Experiment 4

Name: Mahi Palimkar Reg ID:231071036

Batch B

**Aim**: Introduction to Integrated Development Environments (IDEs) and text editor, using VS Code and Sublime Text, Customizing development environments.

# Theory:

## What is an IDE? Explain its purpose and features.

An Integrated Development Environment (IDE) is a software suite that provides developers with essential tools for software development. It typically includes a source code editor, build automation utilities, and a debugger, all bundled into one cohesive application.

* + **Main Objective**: The primary role of an IDE is to enhance the software development process by offering all necessary tools within a unified interface. This integration minimizes the need to switch between different software, boosting developer productivity and streamlining workflows.
  + **Common IDEs**: Popular IDEs include VS Code, PyCharm, Eclipse, and Xcode, each catering to different programming languages and development needs.

## Core Features of an IDE:

* + - **Source Code Editor**: This is where developers write and edit their code, with features like syntax highlighting, code folding, and auto-completion to improve coding efficiency.
    - **Debugger**: The debugger helps track down and fix bugs in the code. It allows developers to set breakpoints, step through code, and inspect variables to diagnose issues.
    - **Compiler**: An IDE's compiler translates the written source code (in a high-level language) into machine code, which the computer can execute directly.
    - **Build Automation**: This feature manages project configurations and automates the building process, including managing dependencies and compiling code.
    - **GUI Designers**: Some IDEs come with visual tools that allow developers to design graphical user interfaces (GUIs) using drag-and-drop components, simplifying UI design.
    - **Integrated Terminal**: An IDE typically includes a terminal or console, allowing developers to run command-line tools and scripts directly within the development environment.

# How do editors differ from IDEs? Provide examples of when you might choose one over the other.

## Text Editors vs. IDEs

* + **Text Editors**: Primarily used for editing plain text, including code.
  + **IDEs**: Designed for comprehensive software development, covering coding, compiling, debugging, and testing.
  + **Text Editors**: Offer basic features like syntax highlighting and search-and-replace functionality.
  + **IDEs**: Provide advanced tools such as debuggers, compilers, version control systems, and testing frameworks.
  + **Text Editors**: Known for being simple, lightweight, and fast.
  + **IDEs**: Tend to be more complex, feature-rich, and resource-intensive.
  + **Text Editors**: Highly customizable through plugins and extensions, allowing for a tailored experience.
  + **IDEs**: Support plugins but come with a wide range of built-in features, offering a more standardized environment.
  + **Text Editors**: Examples include Notepad++, Sublime Text, Vim, and Atom.
  + **IDEs**: Examples include Visual Studio, Eclipse, PyCharm, and IntelliJ IDEA.

## When to Choose a Text Editor:

* + For quick edits to files like configurations or small scripts.
  + On older machines or remote servers with limited resources.
  + When you prefer a straightforward environment without additional tools.
  + If you need a lightweight and highly customizable setup.

## When to Choose an IDE:

* + For complex software development involving multiple files and dependencies.
  + When you need integrated tools for coding, debugging, and testing in one place.
  + For projects requiring version control and collaboration among team members.
  + When working with specific frameworks or databases that benefit from integrated support within the IDE.

# Advantages and disadvantages of using lightweight text editor like sublime text compared to more feature rich like VS Code.

## Advantages of Using Sublime Text Over VS Code:

* + **Speed and Responsiveness**: Sublime Text is extremely fast and responsive, even when handling large files, whereas VS Code can sometimes be sluggish with bigger files.
  + **Resource Efficiency**: Sublime Text consumes minimal system resources, making it lightweight on memory and CPU usage. In contrast, VS Code tends to use more memory and CPU power.
  + **Simplicity**: Sublime Text offers a clean, distraction-free interface focused purely on coding, while VS Code can feel overwhelming due to its numerous features and options.
  + **Customization**: Sublime Text is easier to customize with a quick setup process, allowing users to get up and running faster than with VS Code.

## Advantages of Using VS Code Over Sublime Text:

* + **Integrated Tools**: VS Code comes with built-in debugging, terminal, and Git support, which Sublime Text lacks. This makes VS Code more versatile for development tasks.
  + **Extension Ecosystem**: VS Code has a vast array of extensions with deep integration into the IDE, while Sublime Text has fewer extensions and less seamless integration.
  + **Cloud and Collaboration**: VS Code offers strong integration with cloud services and live collaboration tools, features that are largely absent in Sublime Text.

# Explain steps involved in installing VS Code on Linux Debian System.

* Open the command-line shell. Use command- sudo apt update. It is used to update your system's package list to ensure you have the latest information on available packages.
* sudo apt install software-properties-common apt-transport-https wget. This command is used for installing dependencies.
* Import Microsoft GPG Key. Use command:

wget -qO- https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > packages.microsoft.gpg

sudo install -o root -g root -m 644 packages.microsoft.gpg /usr/share/keyrings/

* Add the Microsoft VS Code repository to your system's sources list.

sudo sh -c 'echo "deb [arch=amd64 signed-by=/usr/share/keyrings/packages.microsoft.gpg] https://packages.microsoft.com/repos/vscode stable main" > /etc/apt/sources.list.d/vscode.list'.

* Now that the repository is set up, update your package list and install Visual Studio Code. Use the following commands:

Sudo apt update Sudo apt install code.

* After the installation is complete, you can launch VS Code using the command ‘code’ in your terminal.

# How to customize the appearance of VS Code? Discuss the process of changing themes and adjusting the layout.

1. Changing Themes

VS Code offers a wide range of themes that alter the color scheme of the editor, including the background, syntax highlighting, and UI elements.

Steps to Change the Theme:

* + Open the Command Palette:
    - Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (macOS) to open the Command Palette.
  + Search for Themes:
    - Type "Preferences: Color Theme" and press Enter. This will bring up a list of installed themes.
  + Select a Theme:
    - Scroll through the list and click on a theme to preview it. Once you find a theme you like, click on it to apply it.
  + Install New Themes:
    - If you want more options, click on "Install Additional Color Themes..." at the bottom of the theme list. This will take you to the VS Code Marketplace, where you can browse and install new themes.

