

A Guide For CS 61A: Structure and Interpretation of Computer Programs

1.Introduction

CS61A作为61系列基础课程的第一门课程，是一门计算机入门导论课程，伯克利大一新生的第一门计算机课程。该课程主要使用Python语言，简要介绍了计算机的各种概念，范围广而涉猎不深，包括高阶函数，抽象，递归和树，OOP，简单的SQL语句，Scheme语法和解释器等概念。

先决条件：如果你甚至没有任何编程经验，且时间充裕，可以考虑[CS 10: The Beauty and Joy of Computing](#)

课程大纲

Chapter 1: Building Abstractions with Functions

主要介绍 编程入门，函数定义与设计，高阶函数和递归等等。

Chapter 2: Building Abstractions with Data

主要介绍 数据抽象，面向对象编程等等

Chapter 3: Interpreting Computer Programs

主要介绍 函数式编程，解释器开发等等

Chapter 4: Data Processing

主要介绍 声明式编程，分布式计算与并行计算

课程主页：

网址：<https://cs61a.org/>

Lecture：

也就是这门课的课程视频，不幸的是Spring2022的课程视频都放在bcourse(Berkeley官方发布视频的地方)上，因此**Spring2022**的视频是看不了的，该Spring2022网站上的Lecture1可以播放，但是内容是对课程政策的一些介绍，没有知识点。

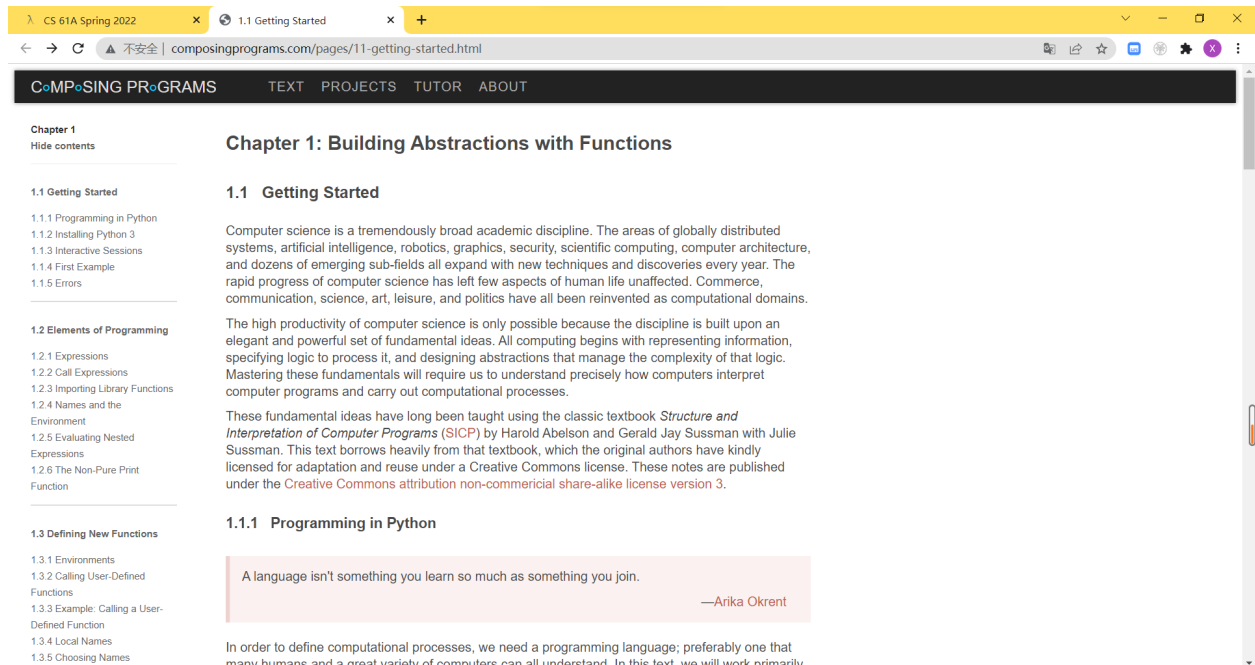
For Public：推荐看Fall2020的视频，Fall2020的所有资源公开度最高，课程视频可在YouTube上播放。如果Fall2020：<https://inst.eecs.berkeley.edu/~cs61a/fa20/>你能看任意年份的视频，推荐看 Instructor**John DeNero** 任教的，讲的比 **Pamela Fox** (也就是最新年份的导师)好

TextBook

这门课的在线教科书

网址：<http://composingprograms.com/pages/11-getting-started.html>

<https://cs61a.org/lab/lab00/#windows>



由课程教授John DeNero亲自编写，基于MIT的经典教材*Structure and Interpretation of Computer Programs*(SICP), 用Python这门语言进行改编，教科书与课程Lecture所讲授的知识内容大体一样，略有扩展，如果时间紧迫在看完Lecture后可以不看textbook

