Paper ID #44570

Visual Studio Code in Introductory Computer Science Course: An Experience Report

Dr. Jialiang Tan, Lehigh University

Dr. Jialiang Tan is a Teaching Assistant Professor in the Department of Computer Science and Engineering at Lehigh University. Her research lies in high-performance computing, program/software analysis, and CS education research.

Dr. Yu Chen, Independent Researcher

Dr. Yu Chen obtained his Ph.D. from the Department of Computer Science at William and Mary in 2023. His research lies in machine learning systems, with a focus on building profiling tools and performing system-algorithm co-design to optimize machine learning applications

Dr. Shuyin Jiao, North Carolina State University

Dr. Shuyin Jiao is an Assistant Teaching Professor in the Department of Computer Science at North Carolina State University. With a primary emphasis on computing education in college, she specializes in teaching introductory programming courses and critical path computer science courses. Her scholarly pursuits are centered on computing education and program analysis, with a dedication to fostering a dynamic and innovative learning environment in the field of computer science education.

Visual Studio Code in Introductory Computer Science Course: An Experience Report

Jialiang Tan, Department of Computer Science and Engineering, Lehigh University
Yu Chen[1], Independent Researcher
Shuyin Jiao, Department of Computer Science, North Carolina State University

Abstract

Involving integrated development environments (IDEs) in introductory-level (CS1) programming courses is critical. However, it is difficult for instructors to find a suitable IDE that is beginner-friendly and offers robust functionality. In this paper, we share our experience of implementing Visual Studio Code (VS Code) in a CS1 programming course. We describe our motivation for selecting VS Code and our approach to introducing it to engineering students. To assist students with diverse programming backgrounds, we provide comprehensive guidance with hierarchical indexing. By seamlessly integrating VS Code, known as a rich text editor, with a selection of extensions, our aim is to streamline the learning process for students by enabling it to function as an IDE. We perform an experimental evaluation of students' programming experience of using VS Code and validate the VS Code together with guidance as a promising solution for CS1 programming courses.

1. Introduction

Integrated Development Environments (IDEs) play an important role in learning a programming language. IDEs offer programmers extensive development abilities. They understand language syntax and provide features such as build automation, code linting, testing, and debugging, which accelerate and simplify the coding process. Through the help of IDEs, students benefit from efficient programming, testing, and debugging. Students can further develop better coding habits and flatten the learning curve of a new language. As a result, more and more instructors start involving IDEs in introductory-level courses such as PyCharm [43], Eclipse [42], and others, to enhance student's coding experiences. In addition to traditional IDEs, many advanced code editors like VS Code and Atom [3], can form an IDE-like environment through the

[1] This work was done when Yu was a Ph.D. student at William and Mary.

integration of various extensions. For simplicity, we refer to both traditional IDEs and advanced code editors as IDEs in this paper.

However, not all IDEs are suitable for introductory-level programming courses, making the selection of an appropriate IDE challenging for instructors. Professional IDEs [11, 12, 39] offer integrated programming environments and many powerful features, but they often lack support for educational purposes. Using them effectively may require advanced knowledge and a significant time investment. For students new to programming, mastering these IDEs while learning a new programming language can be challenging, as they must spend considerable time on software installation and environment setup. On the other hand, education-focused code editors are easy to install and use but are rarely utilized in professional software development cycles. This creates a significant gap for students transitioning from college to industry [33, 34], potentially impacting their future careers.

Thus, there is an urgent need for an appropriate IDE for introductory-level programming classes, which not only provides sufficient education-related features but is also widely used among professional software engineers. We aim to use an IDE that meets four criteria: 1) cross-platform support, allowing students to install and use it identically on different operating systems and hardware; 2) ease to use, enabling students to code effectively with a simple and concise interface; 3) extensive functionalities, including code compilation, project organization, and support for multiple languages; 4) mainstream adoption among professional software engineers, bridging the gap between classes and future careers.

We have identified Microsoft Visual Studio Code (VS Code) as the preferred option. By integrating VS Code with a selection of extensions, it becomes an IDE that incorporates all four aforementioned features. Additionally, VS Code has been widely adopted in many advanced courses in our department, including operating systems, compiler constructions, computer networks, and others. However, it has not received significant attention to CS1 courses. Furthermore, based on our study of 20 computer science departments, none of them specify VS Code as the default IDE in introductory programming courses. Moreover, there is a lack of comprehensive guidance on VS Code for educators and students. Given that students come from diverse backgrounds, with varying majors, learning speeds, and familiarity with different operating systems, comprehensive guidance is essential to facilitate the efforts of both students and educators.

This paper presents the first experience report on the use of VS Code in an introductory (CS1) Python course. We have created the first comprehensive guidance for VS Code, tailored for both students and educators by gathering plugins (i.e., VS Code extensions) for Python education and incorporating various user experiences. The guidance is highly modularized to cater to students with different levels of programming experience. Students who are experienced programmers can jump to the sections they need to learn and skip those they already know, students who are beginner programmers can follow the guidance step-by-step, while those encountering errors can learn how to identify and resolve them. With the help of the guidance, students quickly acquire the necessary knowledge of the programming environment in the class, and transform VS Code into a desired IDE that perfectly complements programming development. We have successfully integrated VS Code and our guidance into the introductory-level Python course in our department. To evaluate the effectiveness of our efforts, we conducted a survey. The results indicate that students highly value the use of VS Code and our guidance. Students with industry internship experiences particularly acknowledge the usefulness of learning Python with VS Code over other education-oriented IDEs, as it is widely used in the industry [40].

