

1.VScode的python环境配置

A anaconda下载

1.下载 Anaconda 安装程序

访问 Anaconda 官方网站的下载页面：<https://www.anaconda.com/products/distribution>

在这个页面，你会看到针对不同操作系统的版本。选择适合你系统的版本并下载。通常，你需要选择一个 64-位的图形化安装程序（.exe 文件对于 Windows，.pkg 文件对于 macOS，.sh 文件对于 Linux）。

2.运行安装程序

下载完成后，按照以下步骤运行安装程序：

Windows

1. 双击下载的 .exe 文件。
2. 按照安装向导的指示进行操作。在安装过程中，大多数设置都可以使用默认值。
3. 在“高级安装选项”(Advanced Options) 步骤，请**勾选**“Add Anaconda to my PATH environment variable”(将 Anaconda 添加到我的 PATH 环境变量)。**除非你打算使用多个 Python 版本并且知道如何手动配置，否则强烈建议勾选此项。**
4. 完成安装。

macOS

1. 双击下载的 .pkg 文件。
2. 按照安装向导的指示进行操作。
3. 在安装向导的最后，你会看到一个提示，告诉你 Anaconda 已经安装成功。

Linux

1. 打开终端 (Terminal)。
2. 导航到你下载 .sh 文件的目录，例如：`cd ~/Downloads`。
3. 运行安装脚本，命令如下：`bash Anaconda3-202X.XX-Linux-x86_64.sh` (将文件名替换为你下载的版本号)。
4. 按照终端的提示操作。在询问你是否接受许可协议时，输入 `yes`。
5. 当提示你是否将 Anaconda 添加到 PATH 时，输入 `yes`。
6. 安装完成后，关闭并重新打开终端，以使更改生效。

3.验证安装

为了确认 Anaconda 已经正确安装，打开一个新的终端（Terminal 或 Anaconda Prompt），然后输入以下命令：

Bash

```
conda --version
```

如果安装成功，终端会显示 Conda 的版本号，例如 `conda 23.7.4`。

你也可以通过输入以下命令来检查 Conda 环境是否正常工作：

Bash

```
conda list
```

如果一切正常，终端会列出你当前 Conda 环境中所有已安装的包。

B 安装 VS Code

vscode下载地址：[Visual Studio Code - Code Editing. Redefined](#)

注意vscode下载时记得勾选“添加到PATH”自动加入环境变量



C 安装 VS Code 扩展（可选）

打开 VS Code，点击侧边栏的**扩展（Extensions）**图标（或者按 `Ctrl + Shift + X`）。在搜索框中输入以下扩展并安装：

- **Python:** 微软官方出品，这是核心扩展，提供了代码补全、调试、Linting（代码检查）、格式化等基础功能。
- **Pylance:** 同样是微软出品的 Python 语言服务器。它能提供更快、更智能的代码补全和类型检查，是 Python 扩展的推荐伴侣。

D 配置 Python 解释器

安装好扩展后，你需要告诉 VS Code 使用哪个 Python 解释器来运行你的代码。

- **打开命令面板:** 按下 `Ctrl + Shift + P`。
- **选择解释器:** 在命令面板中输入 `Python: Select Interpreter`，然后回车。
- **选择路径:** VS Code 会自动检测你电脑上已安装的 Python 解释器。选择你想要使用的版本，这里通常会显示为**你的anaconda下载路径/python.exe**

如果你使用了**虚拟环境**，也可以在这里选择。（本次不讲）

E 尝试运行你的代码

编写一段代码，例如：

```
print("hello world!")
```

在 `ctrl + shift + ~` 打开vscode终端，`python <你的文件名>.py`即可运行

2.copilot学生版获取

copilot，作为微软推出的内置了许多大模型接口的实用工具，在工程完成上可以起到事半功倍倍倍的作用（例如小学期OOP）。

当然，我们也可以使用阿里的qianwen插件或者字节的trae插件代替（毕竟copilot一个月要20刀），不过copilot中集成的Claude Sonnet4、gemini 2.5pro等高级模型也许会给你带来全新的体验，特别是作为学生可以薅羊毛的**学生优惠**。