1. Adjusting the Layout

VS Code's layout can be customized to suit your workflow, including the arrangement of panels, sidebars, and the editor itself.

Customizing the Layout:

* + Sidebars:
    - The sidebar can be toggled by clicking the icons on the Activity Bar (usually on the left). You can move the sidebar to the right by right-clicking the Activity Bar and selecting "Move Side Bar Right".
  + Panel Position:
    - The panel (which includes the terminal, output, problems, and debug console) is usually at the bottom. You can move it to the right side by clicking on the three dots in the panel’s title bar and selecting "Move Panel Right".
  + Split Editor View:
    - You can split the editor to view multiple files side by side by right-clicking on a file tab and selecting "Split Right", "Split Down", or dragging the tab to the desired location.
  + Zen Mode:
    - For a distraction-free environment, you can enable Zen Mode by going to "View" > "Appearance" > "Zen Mode" or pressing Ctrl+K Z (Windows/Linux) or Cmd+K Z (macOS). Zen Mode hides the Activity Bar, Sidebar, and Panel, leaving just the editor.

1. Customizing Font and Icons

You can also change the font style and icon set for a more personalized experience. Changing Font Style:

* + Open Settings:
    - Go to "File" > "Preferences" > "Settings" (Windows/Linux) or "Code" > "Preferences"

> "Settings" (macOS).

* + Search for Font:
    - In the search bar, type "Font" to find settings related to font size, family, and weight.
  + Modify Font Settings:
    - Adjust the font size, family, and weight to your liking. Changing File Icons:
  + Install Icon Themes:
    - Similar to color themes, you can install icon themes by opening the Command Palette with Ctrl+Shift+P or Cmd+Shift+P, then typing "Preferences: File Icon Theme".
  + Select and Apply an Icon Theme:
    - Browse through the available icon themes and select one that suits your preferences.

1. Advanced Customization

For more advanced customization, you can modify the settings in the settings.json file.

* + Open Settings (JSON):
    - Press Ctrl+Shift+P or Cmd+Shift+P and type "Preferences: Open Settings (JSON)". This will open the JSON configuration file where you can manually adjust various settings, including UI behavior, font customization, and more.

# Why might a developer prefer using a text editor over IDE? Provide specific scenarios when this might be a case.

## When and Why a Developer Might Prefer a Text Editor Over an IDE

* + **Minimalistic Interface for Small Scripts**:
    - For quick tasks or small scripts, a developer might opt for a text editor's minimalistic interface to avoid unnecessary features.
    - **Example**: Editing a single Python script or configuration file without needing debugging tools or project management features.

## Low Resource Consumption:

* + - On machines with limited resources, an IDE can be too resource-intensive, making a lightweight text editor a better choice.
    - **Example**: Using a text editor like Sublime Text or Notepad++ on an older laptop to avoid the sluggish performance often associated with heavier IDEs.

## Highly Customizable Environment:

* + - Developers who prefer to customize their tools extensively may find text editors more flexible and adaptable to their workflow.
    - **Example**: A developer using Vim or Emacs, which can be highly customized through plugins and configurations to perfectly suit their specific needs.

## Focused Environment for Simple Tasks:

* + - When working on simple tasks where the advanced features of an IDE might be unnecessary or even distracting, a text editor offers a more focused environment.
    - **Example**: Writing or editing documentation files, such as Markdown or plain text, where the extra features of an IDE aren't needed.

# Explain how the version control system like Git can be integrated over VS code. What are the benefits of such integration?

## Using Git in Visual Studio Code

1. **Install Git and Launch VS Code**:
   * Ensure Git is installed on your system. Then, open Visual Studio Code.

## Open Your Project Folder:

* + Navigate to and open your project folder in VS Code. If your project isn’t already under version control, you can initialize a Git repository by opening the terminal (Ctrl

+ ) and running:

git init

## Manage Your Git Repository:

* + Access Git features such as staging, committing, and viewing history by pressing Ctrl

+ Shift + G.

## Stage and Commit Changes:

* + To stage files, click the ‘+’ icon next to the files. Then, enter a commit message and click the checkmark to commit the changes.

## Branch Management:

* + Manage branches by clicking the branch name in the bottom-left corner of the VS Code window. From here, you can switch, create, or delete branches.

## Cloning and Adding Remotes:

* + To clone a repository, use the following command in the terminal:

bash

Copy code

git clone <repository\_url>

* + If you want to add a remote to an existing repository, use:

bash

Copy code

git remote add origin <repository\_url>

## Benefits of Git Integration in VS Code

* **Seamless Git Workflow**:
  + Integrated Git support in VS Code allows you to manage your repository, make commits, create branches, and handle merges directly within your development environment, streamlining your workflow.

## Visual Tools for Version Control:

* + VS Code provides intuitive visual tools for comparing changes, resolving merge conflicts, and reviewing the history of changes.