Contributions. We make the following four contributions:

- We point out the importance of selecting a proper IDE for introductory-level courses.
- We identify Visual Studio Code, integrated with extensions, as a desired IDE. We investigate it from four aspects and analyze its practicality for introductory-level Python courses.
- We propose the first comprehensive guidance with hierarchical indexing, to guide students with diverse backgrounds.
- We report our experiences of using VS Code in an introductory-level programming course, demonstrating its effectiveness as a satisfactory IDE for the introductory-level Python course. We also validate the value and necessity of the guidance.

Organization. Section 2 reviews the background and related work. Section 3 shows our investigation of VS Code. Section 4 describes the guidance and support we offer to students. Section 5 shows some experimental studies. Section 6 provides the discussion. Section 7 presents conclusions and future work.

2. Background and Related Work

2.1. Python in Education

Python has become the most prominent language in college, it tops the list of the most-taught programming languages, especially in introductory computer science courses, it is mainly due to three facts: *Python is beginner friendly*, it has a simplified syntax with an emphasis on natural language while most programming languages have complex rules, thus it is perfect for beginners to learn and practice [9, 13, 23, 29]. *Python is a powerful tool*, it is applied in many areas like machine learning and big data, especially effective and essential in scientific computing and data analysis [1, 5, 6, 19, 20, 22, 31, 35]. Students with diverse backgrounds use Python for different purposes, e.g. analyzing the market in finance, simulating protein structure in biology. Students who pursue a formal education in computer science or computer science-related majors are extremely likely to continue using Python throughout their careers. *Python is popular*, according to the PYPL index [25] and Stackoverflow Developer Survey 2021 [30], Python is the most popular and the fastest-growing programming language (10.4%) in the world as of June 2022.

2.2. Pedagogical Approaches

Programming language plays an important role in computing education. Many systematic studies explore the teaching in introductory-level programming classes [14, 21, 27, 28]. Through our observation, a tremendous amount of research has been done on how the programming environment can improve students' coding experience in recent years, especially the programming environment for Java and Python, which are the most taught entry-level programming languages in college. The selection of the programming environment is typically decided by instructors. Pedagogical tools and Intelligent Tutoring Systems (ITS) [8, 18, 32, 37, 38] have been designed specifically for programming novices. Also, some works report the study of using industry-level programming environments such as Jupyter Notebooks [2, 4, 10, 36] or Eclipse extension [26], their experience shows how a great IDE improves students' engagement and performance.

2.3. IDEs in Education

Integrated Development Environments (IDEs) refer to software applications that combine all tools needed for a software development project, including an editor, compiler, and debugger. They consolidate different aspects of software development and significantly improve programmers' productivity.

Many types of programming environments exist in the market. Roughly there are two kinds: plain-text/code editor and IDE. Text/code editors, such as NotePad++ or Sublime Text, are easy to install and require minimal configuration but are not directly related to programming. On the other hand, a full-featured IDE combines all the necessary tools for development and testing in one package. However, IDEs require more disk space, memory, or a faster processor, and some may come at a cost due to expensive licenses. Despite these differences, the boundary between the two is not always clear, as users can enhance a text/code editor's functionality by installing plug-ins or extensions. The trade-offs between the time spent on installation/configuration and the power of the functionalities should be considered.

It is important to introduce IDEs in programming courses, especially in introductory-level courses. A great IDE benefits students from two perspectives. Firstly, it assists students in writing correct code. Through our teaching experience, we have observed that students with little or no programming experience often encounter common errors repeatedly. Secondly, it helps students establish good coding habits. Beginners tend to prioritize correctness over quality, resulting in messy code format, meaningless variable names, overly lengthy function, etc. Many IDEs offer features such as code formatting, variable name suggestions, and function length warning, which help students produce clean and professional code that adheres to industry standards. Given the learning curve associated with IDEs, comprehensive guidance and support are essential.

3. Visual Studio Code Investigation

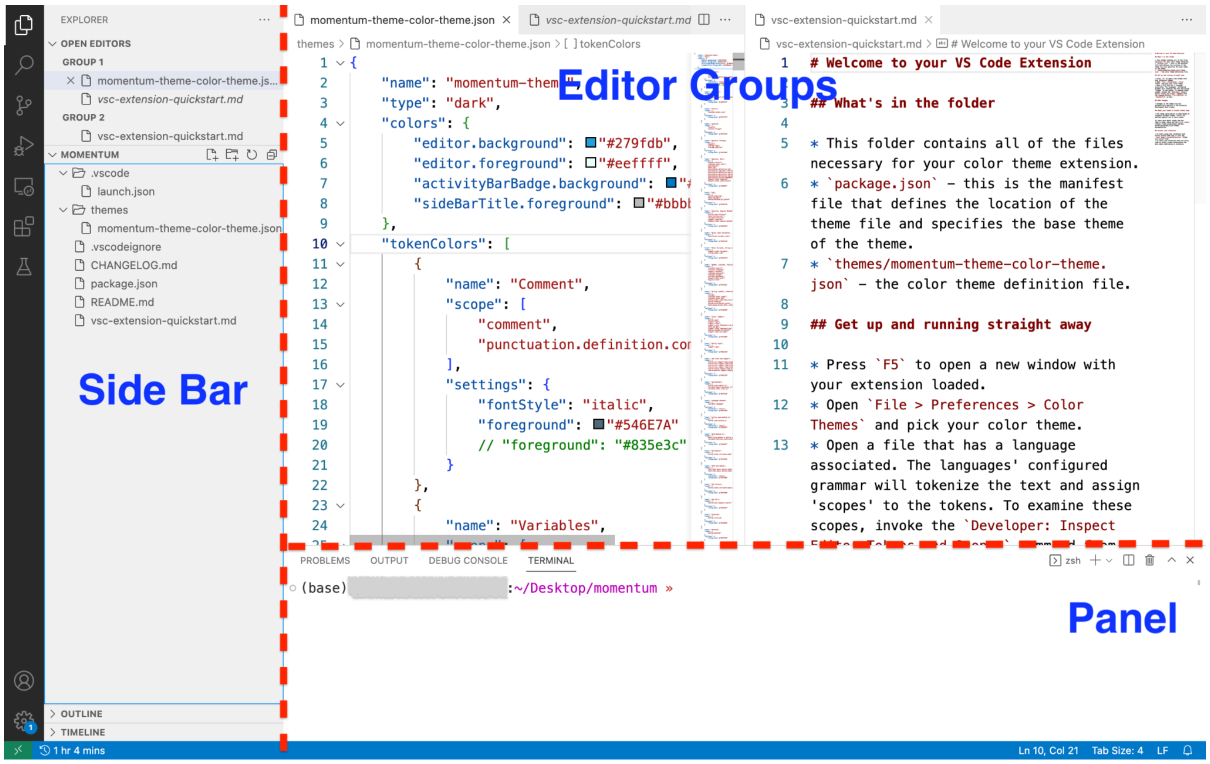
Visual Studio (VS) Code is known as a rich code editor developed and supported by Microsoft, it is free for private or commercial use. One of its core features is its extension support, allowing users to add languages, debuggers, and tools to enhance their development experience. By integrating relevant extensions, VS Code can transform into a desired IDE with powerful functionality. In addition to the standard extensions released by Microsoft, the VS Code Extension Marketplace [17] offers a wide range of extensions contributed by third-party organizations and individual developers. We analyze its practicality for introductory-level Python courses based on four aspects: accessibility, ease of use, functionality, and popularity.

