**MARTIN MAINA**

**SOFTWARE ENGINEERING – ASSIGNMENT 5**

**Installation of VS Code:**

**Describe the steps to download and install Visual Studio Code on Windows 11 operating system. Include any prerequisites that might be needed.**

**Prerequisites:**

* Windows 11
* Administrator rights

**Steps:**

1. **Download VS Code**:
   * Go to [Visual Studio Code website](https://code.visualstudio.com/).
   * Click "Download for Windows".
2. **Run the Installer**:
   * Locate and double-click the downloaded file.
3. **Installation Process**:
   * Click "Next" on the setup window.
   * Accept the license agreement and click "Next".
   * Choose the installation location and click "Next".
   * Select additional tasks (create desktop icon, add to PATH, etc.) and click "Next".
   * Click "Install".
4. **Complete Installation**:
   * Click "Finish" to launch VS Code (if selected).

**After Installation:**

1. **Launch VS Code**:
   * Find it in the Start menu or use the desktop icon.
2. **Install Extensions**:
   * Click the Extensions icon or press Ctrl+Shift+X to add extensions.

**First-time Setup:**

After installing VS Code, here are the initial configurations and settings to adjust for an optimal coding environment:

1. **Settings Configuration**:
   * Open the settings by clicking the gear icon at the bottom left and selecting "Settings" or pressing Ctrl+,.
   * Adjust font size, theme, and tab size according to your preference.
   * Enable auto-save (File > Auto Save) to automatically save changes.
2. **Extensions**:
   * Open the Extensions view by clicking the Extensions icon in the Activity Bar or pressing Ctrl+Shift+X.
   * Install essential extensions based on your needs:
     + **Prettier**: Code formatter.
     + **ESLint**: Linting for JavaScript.
     + **Python**: Python support (if applicable).
     + **Live Server**: Launch a development local server with live reload.
     + **GitLens**: Supercharge the built-in Git capabilities.
     + **Debugger for Chrome**: Debug JavaScript code in the Chrome browser.
3. **Keyboard Shortcuts**:
   * Customize keyboard shortcuts by navigating to File > Preferences > Keyboard Shortcuts or pressing Ctrl+K Ctrl+S.
4. **Version Control**:
   * Set up Git by navigating to View > SCM or pressing Ctrl+Shift+G.
   * Ensure Git is installed and configure your user name and email:

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com"

1. **Terminal**:
   * Open the integrated terminal by clicking Terminal > New Terminal or pressing Ctrl+ ` (backtick).

**User Interface Overview:**

The main components of the VS Code user interface include the Activity Bar, Side Bar, Editor Group, and Status Bar.

1. **Activity Bar**:
   * **Location**: Left side of the window.
   * **Purpose**: Provides access to different views such as Explorer, Search, Source Control, Run & Debug, and Extensions.
   * **Customization**: Click the ellipsis (...) to hide or show specific views.
2. **Side Bar**:
   * **Location**: Next to the Activity Bar.
   * **Purpose**: Displays the content of the selected view from the Activity Bar. For example:
     + **Explorer**: Shows your project's files and folders.
     + **Search**: Allows you to search across files.
     + **Source Control**: Provides Git integration features.
     + **Extensions**: Allows you to manage extensions.
3. **Editor Group**:
   * **Location**: Central part of the window.
   * **Purpose**: Main area where you write and edit your code. You can split this area into multiple editors for side-by-side editing.
   * **Tabs**: Open files are displayed as tabs at the top of this area.
4. **Status Bar**:
   * **Location**: Bottom of the window.
   * **Purpose**: Displays information about the current state of the editor and the workspace, such as line and column numbers, current file type, Git branch, and errors or warnings.
   * **Customization**: Some extensions add information to the Status Bar.

**Command Palette:**

**What is the Command Palette in VS Code, and how can it be accessed? Provide examples of common tasks that can be performed using the Command Palette.**

**Command Palette in VS Code:**

The Command Palette in Visual Studio Code is a powerful tool that allows you to access and execute various commands quickly. It provides a quick way to perform almost any action in VS Code without navigating through menus.

**How to Access the Command Palette:**

* Press Ctrl+Shift+P (Windows/Linux) or Cmd+Shift+P (Mac).
* Alternatively, you can click on the View menu and select "Command Palette".

