (3, 4):
21         posicion += 9
22     elif opcion == 5:
23         posicion -= 5
24     elif 6 <= opcion <= 8:
25         posicion += 8
26     elif 9 <= opcion <= 10:
27         posicion -= 2
28     return max(1, min(posicion, 70))
29
30
31 def imprimir_pista(tortuga, liebre):
32     pista = ['_'] * 70
33     if tortuga == liebre:
34         pista[tortuga - 1] = 'OUCH!'
35     else:
36         pista[tortuga - 1] = 'T'
37         pista[liebre - 1] = 'H'
38
39     print("[", end="")
40     for casilla in pista:
41         print(f"{casilla}" if isinstance(casilla, str) else "_", end="")
42     print("]")
```

## TRABAJO PRÁCTICO INTEGRADOR

```
UBUNTU-AYSO [Running]
File Edit Selection View Go Run Terminal Help
laTortugaYLaLiebre.py > correr_carrera
46 def correr_carrera(num):
47     tortuga = 1
48     liebre = 1
49
50     print(f"\n\n PUM!! Y ARRANCA LA CARRERA ({num}!!!\n")
51     print(f"POSICIÓN INICIAL TORTUGA: {tortuga}")
52     print(f"POSICIÓN INICIAL LIEBRE: {liebre}\n")
53
54     for _ in range(60):
55         reloj = random.randint(1, 10)
56         print(f"• Ajustar las posiciones en {reloj} para ambos")
57
58         tortuga = mover_tortuga(reloj), tortuga
59         liebre = mover_liebre(reloj), liebre
60
61         imprimir_pista(tortuga, liebre)
62         print()
63         time.sleep(0.3)
64
65         if tortuga[0] == 70 or liebre[0] == 70:
66             break
67
68         if tortuga[0] == 70 and liebre[0] == 70:
69             print(" Es un empate!!!")
70         elif tortuga[0] >= 70:
71             print(" LA TORTUGA GANA!!! YAY!!!")
72         elif liebre[0] >= 70:
73             print(" La Liebre gana. ¡Qué mal!")
74
75
76     def menu_principal():
77         carrera_num = 1
78         while True:
79             print("\n*** MENU PRINCIPAL ***")
80             print("1. Iniciar nueva carrera")
81             print("2. Salir")
82             opcion = input("Selecciona una opción: ")
83
84             if opcion == "1":
85                 correr_carrera(carrera_num)
86                 carrera_num += 1
87             elif opcion == "2":
88                 print("¡Gracias por jugar! ☺")
89                 break
90             else:
91                 print("Opción no válida. Intentalo de nuevo.")
92
93
94     if __name__ == "__main__":
95         menu_principal()
```

```
UBUNTU-AYSO [Running]
File Edit Selection View Go Run Terminal Help
laTortugaYLaLiebre.py > correr_carrera
76     def menu_principal():
77         carrera_num = 1
78         while True:
79             print("\n*** MENU PRINCIPAL ***")
80             print("1. Iniciar nueva carrera")
81             print("2. Salir")
82             opcion = input("Selecciona una opción: ")
83
84             if opcion == "1":
85                 correr_carrera(carrera_num)
86                 carrera_num += 1
87             elif opcion == "2":
88                 print("¡Gracias por jugar! ☺")
89                 break
90             else:
91                 print("Opción no válida. Intentalo de nuevo.")
92
93
94     if __name__ == "__main__":
95         menu_principal()
```

## TRABAJO PRÁCTICO INTEGRADOR

The screenshot shows a terminal window titled "UBUNTU-AYSO [Running]" with the date and time "2 de jun 02:22". The window title bar also includes "pythont" and "es". The terminal interface has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS. The main area displays a race simulation between two characters, Tortoise and Hare, with their positions being adjusted in increments of 1 from 1 to 10. The output shows the following sequence of moves:

```
OUCH!
Ajustar las posiciones en 6 para ambos!
Ajustar las posiciones en 4 para ambos!
Ajustar las posiciones en 7 para ambos!
Ajustar las posiciones en 10 para ambos!
Ajustar las posiciones en 8 para ambos!
Ajustar las posiciones en 8 para ambos!
Ajustar las posiciones en 2 para ambos!
Ajustar las posiciones en 4 para ambos!
Ajustar las posiciones en 2 para ambos!
Ajustar las posiciones en 8 para ambos!
Ajustar las posiciones en 1 para ambos!
Ajustar las posiciones en 7 para ambos!
Ajustar las posiciones en 6 para ambos!
Ajustar las posiciones en 3 para ambos!
➊ La Liebre gana. ¡Qué mal!
@@@ MENU PRINCIPAL ===
1. Iniciar nueva carrera
2. Salir
Selecciona una opción: [
```

The status bar at the bottom indicates "Ln 72, Col 23" and "Spaces:4 - UTF-8 LF (Python 3.12.3 64-bit)". There are also several icons in the status bar.

## TRABAJO PRÁCTICO INTEGRADOR

**Link video demostrativo del entorno y ejecución del código:**