Accessibility. VS Code provides cross-platform support, it runs on various operating systems, i.e. macOS, Windows, Linux, and on most available hardware with an identical user interface. VS Code has a small download (< 200 MB) and a disk footprint (< 500 MB), it is lightweight, and perfectly fits every student with different devices.

VS Code supports multi-languages, with extension support for almost every major programming language. Switching between different languages is effortless. Although our class is for Python beginners, students can continue coding with VS Code in later programming classes.

Easy-to-use. VS Code has a compact but simple user interface, as shown in Figure 1. The main editor, located in the center, is where students write their code assignments. Below the editor is the panel, which can display various views such as a debug console or a built-in terminal. The terminal always starts from the root of a certain workspace. On the left side is the sidebar, which contains different views; Figure 1 shows the Explorer view, which helps in locating files. The ability to maintain a single window for coding and debugging enhances programming efficiency.

In addition to its concise user interface, VS Code offers many features to enhance development. One important feature is the workspace configuration setting. A VS Code workspace is the root folder of the current project, allowing users to configure settings that apply specifically to that workspace. This is particularly useful for students who often need different settings (such as interpreter version, dependent libraries, and programming languages) for different projects. While students may initially find these configuration settings daunting, especially when setting up or resetting the environment for a new project, working with VS Code workspaces significantly simplifies this process. It enables users to configure settings within the context of the current workspace, effectively overriding any global user settings.


EXPLORER
OPEN EDITORS
GROUP 1
momentum-theme-color-theme.js...
vsc-extension-quickstart.md
GROUP 2
vsc-extension-quickstart.md
MOMENTUM
.vscode
launch.json
themes
momentum-theme-color-theme.json
.vscodeignore
CHANGELOG.md
package.json
README.md
vsc-extension-quickstart.md
Side Bar
OUTLINE
TIMELINE
1 hr 4 mins
momentum-theme-color-theme.json
vsc-extension-quickstart.md
themes > momentum-theme-color-theme.json > [ ] tokenColors
Editor Groups
vsc-extension-quickstart.md
vsc-extension-quickstart.md > # Welcome to your VS Code Extension
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
zsh
Panel
Ln 10, Col 21
Tab Size: 4
LF

```
{
    "name": "momentum-them
    "type": "dark",
    "colors": {
        "editor.background": "#279fdb",
        "editor.foreground": "#eeffff",
        "activityBarBadge.background": "#
        "sideBarTitle.foreground": "#bbbb
    },
    "tokenColors": [
        {
            "name": "Comment",
            "scope": [
                "comment",
                "punctuation.definition.cor
            ],
            "settings": {
                "fontStyle": "italic",
                "foreground": "#546E7A"
                // "foreground": "#835e3c"
            }
        },
        {
            "name": "Variables",
```

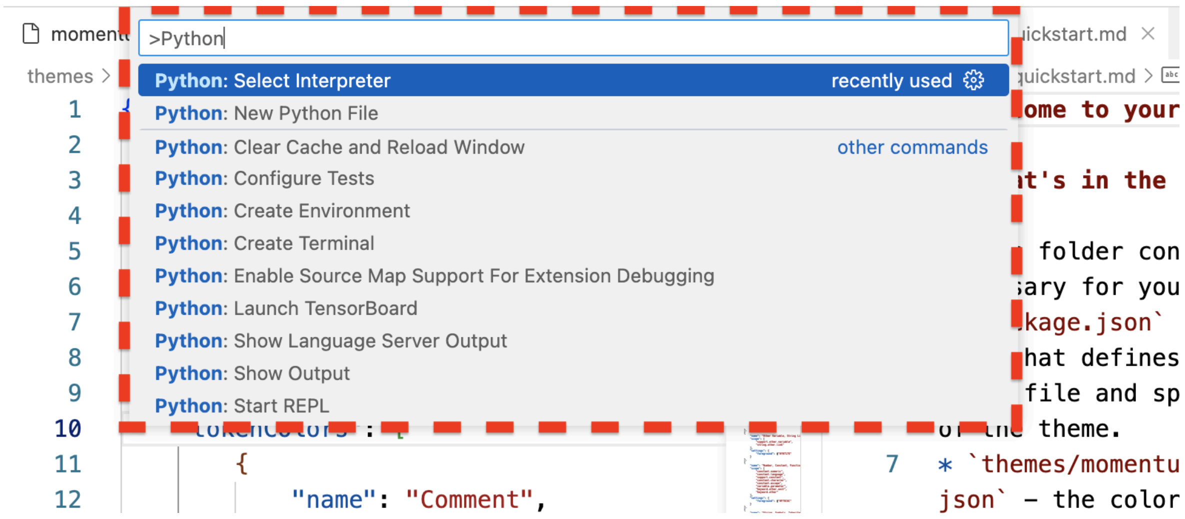
```
# Welcome to your VS Code Extension

## What's in the folder

* This folder contains all of the files
necessary for your color theme extension.
* `package.json` - this is the manifest
file that defines the location of the
theme file and specifies the base theme
of the theme.
* `themes/momentum-theme-color-theme.
json` - the color theme definition file.

## Get up and running straight away

* Press `F5` to open a new window with
your extension loaded.
* Open `File > Preferences > Color
Themes` and pick your color theme.
* Open a file that has a language
associated. The languages' configured
grammar will tokenize the text and assign
'scopes' to the tokens. To examine these
scopes, invoke the `Developer: Inspect
```

```
(base) [redacted]:~/Desktop/momentum »
```