**Examples of Common Tasks Using the Command Palette:**

1. **Opening Files**:
   * Type > Open File to quickly open a file by name.
2. **Navigating to Settings**:
   * Type > Preferences: Open Settings to open the settings editor.
3. **Installing Extensions**:
   * Type > Extensions: Install Extensions to search for and install new extensions.
4. **Git Commands**:
   * Type > Git: Clone to clone a repository.
   * Type > Git: Commit to commit changes.
   * Type > Git: Push to push changes to a remote repository.
5. **Formatting Code**:
   * Type > Format Document to format the current document using the installed formatter.
6. **Running Tasks**:
   * Type > Run Task to run a pre-configured task (e.g., build, test).
7. **Toggling Sidebar Views**:
   * Type > View: Toggle Side Bar to show or hide the Side Bar.
8. **Changing Color Themes**:
   * Type > Preferences: Color Theme to switch between different color themes.
9. **Opening Integrated Terminal**:
   * Type > Terminal: Create New Integrated Terminal to open a new terminal window.
10. **Running and Debugging Code**:
    * Type > Debug: Start Debugging to start debugging the current file or project.
    * Type > Debug: Run to run the current file without debugging.

**Extensions in VS Code:**

**Discuss the role of extensions in VS Code. How can users find, install, and manage extensions? Provide examples of essential extensions for web development.**

**Role of Extensions in VS Code:**

Extensions in Visual Studio Code enhance and customize the editor's functionality. They allow users to add support for different programming languages, debuggers, and tools, integrate with services, and customize the appearance and behavior of the editor to fit their needs.

**Finding, Installing, and Managing Extensions:**

1. **Finding Extensions**:
   * **Via the Extensions View**: Click on the Extensions icon in the Activity Bar on the side of the window or press Ctrl+Shift+X to open the Extensions view.
   * **Marketplace**: Browse the [Visual Studio Code Marketplace](https://marketplace.visualstudio.com/vscode) to explore available extensions.
2. **Installing Extensions**:
   * In the Extensions view, search for the extension you want by typing its name or functionality in the search bar.
   * Click the "Install" button on the extension you wish to add.
   * Once installed, some extensions might require a restart of VS Code to activate.
3. **Managing Extensions**:
   * **View Installed Extensions**: Open the Extensions view and click on "Installed" to see all the extensions you have installed.
   * **Disable or Enable Extensions**: Click on the gear icon next to the extension and select "Disable" or "Enable".
   * **Uninstall Extensions**: Click on the gear icon next to the extension and select "Uninstall".
   * **Update Extensions**: If an update is available, a reload button or update indicator will appear. Click it to update the extension.

**Essential Extensions for Web Development:**

1. **Prettier - Code Formatter**:
   * Automatically formats your code to ensure consistency.

**Live Server**:

* + Launches a local development server with live reload feature for static and dynamic pages.

**Integrated Terminal:**

**Describe how to open and use the integrated terminal in VS Code. What are the advantages of using the integrated terminal compared to an external terminal?**

**How to Open the Integrated Terminal:**

1. **Using the Menu**:
   * Navigate to Terminal > New Terminal from the top menu.
2. **Keyboard Shortcut**:
   * Press Ctrl+ (backtick) on Windows/Linux orCmd+ on Mac.
3. **Command Palette**:
   * Open the Command Palette (Ctrl+Shift+P or Cmd+Shift+P) and type > Terminal: Create New Integrated Terminal, then select it.

**Using the Integrated Terminal:**

1. **Creating a New Terminal**:
   * You can create multiple terminals by clicking the plus (+) icon in the terminal panel.
2. **Switching Between Terminals**:
   * If you have multiple terminals open, you can switch between them by clicking on the terminal tabs at the top of the terminal panel.
3. **Running Commands**:
   * Type your commands as you would in any terminal and press Enter to execute them.
4. **Splitting Terminals**:
   * You can split the terminal view by clicking the split terminal icon in the terminal panel or using the Command Palette (> Terminal: Split Terminal).
5. **Closing Terminals**:
   * Close a terminal by clicking the trash can icon in the terminal panel or typing exit and pressing Enter.