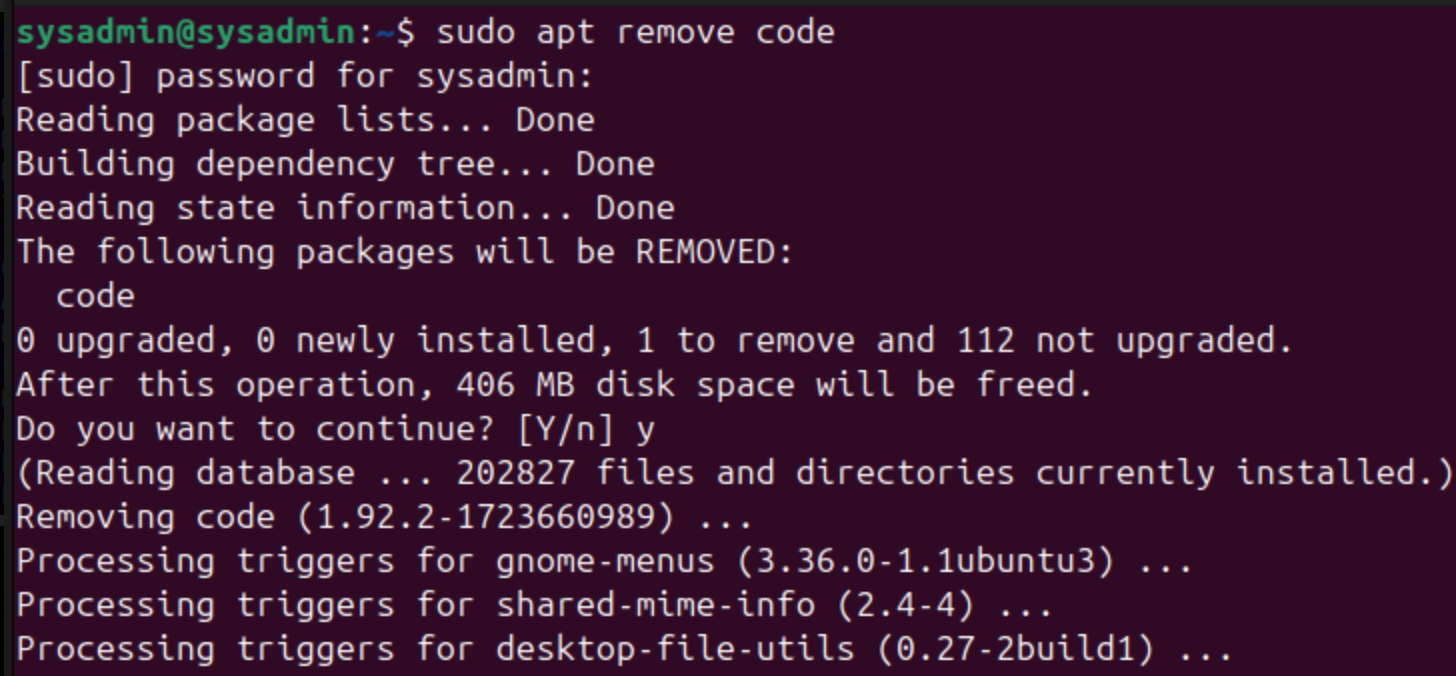
## Commit History and Blame Feature:

* + View commit history and use the "blame" feature to track who made specific changes, aiding in code reviews and ensuring accountability.

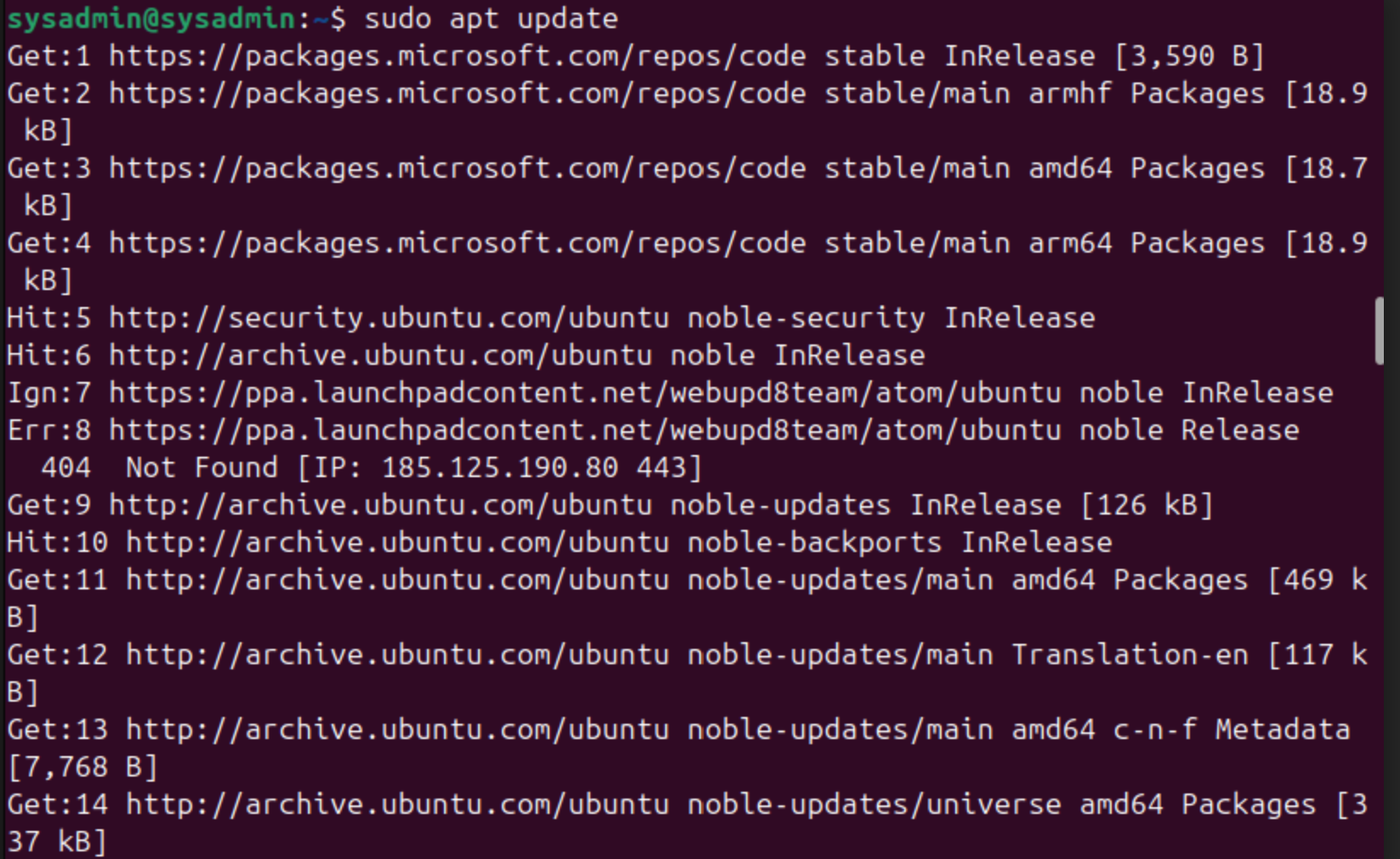
## Branch Management:

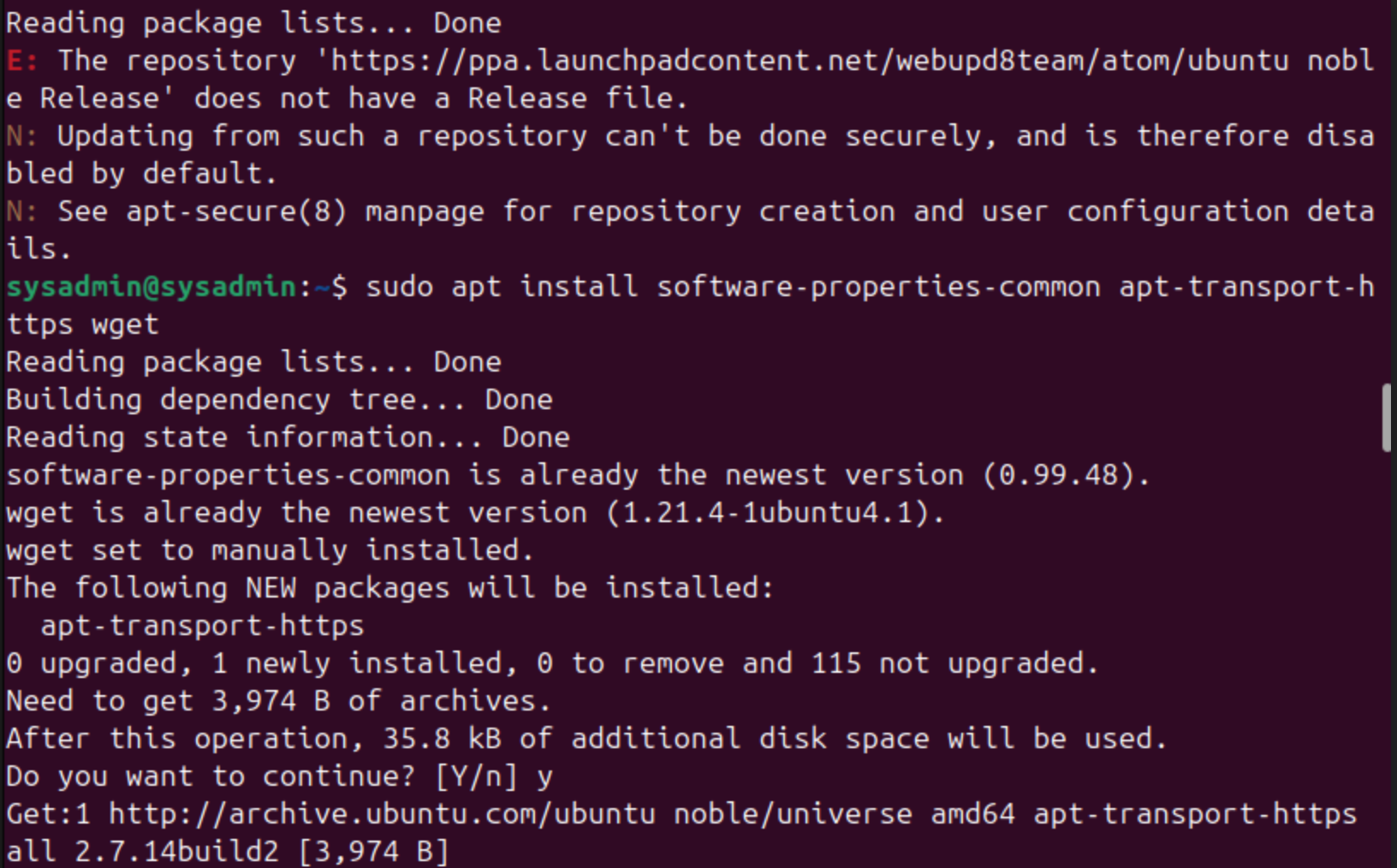
* + Easily switch, create, and manage branches within the editor, enabling smoother handling of multiple feature developments and hotfixes.

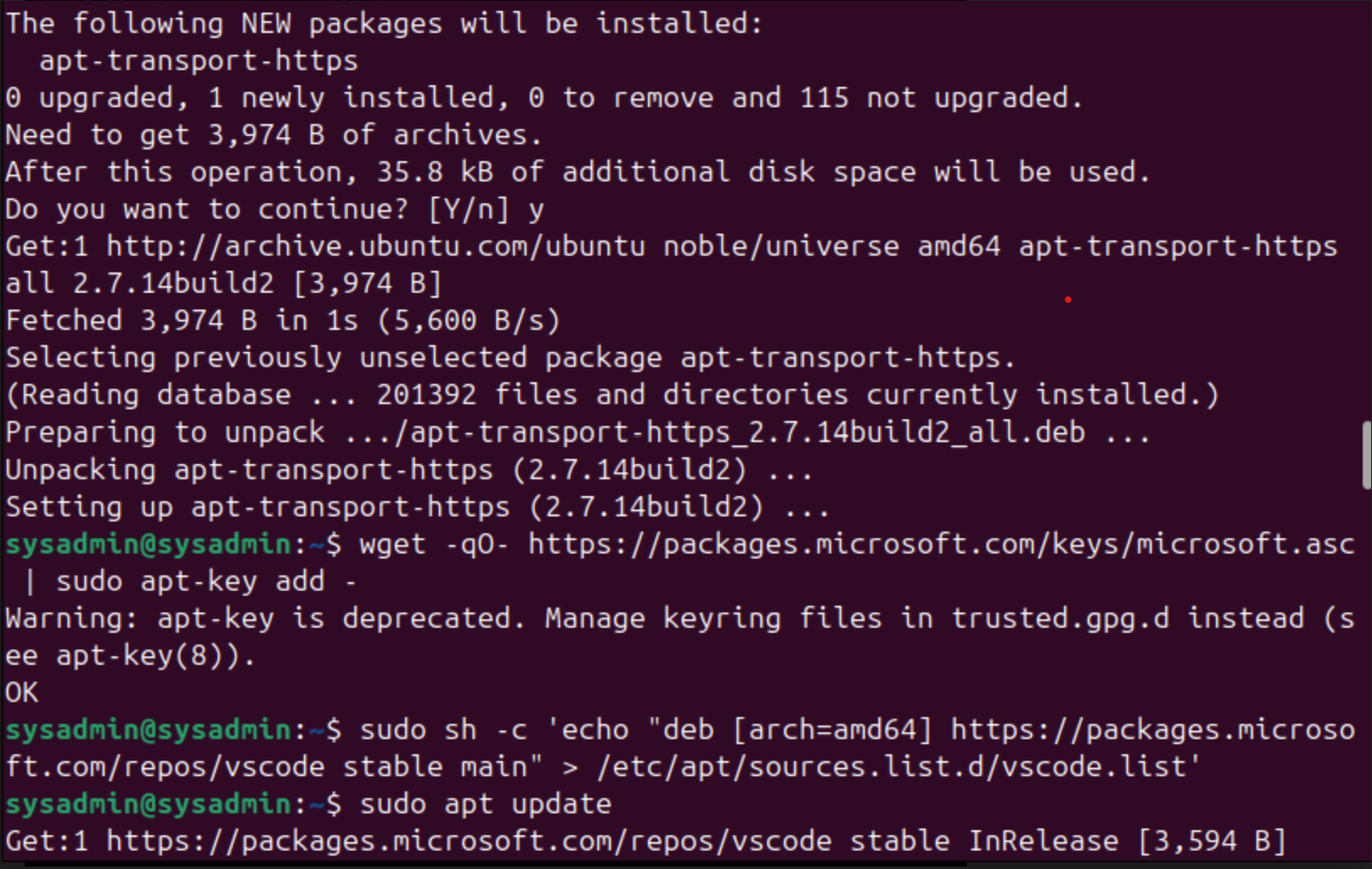
Uninstalling VS Code:

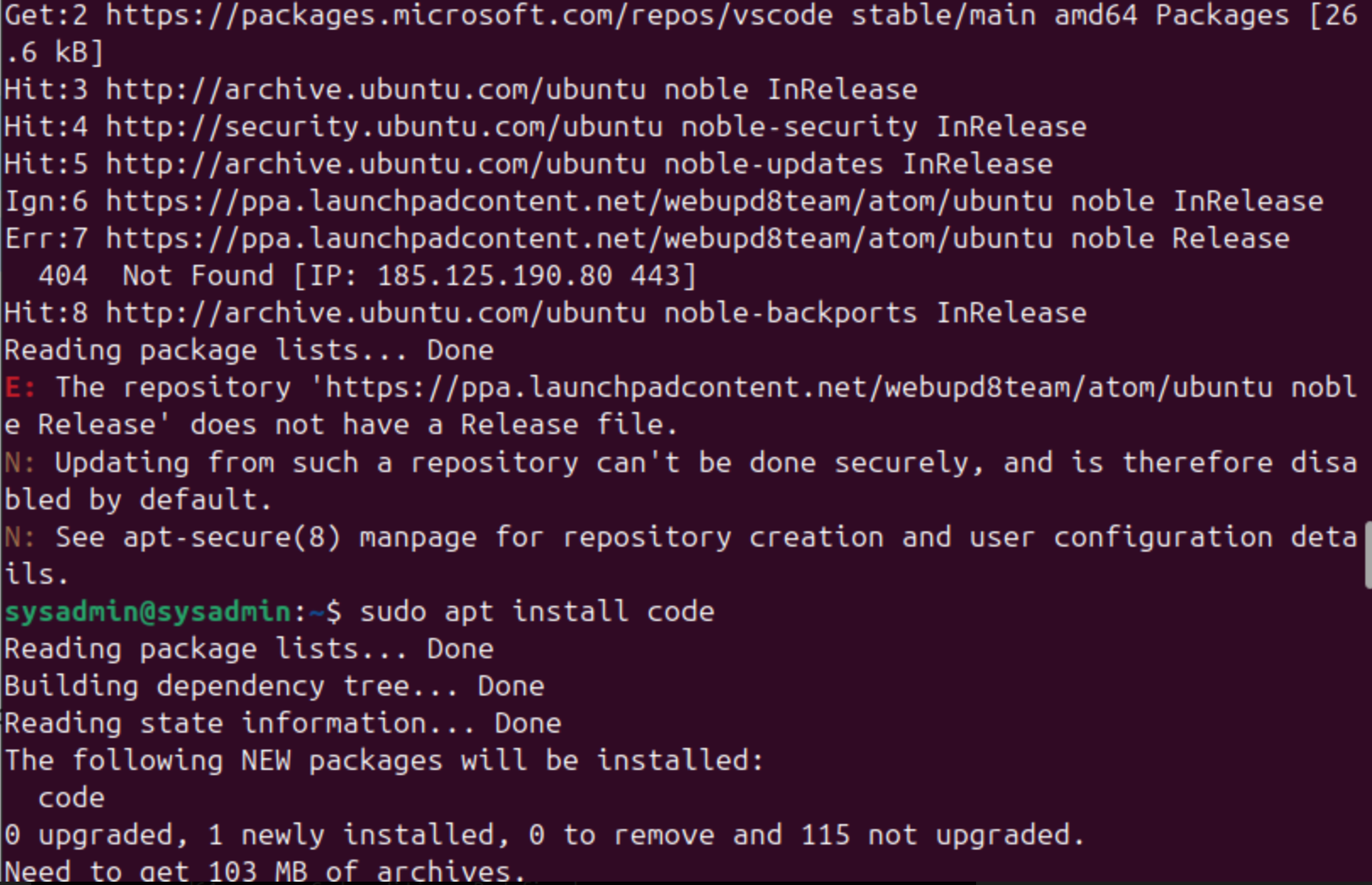


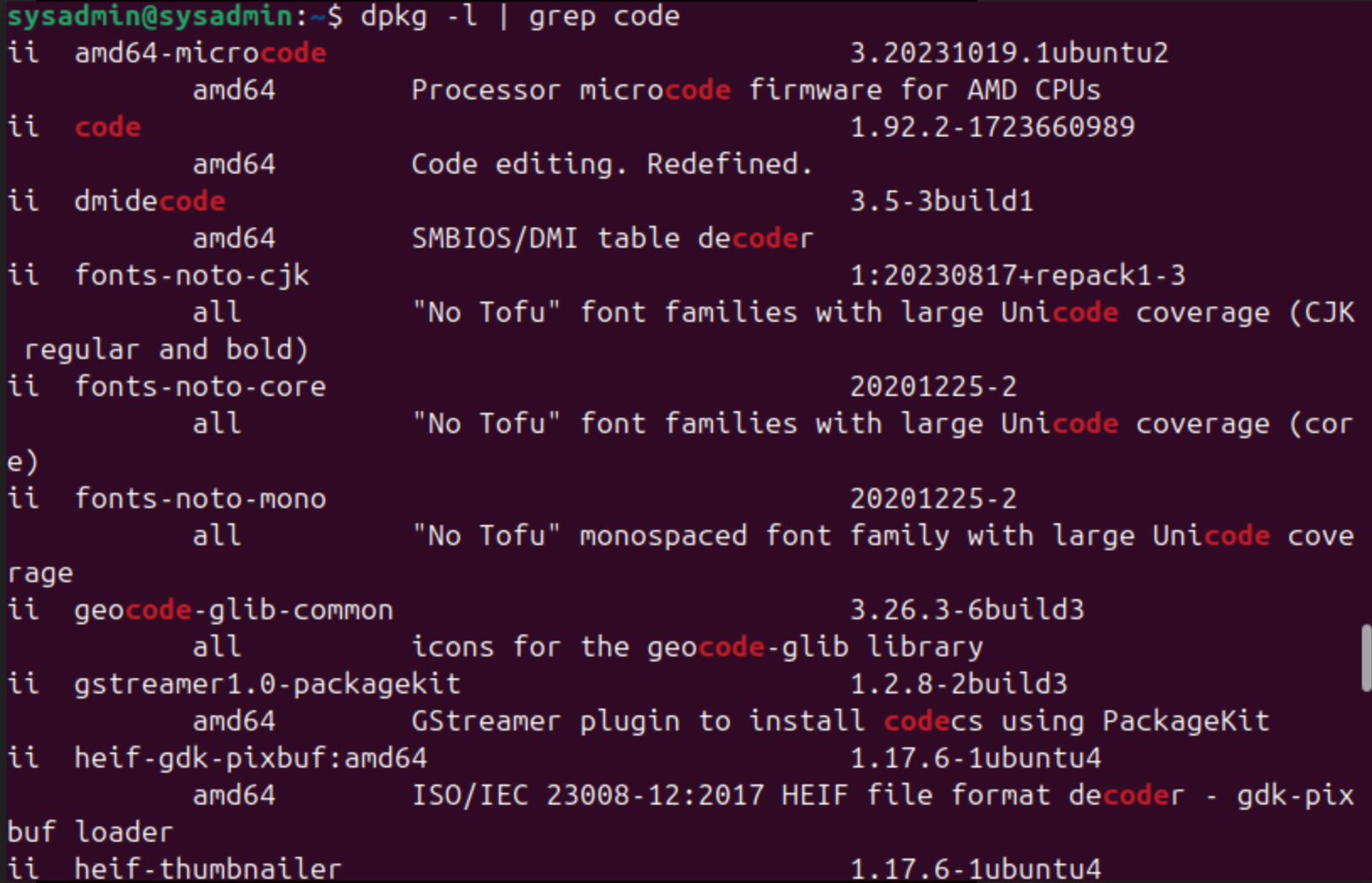
Installing VS Code:



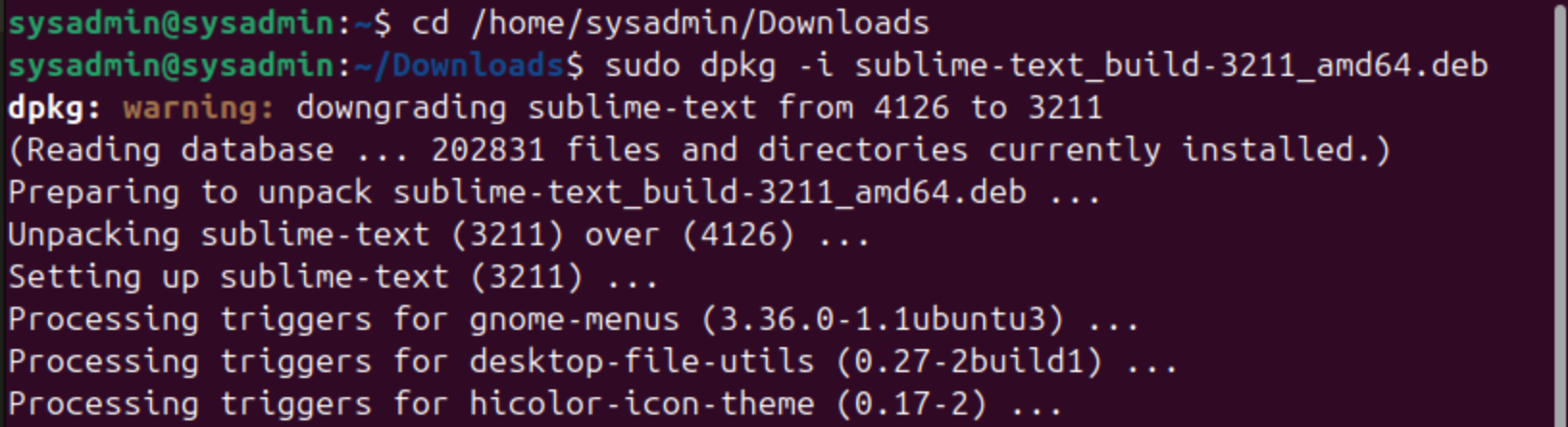




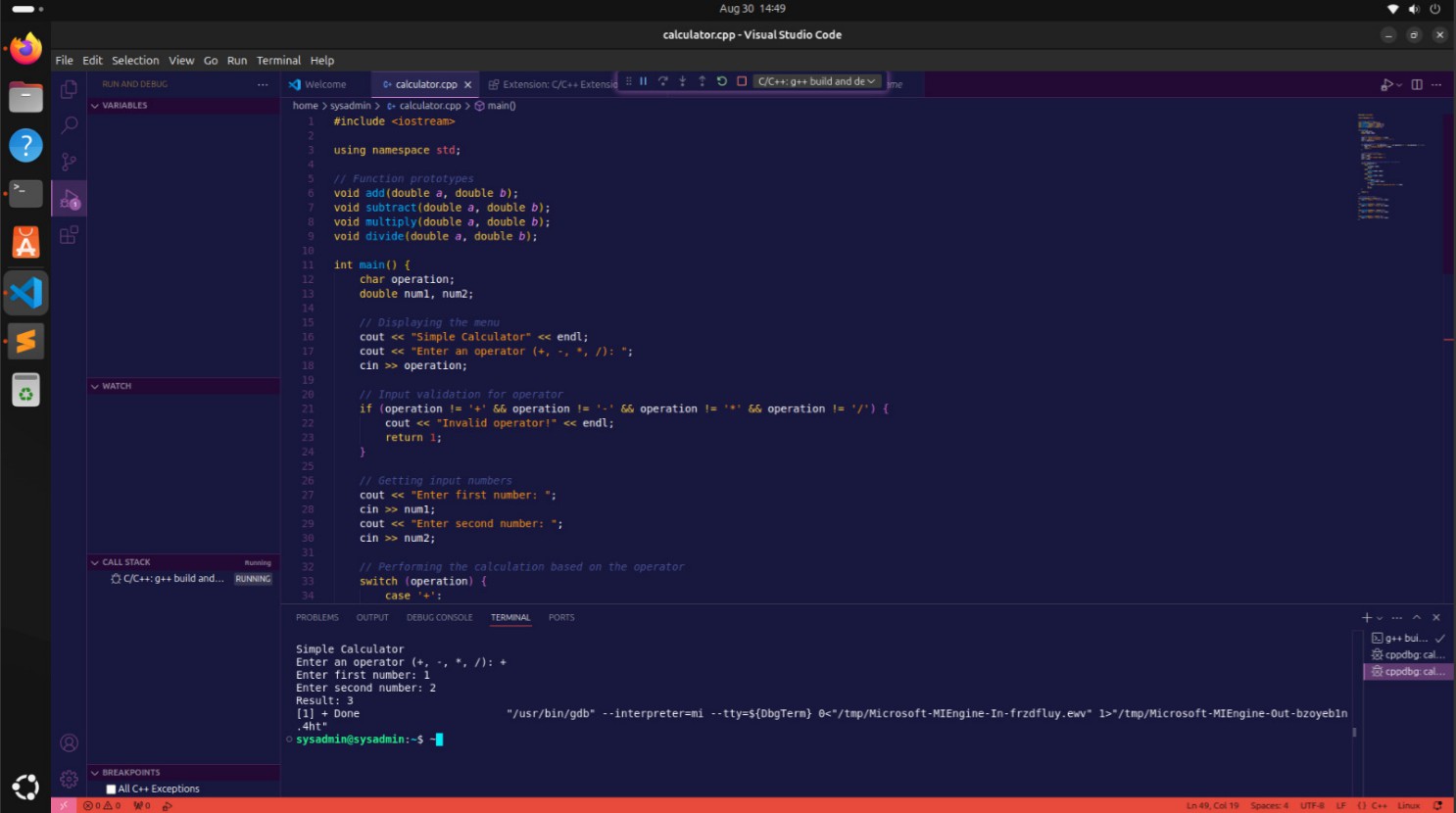