Fig.1. The basic layout of VS Code user interface [7] that we introduced to students. The editor is the area to edit files, students can open multiple editors at the same time side by side vertically or horizontally. The panel below the editor is for output or debug information, errors, and warnings, or an integrated terminal. The side bar contains different views like the Explorer or Extension Marketplace, to assist students while working on the projects or downloading extensions.

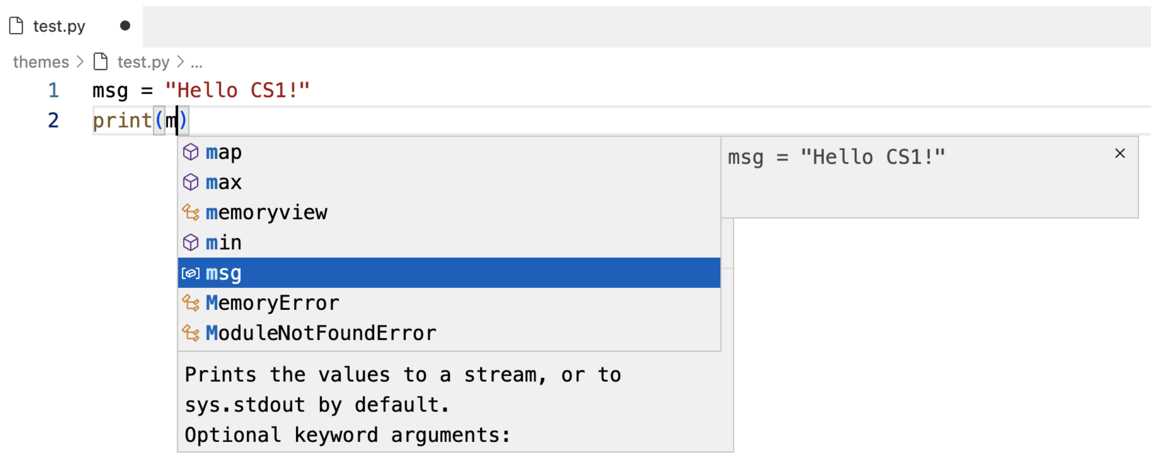
Furthermore, VS Code supports remote development. The *Remote SSH* extension allows opening a remote folder on any remote machine, virtual machine, or container running SSH server. Another useful feature favored by many developers is the *Command Palette*, Figure 2 shows an example, where users would have access to all of the functionalities of VS Code, including commands of your installed extensions.

Functionality. In addition to its basic features as a source code editor, VS Code offers extensions to enhance its functionality. With over 30,000 extensions available in the Marketplace, users can select their preferred extensions and customize their installation to improve their programming experience. For example, in our Python course, we recommend that students install the Python extension developed by Microsoft. This extension provides strong support for the Python language and includes powerful features such as IntelliSense [15], code formatting, debugging, variable explorer, and more. which offers strong support for the Python language. IntelliSense is a particularly useful feature that provides suggestions as you type. It offers intelligent code completion based on the language semantics and the source code you've written. Figure 3 shows an example, where IntelliSense suggests using variable `msg` in the `print` function.



```
>Python
Python: Select Interpreter                                  recently used
Python: New Python File
Python: Clear Cache and Reload Window                       other commands
Python: Configure Tests
Python: Create Environment
Python: Create Terminal
Python: Enable Source Map Support For Extension Debugging
Python: Launch TensorBoard
Python: Show Language Server Output
Python: Show Output
Python: Start REPL
```

Fig.2. The red box indicated the Command Palette of VS Code. Students can access all functionality of VS Code and install extensions through the Command Palette.



```python
msg = "Hello CS1!"
print(m)
```

```
map
max
memoryview
min
msg
MemoryError
ModuleNotFoundError
```

```
msg = "Hello CS1!"
```

```
Prints the values to a stream, or to
sys.stdout by default.
Optional keyword arguments:
```

Fig.3. An example of code completion supported by VS Code, where IntelliSense suggests using the variable `msg` in `print` function.

Popularity. VS Code is widely favored among real-world developers. According to the Stack Overflow 2021 Developer survey [30], it ranks as the most popular developer environment tool, with 71.06% of over 80,000 respondents reporting its use. In the latest Stack Overflow Developer Survey [40], it continues to be the preferred developer environment tool among all developers, with the number increasing to 73.71% of over 90,000 respondents. Due to the great user amount, bugs/issues are fixed/solved in time. VS Code updates frequently, from Mar 2023 to Mar 2024, VS Code development team made 33 releases in total, updating with big change almost every month[2]. Thanks to its large user base, bugs and issues in VS Code are promptly addressed. The software receives frequent updates, with the VS Code development team releasing a total of 33 updates from March 2023 to March 2024, introducing a new version each month with new features and important bug fixes.