**Advantages of Using the Integrated Terminal:**

1. **Convenience**:
   * The integrated terminal allows you to stay within VS Code, avoiding the need to switch between the editor and an external terminal. This saves time and maintains focus.
2. **Project Context**:
   * The terminal starts in the workspace's root directory by default, ensuring commands are run in the correct context without needing to navigate to the project directory manually.
3. **Multiple Terminals**:
   * You can open multiple terminals and run different tasks simultaneously, easily switching between them within the same window.
4. **Theming and Customization**:
   * The integrated terminal can match your VS Code theme, providing a cohesive visual experience. You can customize the appearance and behavior of the terminal in the settings.
5. **Task Integration**:
   * Integrate terminal commands with VS Code tasks to automate build, test, and deployment workflows directly from the editor.
6. **Shortcut Accessibility**:
   * Quickly open, close, and manage terminals using keyboard shortcuts, streamlining repetitive tasks.
7. **Unified Interface**:
   * Having everything in one interface simplifies development, debugging, and testing processes. You can view terminal output alongside your code without leaving the editor.

**File and Folder Management:**

**Explain how to create, open, and manage files and folders in VS Code. How can users navigate between different files and directories efficiently?**

**File and Folder Management in VS Code**

Managing files and folders efficiently is crucial for a streamlined development workflow. Here’s how you can create, open, and manage files and folders in Visual Studio Code:

**Creating Files and Folders:**

1. **Using the Explorer View**:
   * **Open the Explorer**: Click the Explorer icon in the Activity Bar on the side of the window or press Ctrl+Shift+E.
   * **Create a New File**: Right-click on a folder in the Explorer view and select New File, or press Ctrl+N to create a new untitled file.
   * **Create a New Folder**: Right-click on a folder in the Explorer view and select New Folder.
2. **Using Keyboard Shortcuts**:
   * **New File**: Press Ctrl+N to create a new untitled file.
   * **Save As**: After creating a new untitled file, press Ctrl+S to save it with a specific name and location.

**Opening Files and Folders:**

1. **Open a File**:
   * **File Menu**: Navigate to File > Open File... and select the file you want to open.
   * **Keyboard Shortcut**: Press Ctrl+O to open the file dialog and select a file.
2. **Open a Folder**:
   * **File Menu**: Navigate to File > Open Folder... and select the folder you want to open.
   * **Keyboard Shortcut**: Press Ctrl+K Ctrl+O to open the folder dialog and select a folder.
3. **Drag and Drop**:
   * Drag a file or folder from your file explorer (Windows Explorer or Finder on Mac) and drop it into the VS Code window.

**Managing Files and Folders:**

1. **Rename**:
   * Right-click on a file or folder in the Explorer view and select Rename, or click the file/folder name and press F2.
2. **Delete**:
   * Right-click on a file or folder in the Explorer view and select Delete, or click the file/folder name and press Delete.
3. **Move**:
   * Drag and drop files or folders within the Explorer view to move them.

**Navigating Between Files and Directories Efficiently:**

1. **File Tabs**:
   * Open files appear as tabs at the top of the editor. Click on the tabs to switch between files.
2. **Quick Open**:
   * Press Ctrl+P (Windows/Linux) or Cmd+P (Mac) to open the Quick Open dialog. Type the name of the file you want to open and select it from the list.
3. **Go to File**:
   * Press Ctrl+T to quickly navigate to any file in the workspace.
4. **Breadcrumbs**:
   * Enable Breadcrumbs by clicking the View > Show Breadcrumbs or pressing Ctrl+Shift+. (dot). This provides a breadcrumb trail of the current file's location within the folder structure.
5. **Explorer View**:
   * Use the Explorer view to browse through your project’s files and folders. Expand and collapse folders to navigate through the directory structure.
6. **File Search**:
   * Press Ctrl+Shift+F to open the search panel. Type the name or part of the content of the file you’re looking for. The search results will display matching files and their locations.
7. **Sidebar Navigation**:
   * Use the Activity Bar icons (Explorer, Search, Source Control, etc.) to switch between different views and tools within the Sidebar.

**Settings and Preferences:**

**Where can users find and customize settings in VS Code? Provide examples of how to change the theme, font size, and keybindings.**

**Accessing Settings:**

1. **Settings UI**:
   * Open the settings by clicking the gear icon in the lower left corner of the window and selecting Settings, or by pressing Ctrl+,.
2. **Settings JSON**:
   * For advanced users who prefer to edit the settings directly in JSON format, click the {} icon in the top right corner of the Settings UI.