下面我们来看看如何**白嫖**copilot吧！（参考：<https://mp.weixin.qq.com/s/3TTD-Qz9FFn7eLwSU1Qqyw>）

1. 准备工作

首先用**学生邮箱**注册一个 github 账号（已经有 github 号的同学可以添加学生邮箱，并设置成首选）然后将个人主页按下图模板设置好：

Name

 拼音名，姓在后，中间打空格

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

  Remove 你的教育邮箱

You can manage verified email addresses in your [email settings](#)

Bio

改为相应学校

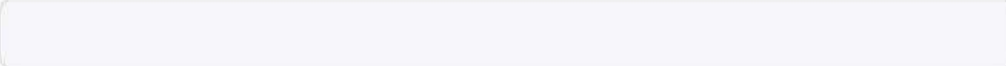
I am a grade 2024 undergraduate from Tsinghua University, and I hope to learn code in depth on github platform

You can @mention other users and organizations to link to them.

Pronouns

he/him 性别

URL



ORCID iD

You have a connected ORCID iD 0009-0002-8970-2449 for the account @zgdlit.

☒ Display your ORCID iD on your GitHub profile


Disconnecting your ORCID iD may affect areas of your profile where your ORCID iD is displayed.


 Disconnect your ORCID iD

Social accounts

 Link to social profile 1

 Link to social profile 2

 Link to social profile 3

 Link to social profile 4

Company 相应学校

Tsinghua University

You can @mention your company's GitHub organization to link it.

Location 学校地址

30 Shuangqing road, Haidian District, Beijing City

☒ Display current local time

Other users will see the time difference from their local time.

Time zone

(GMT+08:00) Beijing

All of the fields on this page are optional and can be deleted at any time, and by filling them out, you're giving us consent to share this data wherever your user profile appears. Please see our [privacy statement](#) to learn more about how we use this information.

Update profile

Public profile

Account

Appearance

Accessibility

Notifications

Access

Billing and licensing

New ^

Overview

Usage

Budgets and alerts

Licensing

Payment information

Payment history

Additional billing details

Education benefits

Emails

Password and authentication

Sessions

SSH and GPG keys

Organizations

Enterprises

Payment information

Billing information

Add your information to show on every invoice

First name *

Last name *

Address (Street, P.O. box) *

北京市海淀区清华园一号清华大学

Address line 2 (Apartment, suite, unit)

City *

北京

Country/Region *

China

State/Province

Postal/Zip code (9-digit zip code for US)

Required for certain countries

Required for certain countries

VAT/GST ID

Save billing information

Cancel

然后需要设置双重验证，在手机上下载 Microsoft Authenticator（应用商店搜索即可）在 github 的 password and authentication 栏目中（就在 payment information 下面），在 Two-factor methods 下选择 **Authenticator app**，按照网页要求进行设置即可

Two-factor authentication



...

Two-factor authentication adds an additional layer of security to your account by requiring more than just a password to sign in. [Learn more about two-factor authentication.](#)

Preferred 2FA method
Set your preferred method to use for two-factor authentication when signing into GitHub.


Authenticator app ▾

Two-factor methods

 **Authenticator app** Configured 


Use an authentication app or browser extension to get two-factor authentication codes when prompted.

Edit

 **SMS/Text message**


Get one-time codes sent to your phone via SMS to complete authentication requests.

Add

 **Security keys**

Security keys are webauthn credentials that can only be used as a second factor of authentication.


Edit

 **GitHub Mobile** Configured 2 devices

GitHub Mobile can be used for two-factor authentication by installing the GitHub Mobile app and signing in to your account.

Show

Recovery options



 **Recovery codes** Viewed

Recovery codes can be used to access your account in the event you lose access to your device and cannot receive two-factor authentication codes.

View

2. 学生认证

Access

-  **Billing and licensing** New 
- Overview

Usage



Budgets and alerts

Licensing

Payment information

Payment history

Additional billing details

 Education benefits 

新注册的 `github` 账号可能会认证失败，稍等几天就好

github会根据学生邮箱自动选择学校

之后会要求选择用什么做证明，选第一项即可，学生证是最好的证明

4. copilot配置

1. github部分

在copilot设置中把能勾的全勾上

Public profile

Account

Appearance

Accessibility

Notifications

Access

Billing and licensing

Emails

Password and authentication

Sessions

SSH and GPG keys

Organizations

Enterprises

Moderation

Code, planning, and automation

Repositories

Codespaces

Packages

Copilot

Features

Coding agent

Pages

Saved replies

Security

Code security

Integrations

Applications

Scheduled reminders

Archives

Security log

Sponsorship log

<> Developer settings

GitHub Copilot

GitHub Copilot Pro is active for your account

You currently have an active [Copilot Pro subscription](#).