The high popularity of VS Code is one of the main reasons why we introduce it to our Python class. Students often encounter challenges when switching between different IDEs, so using VS Code allows them to maintain consistency across various programming language classes and in their future careers.

4. Guidance and Support

To accelerate the learning process of VS Code for students with diverse backgrounds, we are introducing comprehensive guidance in our Python class. The guidance is divided into separate tutorials, with the following subsections detailing each tutorial.

Download and installation of VS Code. In this tutorial, students are guided to the download page and instructed on how to choose the correct version based on their operating system. We introduce the workspace immediately after installation because it is the fundamental unit for project manipulation. We provide examples of how to start a project in VS Code and how to organize files within a project.

Download and installation of Python. In our past teaching experience, students encountered challenges with multiple versions of Python, including: 1. confusion between Python 2 and Python 3, 2. installing the incorrect version, and 3. having both Python 2 and Python 3 installed on a single device without knowing how to switch between them.

In the tutorial, we require students to use Python 3 and emphasize the differences between Python 2 and Python 3. Before installation, students are instructed to check if they have any versions of Python pre-installed. If they don't have Python installed or have Python 2 installed, they are directed to the newest Python 3 download link. If they already have any version of

[2] https://github.com/microsoft/vscode/releases

Python 3 installed, there is no need to reinstall it. For students with multiple versions of Python installed, we provide a detailed guide on switching Python versions in the next tutorial.

Set up Python environment. Next, students install the Python extension and configure the Python environment in VS Code. Based on our teaching experience, students often struggle with programming environment settings. Therefore, we focus extensively on explaining where and how to configure these settings.

Following this, students are directed to the extension marketplace to locate and install the Python extension developed by Microsoft. This extension is compatible with any operating system and Python version, and includes the Jupyter extension in its installation bundle. The Python extension enhances the coding experience with features like code completion and IntelliSense (based on the currently selected Python interpreter). We offer step-by-step guidance, accompanied by figures, on how to select a Python interpreter from the *Command Palette*. Additionally, we provide instructions on changing the programming language mode for students interested in working with other programming languages in subsequent classes.

Run Python examples. Once the Python environment is set up, VS Code becomes a lightweight Python IDE. In this tutorial, we guide students through writing and running their first simple Python program from scratch. We also explain how Python extensions improve coding. As part of the tutorial, we require students to generate expected outputs for example programs, which ensures that they have identical settings for the Python environment.

Install and run Jupyter extension. We utilize VS Code's Jupyter extension to run in-class coding exercises to enhance student engagement [2, 24]. With the Jupyter extension, students open and run *.ipynb* files on VS Code just like Python files. This extension offers basic notebook support for language kernels and enables the use of any Python environment to be used as a Jupyter kernel. In this tutorial, we provide students with guidance on installing the Jupyter extension, creating new Jupyter files, and managing and running the code cells.

Install and use scientific packages. Python Packages are highly effective for addressing complex problems in scientific computing, data visualization, data manipulation, and many other fields. In this tutorial, we instruct students on how to install and import Python libraries, e.g., *Numpy*, *Pandas*, *Matplotlib*, and *SciPy*. Then students are tasked with learning and utilizing basic APIs, with the example codes provided for reference. To successfully complete this tutorial, students are required to plot a specific figure using Matplotlib functions.

4.1. Improvements for Guidance

To enhance the guidance provided to students, we have implemented the following optimizations:

- Operating system specific guidance: While both VS Code and Python are well-supported across platforms, we recognize that minor differences among operating systems can impact the learning experience (especially for Python environment setup). Therefore, we offer two versions of guidance tailored for MacOS and Windows users, respectively.
- Multiple access points: We understand that students have different preferences for accessing guidance. Hence, we offer two ways to access the guidance: PDF download and website. While the traditional approach involves uploading guidance as PDFs to the course learning management system, we have also created a website version for improved representation. Additionally, in light of COVID-19, which has limited in-person instruction, we provide video guidance to enhance student engagement.
- Continuous improvement: We actively collect feedback from students and use it to regularly update and improve the guidance material, ensuring that it remains relevant and effective.

4.2. Hierarchical Indexing

We provide detailed tutorials with step-by-step instructions to help students set up their programming environment effortlessly. However, students' preferences for tutorials vary based on their programming backgrounds. Tutorials with a flat structure may not meet the needs of all students. For example, those who prefer video tutorials may not want to follow the video's pace from the beginning. A common approach is to start with a verbal demonstration and only refer to the video when encountering problems. However, it can be challenging to find the relevant timestamp in the video based on text instructions if the tutorials are linearly structured. Moreover, students with some programming knowledge may require more concise instructions. For instance, they may have already installed Python and VS Code but need guidance on setting up the Python environment in VS Code, which flat structured tutorials may not provide. On the other hand, students who are familiar with environment setup may find it difficult to extract key messages from flat-structured tutorials.

We have implemented hierarchical indexing in our tutorials to cater to diverse student requirements. This indexing comprises three levels:

- High-level indexing: At the beginning of each tutorial, we provide an abstract that includes all key messages such as software names, download links, software versions, etc.

- Medium-level indexing: Each tutorial is segmented into several parts, each with its own summary and subtitles. We list these subtitles and provide links to the corresponding positions in the tutorial. This allows students to easily locate the information they need.
- Low-level indexing: We have created mappings between the verbal and video demonstrations. Specifically, verbal demonstrations for each step within a subtitle are linked to the corresponding video timestamps. This enables students to switch between verbal and video demonstrations when they encounter problems, facilitating a more effective learning experience.