**Changing the Theme:**

1. **Using the Settings UI**:
   * Open the settings (Ctrl+,).
   * In the search bar at the top, type "theme".
   * Select Color Theme under Preferences.
   * Choose your preferred theme from the list.
2. **Using the Command Palette**:
   * Open the Command Palette (Ctrl+Shift+P).
   * Type > Preferences: Color Theme and select it.
   * Choose your preferred theme from the list.

**Changing the Font Size:**

1. **Using the Settings UI**:
   * Open the settings (Ctrl+,).
   * In the search bar at the top, type "font size".
   * Adjust the Editor: Font Size setting to your preferred size.
2. **Using Settings JSON**:
   * Open the settings in JSON format (Ctrl+Shift+P, type > Preferences: Open Settings (JSON) and select it).
   * Add or modify the following line:

"editor.fontSize": 16

* + Replace 16 with your desired font size.

**Changing Keybindings:**

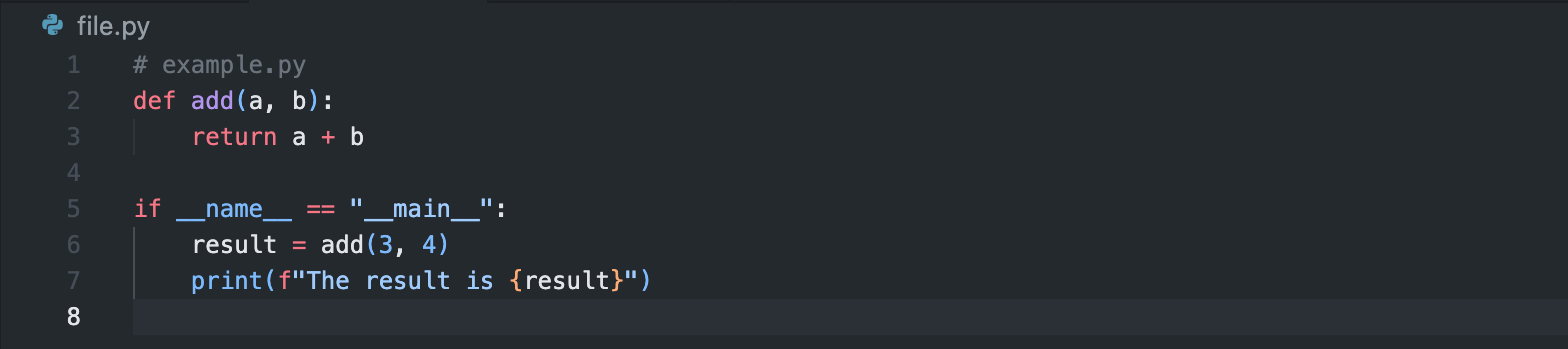
1. **Using the Keyboard Shortcuts UI**:
   * Open the Keyboard Shortcuts settings by clicking the gear icon in the lower left corner of the window and selecting Keyboard Shortcuts, or by pressing Ctrl+K Ctrl+S.
   * Use the search bar to find the command you want to rebind.
   * Click the pencil icon next to the command and press the new key combination.

**DEBUGGING:**

**Key Debugging Features in VS Code:**

1. **Breakpoints**:
   * **Set/Remove Breakpoints**: Click in the gutter next to the line numbers or press F9.
   * **Conditional Breakpoints**: Right-click on a breakpoint and set a condition for it to trigger.
2. **Watch**:
   * **Watch Expressions**: Add variables or expressions to the Watch panel to monitor their values as you step through the code.
3. **Call Stack**:
   * **View Call Stack**: Inspect the call stack to understand the sequence of function calls leading to the current point of execution.
4. **Variables**:
   * **Inspect Variables**: View and inspect variable values in the Variables panel.
5. **Debug Console**:
   * **Evaluate Expressions**: Use the Debug Console to execute code and evaluate expressions in the current debugging context.
6. **Step Controls**:
   * **Step Over (F10)**: Execute the next line of code but don't step into functions.
   * **Step Into (F11)**: Step into the function calls.
   * **Step Out (Shift+F11)**: Step out of the current function.
   * **Continue (F5)**: Continue execution until the next breakpoint.
7. **Inline Values**:
   * **View Inline Values**: See variable values directly in the editor as you debug.