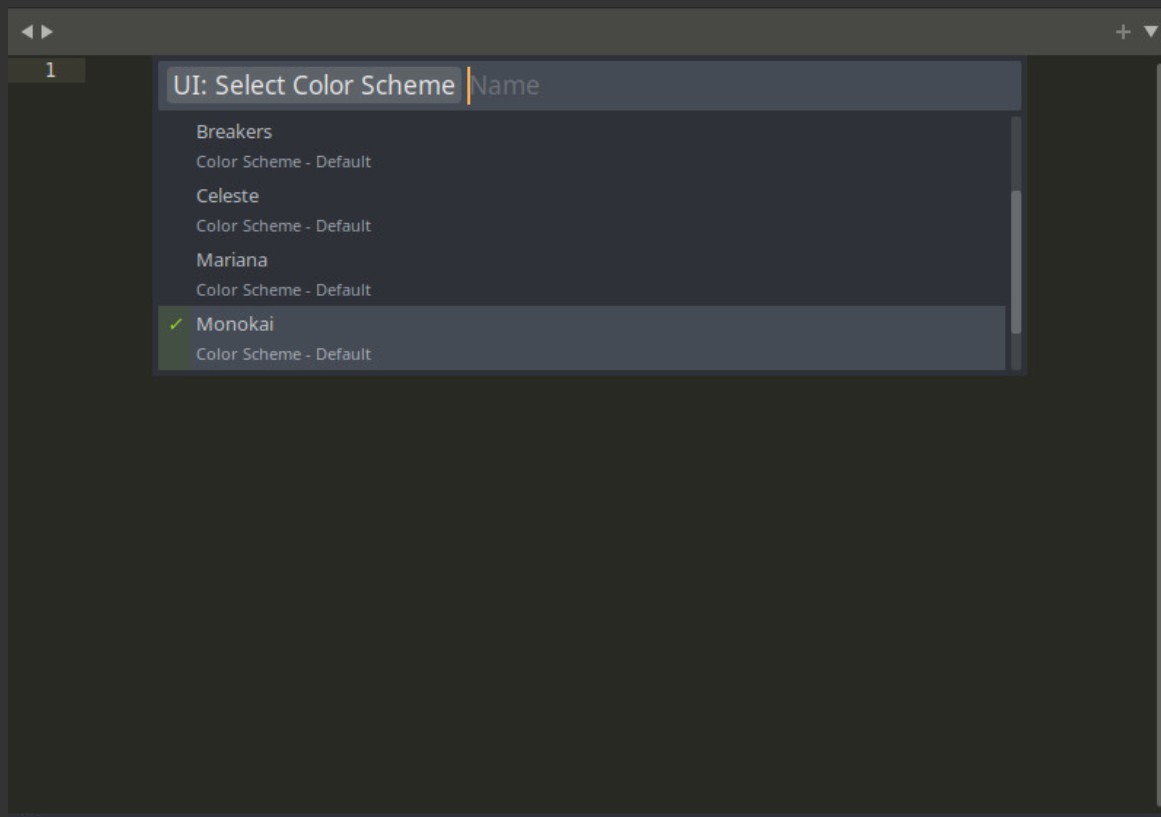


Installing Sublime Text:

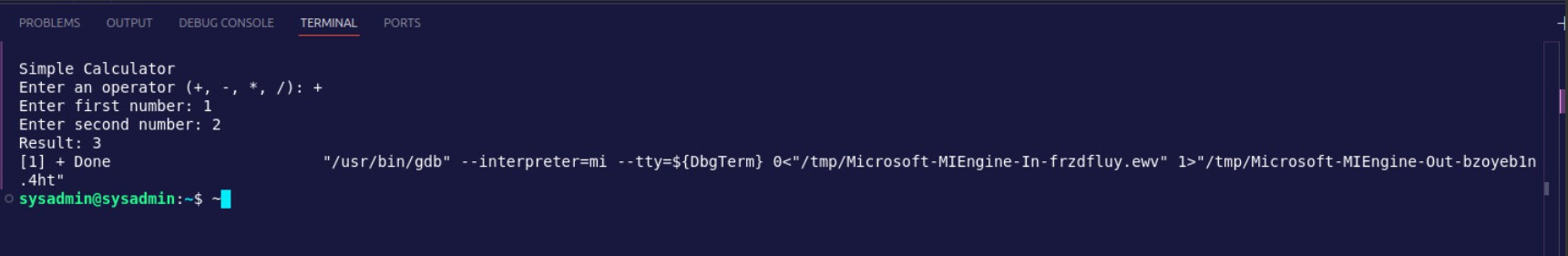


Customization in VS Code:

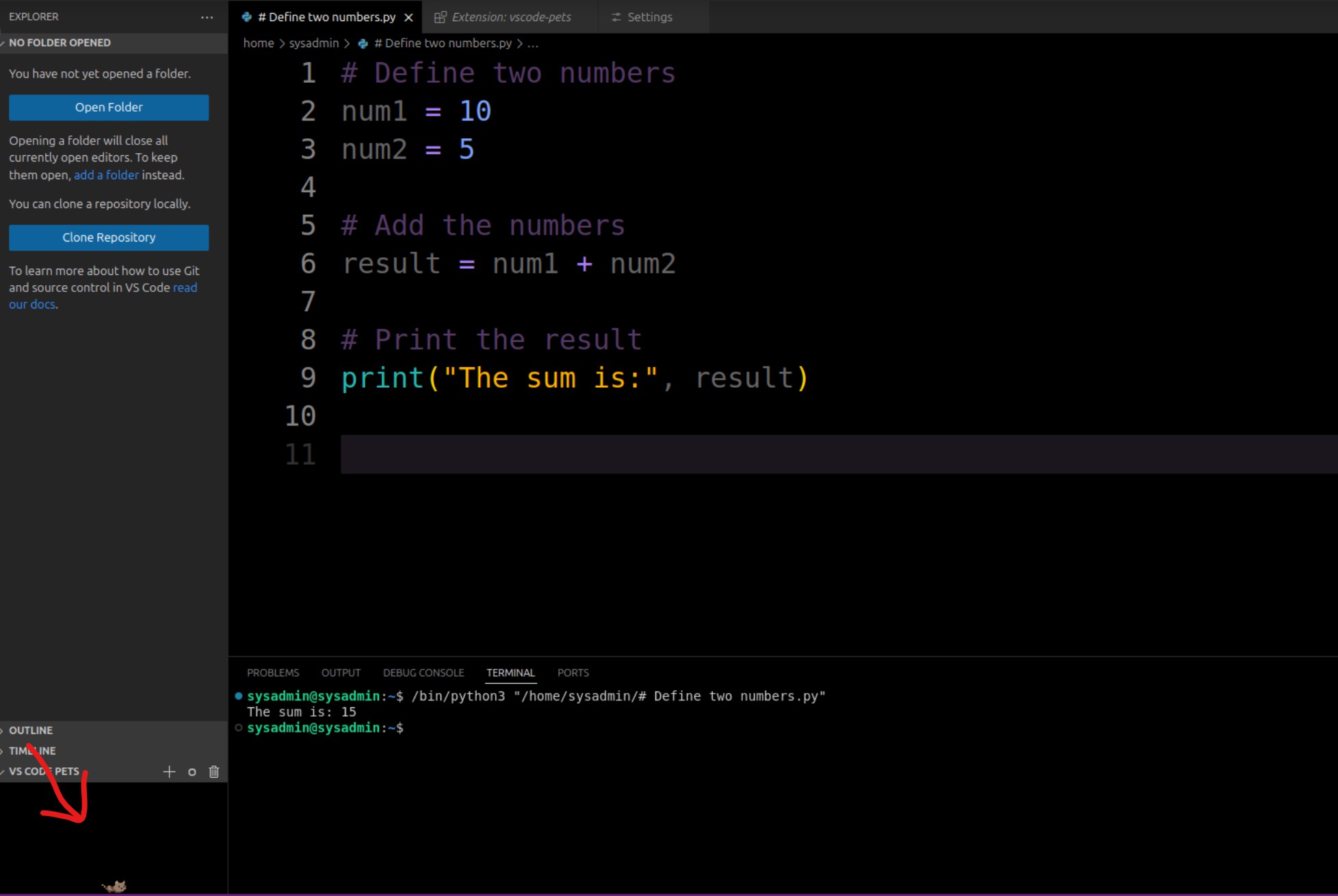




Program output



Customization in sublime text:



Using pet theme here

**Conclusion:** In conclusion, the introduction to Integrated Development Environments (IDEs) and text editors, using tools like VS Code and Sublime Text, highlights the significance of choosing the right development environment for your workflow. While IDEs offer comprehensive features that support the entire software development lifecycle, text editors provide a lightweight and customizable alternative for simpler tasks. Customizing your development environment, whether through themes, layouts, or extensions, allows for a more personalized and efficient coding experience, catering to both your needs and preferences.