5. Evaluation and Student Responses

In this section, we evaluate two objectives:

- Validate VS Code integrating with relevant extensions, is a suitable IDE for the introductory-level Python programming courses
- Validate the value and necessity of VS Code's Guidance

5.1. Course Description

We introduce VS Code and VS Code guidance to a CS1 Python programming course in Fall 2022 and Spring 2023 semesters, the course enrollment is 141 and 93, respectively. This CS1 course is restricted for non-CS students, with an emphasis on basic programming skills and engineering applications. Students learn how to solve problems through writing Python programs, particular elements include: the development of Python programs from specifications; documentation and coding style; use of data types, control structures and data structures; abstractions and verification.

The course employs a flipped classroom format. In the first week' lab, students set up their coding environment on their computers following the VS Code Guidance. Lab instructors are available to offer additional assistance if necessary. Each week, students begin by conceptual learning the topics covered in the week through online videos and self-check quizzes. They then participate in a 100-minute class exercises session and a 165-minute lab exercises session to engage in active learning activities. Following class, students work on weekly homework assignments and project tasks to reinforce their understanding and skills. Students are required to run the in-class coding exercises via the Jupyter Notebook extension installed in VS Code and complete the programming tasks (homework assignments, lab exercises, and projects) in VS Code independently.

5.2. VS Code and VS Code Guidance Evaluation

We designed a survey of 21 questions to gather students' feedback on both VS Code and the VS Code Guidance at the end of the semester. The survey first collects information about students' backgrounds, then asks them to rate VS Code and the guidance based on their programming experience throughout the semester. We received a total of 82 valid responses, with 42 responses from the Fall 2022 and 40 responses from the Spring 2023.

First, we collected students' background information. In Fall 2022 and Spring 2023, 79% and 83% of students were freshmen and sophomores, respectively. Additionally, 86% and 88% of students, respectively, had not taken any Python-related programming courses in the past, and 74% and 75% of them had less than one year of coding experience.

Next, students were asked to rate the performance of both VS Code and the VS Code Guidance throughout the semester. They rated VS Code based on the following four aspects, with the degree of satisfaction ranging from 1 to 5, where 1 indicates strong dissatisfaction and 5 indicates strong satisfaction:

- Visual appeal: Students rated their satisfaction with VS Code's user interface and editor layout.
- Extension ecosystem: Students rated the ease of searching for, installing, and uninstalling extensions.
- Debugging experience: Students rated the effectiveness of VS Code's built-in debugger in accelerating their edit, compile, and debug loop, as well as the helpfulness of recommended debugger extensions.
- Editing experience: Students rated the overall editing experience of VS Code, including the usefulness and beginner-friendliness of basic editing features (such as keyboard shortcuts and Command Palette), as well as the effectiveness of IntelliSense (code editing features such as code completion, parameter info, and content assist).

Based on the responses, in the Fall 2022 semester, 61% of students did not have experience with IDEs or coding platforms in the past, while 39% had used some other IDEs. In the Spring 2023 semester, 70% of students did not have experience with IDEs or coding platforms in the past, while 30% had used some other IDEs such as Matlab, RStudio, etc.

As shown in Table 1, the average ratings for visual appeal, extension ecosystem, debugging experience, and editing experience were 4.17, 3.81, 4.02, and 4.05, respectively, in Fall 2022, and 4.28, 3.71, 3.7, and 4.73, respectively, in Spring 2023. We released a VS Code extension that periodically recommends and installs themes for students, which significantly improved their editing experience[3]. Regarding students' overall satisfaction with the VS Code guidance, it was 4.2 in Fall 2022 and 4.0 in Spring 2023.

Table.1. The averaged answers of students' satisfaction (82 students in total) over four aspects: visual appeal, extension ecosystem, debugging experience, and editing experience. The degree of satisfaction lies between 1 to 5, in which 1 is strongly dissatisfied and 5 is strongly satisfied.

Semester	Visual Appeal	Extension	Debugging	Editing
Fall 22	4.17	3.81	4.02	4.05
Spring 23	4.28	3.71	3.7	4.73

Among the total 82 students, 74% students consider VS Code easy to install and use with the help of VS Code Guidance, and 76% of students consider VS Code to be a good IDE for coding. There are 13 students reported having issues or trouble with the VS Code guidance throughout the semester, and we have collected and addressed these issues in the updated VS Code guidance.

5.3. Class Averages

We also compared the class averages of students' final grades from Spring 2022 with Fall 2022 and Spring 2023. The results are shown in Table 2. In Spring 2022, the class average total score was 82.86%, while it was 85.10% in Fall 2022 and 84.34% in Spring 2023.

While the comparison in Table 2 indicates that the class averages of Fall 2022 and Spring 2023 are higher than those of Spring 2022, we cannot solely attribute the average increase to VS Code or the VS Code guidance. In Spring 2022, we used the traditional classroom, and students were required to use Spyder integrated within Anaconda. We adopted a flipped classroom approach and utilized VS Code in Fall 2022 and Spring 2023. Based on students' experiences, we observed that both using VS Code and the VS Code guidance had positive effects on completing coding assignments.

Table 2. The class average of total score in the Spring 2022, Fall 2022, and Spring 2023 semesters.

Semester	Spring 22	Fall 22	Spring 23
Class Average	82.86%	85.10%	84.34%

[3] We have collected students' feedback, but the data will not be provided in this paper since it has been discussed in other papers.

5.4. Student Comments

At the end of the student survey, students were invited to comment on their experience with VS Code and the VS Code guidance throughout the semester. One student described their overall experience as:

> I can view Python files and text files alike without even making a new window using VS Code. The interface is visually appealing. The VS Code tutorial is straightforward. Instructions are easy to follow. That's all a tutorial really needs, in my opinion.