Get started by installing the extension in your preferred environment.

Copilot in your IDE

Copilot in the CLI

Chat in GitHub Mobile

More features

Features

<div>Editor preview features</div> <div>You can use preview features in your editor. Learn more.</div>	Enabled
<div>Copilot in GitHub.com</div> <div>You can use Copilot Chat in GitHub.com and Copilot for pull requests. Copilot code review and preview features are only available for paid licenses. Learn more about Copilot in GitHub.com. Learn more about Copilot in GitHub.com</div>	Enabled
<div>Copilot in the CLI</div> <div>You can use GitHub Copilot for assistance in terminal.</div>	Enabled
<div>Copilot in GitHub Desktop</div> <div>You can use GitHub Copilot for assistance in GitHub Desktop.</div>	Enabled
<div>Copilot Chat in the IDE</div> <div>You can use GitHub Copilot Chat in the IDE to explain a piece of code, propose bug fixes, or generate unit tests in a chat interface from your editor.</div>	Enabled
<div>Copilot Chat in GitHub Mobile</div> <div>You can use Copilot Chat in GitHub Mobile personalized to a codebase.</div>	Enabled
<div>Copilot can search the web</div> <div>Copilot can answer questions about new trends and give improved answers, via Bing. See Microsoft Privacy Statement</div>	Enabled
<div>Anthropic Claude 3.5 Sonnet in Copilot</div> <div>You can use the latest Claude 3.5 Sonnet model. Learn more about how GitHub Copilot serves Claude 3.5 Sonnet.</div>	Enabled
<div>Anthropic Claude 3.7 Sonnet in Copilot</div> <div>You can use the latest Anthropic Claude 3.7 Sonnet model.</div>	Enabled
<div>Anthropic Claude Sonnet 4 in Copilot</div> <div>You can use the latest Anthropic Claude Sonnet 4 model. Learn more about how GitHub Copilot serves Anthropic Claude Sonnet 4.</div>	Enabled
<div>Google Gemini 2.0 Flash in Copilot</div> <div>You can use Google's Gemini 2.0 Flash model in Copilot. Learn more about the public preview of Gemini 2.0 Flash.</div>	Enabled
<div>Google Gemini 2.5 Pro in Copilot</div> <div>You can use the latest Google Gemini 2.5 Pro model. Learn more about how GitHub Copilot serves Google Gemini 2.5 Pro.</div>	Enabled
<div>OpenAI o4-mini in Copilot</div> <div>You can use the latest OpenAI o4-mini model. Learn more about how GitHub Copilot serves OpenAI o4-mini.</div>	Enabled
<div>Dashboard entry point</div> <div>Allows instant chatting when landing on GitHub.com</div>	Enabled

Privacy

<div>Suggestions matching public code (duplication detection filter)</div> <div>Copilot can allow or block suggestions matching public code. Learn more about code suggestions.</div>	Allowed
<div><input checked="" type="checkbox"/> Allow GitHub to use my data for product improvements</div>	



2.copilot部分

在扩展中下载**GitHub Copilot**

安装成功后右下角会弹出登录提醒，登录刚刚开通copilot的账号即可

随后即可使用copilot的所有功能啦！

包括**智能代码补全**、**编辑器内联聊天**、**右侧聊天栏**、以及**agent mode**

3.python基础语法讲解

此处如果想要了解python的进一步应用，可以参考科协同学编写的文档：[DA科协python进阶用法详解文档](#)

变量和数据类型

Python 是一种**动态类型**语言，这意味着你不需要提前声明变量的类型。直接赋值就可以了。

Python

```
# 整数
age = 30
# 浮点数
pi = 3.14
# 字符串
name = "Alice"
# 布尔值
is_student = True
```

Python 有多种内置的数据结构，非常实用：

- **列表 (List)**：可变序列，用方括号 `[]` 表示。

Python

```
fruits = ["apple", "banana", "cherry"]
```

- **元组 (Tuple)**：不可变序列，用圆括号 `()` 表示。

Python

```
coordinates = (10, 20)
```

- **字典 (Dictionary)**：键值对的集合，用花括号 {} 表示。

Python

```
person = {"name": "Bob", "age": 25}
```

缩进与代码块

这是 Python 最独特的语法特点。它**使用缩进来表示代码块**，而不是像其他语言那样使用花括号 {}。

- 通常使用 **4 个空格**进行缩进。
- 缩进必须保持一致。

Python

```
# 这是一个代码块的例子
if age > 18:
    print("成年人") # 这里的代码块需要缩进
    print("可以投票")
else:
    print("未成年人")
```