**Debugging Session:**

****

1. **Set a Breakpoint**:
   * Set a breakpoint on the line result = add(3, 4).
2. **Start Debugging**:
   * Select Python: Current File from the configuration drop-down and press F5.
3. **Hit Breakpoint**:
   * The debugger will stop at the breakpoint. The Variables panel will show the current values of variables.
4. **Step Over**:
   * Press F10 to step over the result = add(3, 4) line and observe the result value in the Variables panel.
5. **Inspect Variables**:
   * Check the values of a and b in the Variables panel or add them to the Watch panel.
6. **Evaluate in Debug Console**:
   * Use the Debug Console to run add(5, 6) and see the result without modifying the code.

**Using Source Control:**

**How can users integrate Git with VS Code for version control? Describe the process of initializing a repository, making commits, and pushing changes to GitHub.**

**Using Source Control (Git) in VS Code**

Visual Studio Code provides seamless integration with Git, making it easy to manage version control directly from the editor. Here's how you can initialize a repository, make commits, and push changes to GitHub using VS Code:

**Initializing a Repository:**

1. **Open Your Project**:
   * Ensure your project folder is open in VS Code.
2. **Open the Source Control View**:
   * Click on the Source Control icon in the Activity Bar on the side of the window (looks like a branch), or press Ctrl+Shift+G.
3. **Initialize Git Repository**:
   * Click Initialize Repository or Initialize Git Repository... in the Source Control view.
   * Alternatively, open the integrated terminal (`Ctrl+``) and navigate to your project directory. Run:

git init

**Making Commits:**

1. **Stage Changes**:
   * In the Source Control view, you'll see a list of changed files. Click the + button next to each file you want to include in the commit to stage them.
2. **Commit Changes**:
   * Enter a commit message in the text box at the top of the Source Control view.
   * Click the check mark (✓) to commit the changes.

**Pushing Changes to GitHub:**

1. **Link Your Repository to GitHub**:
   * Make sure you have a GitHub repository created where you want to push your local repository.
   * Obtain the HTTPS or SSH URL of your GitHub repository.
2. **Add Remote Repository**:
   * Open the integrated terminal (`Ctrl+``) and add the remote repository URL as follows:

git remote add origin <remote\_repository\_url>

* + Alternatively, use the command palette (Ctrl+Shift+P) and search for Git: Add Remote.

1. **Push Commits to GitHub**:
   * After committing your changes locally, click the ellipsis (...) next to the commit message in the Source Control view.
   * Choose Push to push your committed changes to the remote GitHub repository.

**Key Git Features in VS Code:**

1. **Branch Management**:
   * Create, switch, and delete branches directly from the Source Control view.
2. **Diff Viewer**:
   * View line-by-line differences between files to see what has changed before committing.
3. **Stash**:
   * Temporarily store changes that are not ready to be committed using the Stash feature in the Source Control view.
4. **History**:
   * View the commit history of your repository and revert to previous versions if needed.
5. **Pull**:
   * Pull changes from the remote repository to update your local repository with the latest changes.

**Example Workflow:**

1. **Initialize Repository**:
   * Open your project folder in VS Code and initialize Git as described.
2. **Make Changes**:
   * Modify existing files or create new ones.
3. **Stage and Commit**:
   * Stage the changes and commit them with a descriptive message.
4. **Push to GitHub**:
   * Add the remote GitHub repository URL and push your changes to GitHub.

**Reference**:

Decan, A., Mens, T., Mazrae, P. R., & Golzadeh, M. (2022, October). On the use of github actions in software development repositories. In *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)* (pp. 235-245). IEEE.

Harper, R. (1997). *U.S. Patent No. 5,631,088*. Washington, DC: U.S. Patent and Trademark Office.

Stone, D., Jarrett, C., Woodroffe, M., & Minocha, S. (2005). *User interface design and evaluation*. Elsevier.

Tanaka, J. (2011). *U.S. Patent Application No. 13/146,743*.

Brooks, G., Hansen, G. J., & Simmons, S. (1992). A new approach to debugging optimized code. *ACM SIGPLAN Notices*, *27*(7), 1-11.