Some students prefer the dark mode in VS Code. One student claimed that it improves their coding experience:

> I like VS Code uses color coordination to make different code types and functions stand out. I also like that by default the background is black it makes it easier to read. 90% of the time it's very easy to find mistakes within codes. Overall the tutorial is easy to follow.

We have observed that many students prefer dark mode over bright mode, possibly because dark mode or dark themes make syntax highlighting stand out more. With a light background, the code text is mostly darker colors, whereas it is more colorful with a dark background. Additionally, changing themes in VS Code is quite easy compared to pedagogical IDEs. Students highly appreciate the theme extensions in VS Code. One student claimed:

> I like the way VS Code installs extensions. I especially love theme extensions in the Marketplace, whenever I am bored with codes I change my theme, and it just becomes a totally new software!

From the feedback, we have gathered many constructive suggestions. Some students have complained about the complexity of commands on the terminal. While it's true that we may include plenty of terminal instructions at the beginning of class, the instructor believes that these instructions/commands are necessary for this class. To address this issue, we have divided the command study into different labs and have added detailed explanations for every instruction.

Another student considered the VS Code guidance useful but suggested that we could involve more helper extensions in class:

> Tutorials are thorough, straight to the point. But in my opinion, VS Code lacks collaborative elements, especially in the lab. I feel that collaboration via a shared document would be beneficial because, without it, one user mostly does the code by themselves.

The Live Share extension [41] enables students to share screens and collaborate with classmates, TAs, and instructors on the same code without the need to synchronize code or configure the same development tools, settings, or environment. This ability to work together and independently provides a collaboration experience that closely resembles in-person collaboration [16]. We plan to enable Live Share for classroom lectures (in read-only mode) and remote office hours in Fall 2024. The use of Live Share in pair programming is still under evaluation and consideration.

To summarize, students' overall experience with VS Code and the VS Code guidance is positive and encouraging. This indicates that both VS Code and the guidance are valuable and promising for the CS1 course.

5.5. Issues with Jupyter Notebook in VS Code

We have observed two issues with running Jupyter Notebook files in VS Code: 1. More than 10 students (using both Mac and Windows devices), reported that when running a cell or all cells in a Jupyter Notebook file, it loads for a long time but does not show any content; 2. Some students reported that when running a cell or all cells in a Jupyter Notebook file, VS Code crashes. While restarting or upgrading VS Code solved the issue for 2 students, for most cases we can not fix these as running cells bloat device memory. To address these issues, we have switched to using the Jupyter Notebook web-based platform instead of the Jupyter Notebook extension in VS Code.

6. Discussions

Discussions on education-related concerns. VS Code offers advanced features like code auto-completion and function signature hints, which raises debates regarding their use in introductory-level programming courses. Some argue that these supplementary features may lead to a blind dependence on IDEs, potentially hindering students' understanding of the code. However, in practice, our teaching plan covers all necessary programming knowledge and concepts. We believe that features like auto-completion and function signature hints can actually help students learn faster and more efficiently. Moreover, these features can be customized or completely disabled in VS Code if required, allowing instructors to tailor the IDE to their teaching approach.

Discussions on some limitations. We acknowledge some limitations in this study. We only evaluated an introductory Python course, as Python is the introductory programming language offered by our department. However, VS Code supports multiple programming languages, such as Matlab, Java, and C/C++. It is straightforward to adapt our guidance to courses with other programming languages. In the future, we plan to partner with instructors in our department to report more experiences with VS Code in various programming courses.

7. Conclusions and Future Work

This paper describes the experiences with introducing Visual Studio Code in an introductory (CS1) Python programming course at a large engineering university. We investigate the effectiveness of VS Code as a desired IDE for CS1 programming courses and develop comprehensive guidance to assist students with varying programming backgrounds. We perform evaluations among students and validate the practicality of VS Code and assess the quality of our VS Code guidance. We continuously update and improve the guidance based on collected feedback. We are practicing VS Code and our VS Code guidance in additional CS1 programming courses that cover languages beyond Python.

Our future work is two-fold. First, we will continue exploring useful VS Code extensions available in the marketplace that benefit education. Second, we will develop our own extensions to further support education-related activities. Specifically, we aim to create extensions that enhance student engagement during programming assignments.

REFERENCES

[1]. Abadi, Martın, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow. org (2015)." *https://www. tensorflow. org* (2015).

[2]. Al-Gahmi, Abdulmalek, Yong Zhang, and Hugo Valle. "Jupyter in the Classroom: An Experience Report." *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education-Volume 1. 2022.*

[3] Atom. 2022. Atom Homepage. *http://atom.io/.*

[4] Barba, Lorena A., Lecia J. Barker, Douglas S. Blank, Jed Brown, Allen B. Downey, Timothy George, Lindsey J. Heagy et al. "Teaching and learning with Jupyter." *Recuperado: https://jupyter4edu. github. io/jupyter-edu-book* (2019): 1-77.

[5] Chen, Yu, Zhenming Liu, Bin Ren, and Xin Jin. "On efficient constructions of checkpoints." *arXiv preprint arXiv:2009.13003* (2020).

[6] Chen, Yu, Ivy B. Peng, Zhen Peng, Xu Liu, and Bin Ren. "Atmem: Adaptive data placement in graph applications on heterogeneous memories." In *Proceedings of the 18th ACM/IEEE International Symposium on Code Generation and Optimization*, pp. 293-304. 2020.

[7] Visual Studio Code. 2022. User Interface. *https://code.visualstudio.com/docs/getstarted/userinterface.*

[8] Tyne Crow, Andrew Luxton-Reilly, and Burkhard Wuensche. 2018. Intelligent tutoring systems for programming education: a systematic review. *In Proceedings of the 20th Australasian Computing Education Conference*. 53–62.