条件语句

使用 if, elif 和 else 来实现条件判断。

Python

```
score = 85

if score >= 90:
    print("优秀")
elif score >= 60:
    print("及格")
else:
    print("不及格")
```

循环语句

Python 中有两种主要的循环：

- **for 循环**：用于遍历序列（如列表、元组、字符串等）。

Python

```
for fruit in fruits:
    print(fruit)
```

- **while 循环**：当条件为真时，循环会一直执行。

Python

```
count = 0
while count < 5:
    print(count)
    count += 1
```

函数

使用 `def` 关键字来定义一个函数。

Python

```
def greet(name):
    """这个函数用于打招呼"""
    return f"你好, {name}!"

message = greet("Alice")
print(message) # 输出: 你好, Alice!
```

注释

使用 `#` 符号来添加单行注释。

Python

```
# 这是一行注释，解释了下面的代码
total = 100 + 50 # 可以在代码后面添加注释
```

4.linux命令行

1. 核心概念

- 终端 (Terminal): 一个程序, 让你能输入命令并看到输出。
- 命令提示符 (Prompt): 通常以 \$ 或 # 结尾, 表示你可以输入命令了。# 通常表示你拥有管理员权限 (root)。
- 语法: 大多数命令遵循 命令 [选项] [参数] 的格式。
例如: `rm -rf /` (删除所有文件)(千万别做这个, “删库跑路”, 会删掉根目录下的所有文件, 造成**不可逆**的危害, 不过没有sudo权限的话应该也做不了嘿嘿)
- 命令: 你想执行的操作, 如 `ls`。
- 选项 (Options/Flags): 用 - 或 -- 开头, 用来修改命令的行为, 如 `ls -l`。
- 参数 (Arguments): 命令要操作的对象, 如 `ls my_folder`。

2. 文件和目录管理

pwd (Print Working Directory)

作用: 显示你当前所在的目录。

示例: `pwd`

ls (List)

作用: 列出当前目录下的文件和子目录。

常用选项:

-l: 以长格式 (详细信息) 列出。

-a: 列出所有文件, 包括隐藏文件 (以 . 开头)。

-h: 结合 -l 使用, 以人类可读的格式显示文件大小。

示例: `ls -lah`

cd (Change Directory)

作用: 切换到另一个目录。

常用参数:

`cd /path/to/directory`: 切换到指定路径。

`cd ..`: 返回上一级目录。

`cd ~`: 返回用户主目录 (Home)。

`cd -`: 返回上一次所在的目录。

示例: `cd Documents`

mkdir (Make Directory)

作用: 创建一个新目录。

示例: `mkdir new_folder`

rm (Remove)

作用: 删除文件或目录。请谨慎使用, 删除的文件通常无法恢复。

常用选项:

`-r`: 递归删除, 用于删除目录及其内容。

`-f`: 强制删除, 不进行任何提示。

示例: `rm file.txt`, `rm -rf folder/`

cp (Copy)

作用: 复制文件或目录。

常用选项: `-r`: 递归复制, 用于复制目录。

示例: `cp file.txt new_file.txt`, `cp -r folder/ new_folder/`

mv (Move)

作用: 移动或重命名文件/目录。

示例: `mv old_file.txt new_file.txt` (重命名), `mv file.txt /tmp/` (移动)

cat (Concatenate)

作用: 显示文件内容。适用于查看短文件。

示例: `cat file.txt`

less

作用: 分屏查看文件内容。适用于查看长文件, 按 `q` 退出。

示例: `less long_file.log`

head / tail

作用：head 显示文件开头几行，tail 显示文件末尾几行。

常用选项：-n [行数]

示例：head -n 5 file.txt, tail -n 10 file.txt

touch

作用：创建一个空文件，或更新一个文件的修改时间。

示例：touch new_empty_file.txt

sudo (Super User Do)

作用：以管理员权限执行命令。(可能会让你输密码)

示例：sudo apt update（在 Debian/Ubuntu 系统中更新软件源）