[9] Cutting, Vineesh, and Nehemiah Stephen. "A Review on using Python as a Preferred Programming Language for Beginners." (2021).

[10] DePratti, Roland. "Using Jupyter notebooks in a big data programming course." *Journal of Computing Sciences in Colleges* 34, no. 6 (2019): 157-159.

[11] Eclipse Foundation. 2022. Eclipse Homepage. *https://www.eclipse.org/ide/.*

[12] JetBrains. 2022. Essential tools for software developers and teams. *http://www.jetbrain.com/.*

[13] Khoirom, Selina, Moirangthem Sonia, Borishphia Laikhuram, Jaeson Laishram, and Tekcham Davidson Singh. "Comparative analysis of Python and Java for beginners." *Int. Res. J. Eng. Technol* 7, no. 8 (2020): 4384-4407.

[14] Rodrigo Pessoa Medeiros, Geber Lisboa Ramalho, and Taciana Pontual Falcão. 2018. A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education* 62, 2 (2018), 77–90.

[15] Microsoft. 2022. IntelliSense. *https://code.visualstudio.com/docs/editor/intellisense.*

[16] Visual Studio Code. Collaborate with Live Share.
https://code.visualstudio.com/learn/collaboration/live-share

[17] Microsoft. 2022. VS Code Marketplace. *https://marketplace.visualstudio.com/.*

[18] Samy S Abu Naser. 2008. Developing an intelligent tutoring system for students learning to program in C++. *Information Technology Journal,* Scialert 7, 7 (2008), 1055–1060.

[19] Oliphant, Travis E. *Guide to numpy*. Vol. 1. USA: Trelgol Publishing, 2006.

[20] Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. "Automatic differentiation in pytorch." (2017).

[21] Arnold Pears, Stephen Seidman, Lauri Malmi, Linda Mannila, Elizabeth Adams, Jens Bennedsen, Marie Devlin, and James Paterson. 2007. A survey of literature on the teaching of introductory programming. Working group reports on *ITiCSE on Innovation and technology in computer science education* (2007), 204–223.

[22] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al. "Scikit-learn: Machine learning in Python." *the Journal of machine Learning research* 12 (2011): 2825-2830.

[23] Pellet, Jean-Philippe, Amaury Dame, and Gabriel Parriaux. "How beginner-friendly is a programming language? A short analysis based on Java and Python examples." (2019).

[24] Perez, Fernando, and Brian E. Granger. "Project Jupyter: Computational narratives as the engine of collaborative data science." *Retrieved September* 11.207 (2015): 108.

[25] PYPL. 2022. PopularitY of Programming Language. *http://pypl.github.io/PYPL.html.*

[26] Charles Reis and Robert Cartwright. 2004. Taming a professional IDE for the classroom. *In Proceedings of the 35th SIGCSE technical symposium on Computer science education*. 156–160.

[27] Norsaremah Salleh, Emilia Mendes, and John Grundy. 2010. Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review. *IEEE Transactions on Software Engineering* 37, 4 (2010), 509–525.

[28] Carsten Schulte and Jens Bennedsen. 2006. What do teachers teach in introductory programming?. *In Proceedings of the second international workshop on Computing education research*. 17–28.

[29] Srinath, K. R. "Python–the fastest growing programming language." *International Research Journal of Engineering and Technology* 4, no. 12 (2017): 354-357.

[30] Stackoverflow. 2021. 2021 Developer Survey. *https://insights.stackoverflow.com/survey/2021#section-most-popular-technologies-integrated-development-environment.*

[31] Tan, Jialiang, Yu Chen, Zhenming Liu, Bin Ren, Shuaiwen Leon Song, Xipeng Shen, and Xu Liu. "Toward efficient interactions between Python and native libraries." In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 1117-1128. 2021.

[32] Python Tutor. 2021. Learn Python, JavaScript, C, C++, and Java. *https://pythontutor.com/.*

[33] Valstar, Sander, Sophia Krause-Levy, Alexandra Macedo, William G. Griswold, and Leo Porter. "Faculty views on the goals of an undergraduate CS education and the academia-industry gap." In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, pp. 577-583. 2020.

[34] Valstar, Sander, Caroline Sih, Sophia Krause-Levy, Leo Porter, and William G. Griswold. "A quantitative study of faculty views on the goals of an undergraduate CS program and preparing students for industry." In *Proceedings of the 2020 ACM Conference on International Computing Education Research*, pp. 113-123. 2020.

[35] Van Der Walt, Stefan, S. Chris Colbert, and Gael Varoquaux. "The NumPy array: a structure for efficient numerical computation." *Computing in science & engineering* 13, no. 2 (2011): 22-30.

[36] Van Dusen, Eric. "Jupyter for teaching data science." *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. 2020.

[37] Kelsey Van Haaster and Dianne Hagan. 2004. Teaching and Learning with BlueJ: an Evaluation of a Pedagogical Tool. *Issues in Informing Science & Information Technology* 1 (2004).

[38] Kurt VanLehn. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational psychologist* 46, 4 (2011), 197–221.

[39] vim. 2022. Vim The Editor. *https://www.vim.org/*

[40] Stackoverflow. 2023. 2023 Developer Survey. *https://survey.stackoverflow.co/2023/#most-popular-technologies-new-collab-tools*

[41] Microsoft Marketplace. Live share extension pack download page. *URL: https://marketplace.visualstudio.com/items?itemName=MS-vsliveshare.vsliveshare*

[42] Eclipse Foundation, Eclipse IDE homepage, *URL: https://eclipseide.org/*

[43] PyCharm homepage, JetBrains, *URL: https://www.jetbrains.com/pycharm/features/*