If you are good at Chinese :

<https://wizardforcel.gitbooks.io/sicp-py/content/1.1.html>

SICP的翻译版GitBookWindows' Subsystem For Linux 2

Labs , Discuss, HomeWork, Projects

CS61a总共4个项目，10个homework，14个discussion，14个lab，包含最著名的Project : Scheme

对于Discuss，在看完Lecture之后就可以做了，做完还可以听视频讲解，Discuss会讲解知识难点和考试例题，且会明确给出solution

对于 Labs等等，最重要的是理解题意，例如 Hog , Cat 等 Assignments需要反复读题

当你做完Labs之后，可以在网站上找到对应的标准答案Solution

对于新手，千万不要担心太难而不敢下手，所有作业均有完善的代码框架，详细的作业说明文档，同时每个Project也都有详尽的handout文档，你只需要应用目前学到的一些知识，就可以做出一个自己的项目

Note : 由于每学期的Lab和HomeWork的答案在下学期前会下架, 因此要及时保存 Solutions, 推荐做 Spring 2022 (也就是最新学期)的Labs和HW, 而听课看视频则看Fall **Anaconda2020**

关于评测: 不同于CS61B 拥有Gradescope的 Autograder , CS61A拥有完整的本地测试 , 全自动的评分脚本 , 相比之下更加方便 !

2. Getting Start

接下来让我们配置CS61a的课程所需环境, 考虑到上该门课的人可能是刚入门计算机的小白, 可能不太熟悉各种终端下的Command Line , 因此对环境进行最简化配置 以下环境配置均在**Windows**下进行

Spring 2022 课程官方推荐Windows下安装WSL2(), 具体流程可参考

对于 MacOS, Linux , 参考<https://cs61a.org/lab/lab00/#install-a-terminal>

由于WSL2涉及多种Linux下的Command Line, 我们使用一种更简单的Bash , 这也是Fall 2020版本推荐的做法, 如果你想参考详细过程 , 请访问

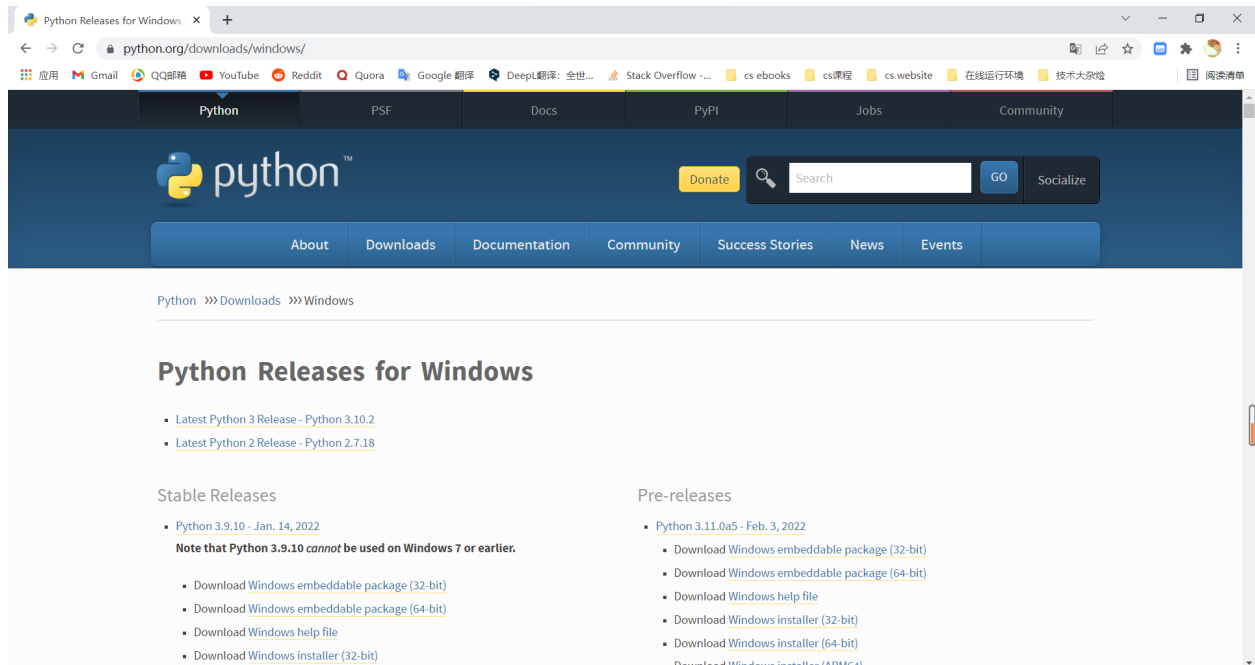
<https://inst.eecs.berkeley.edu/~cs61a/fa20/lab/lab00/#install-a-terminal>。

A WalkThroug

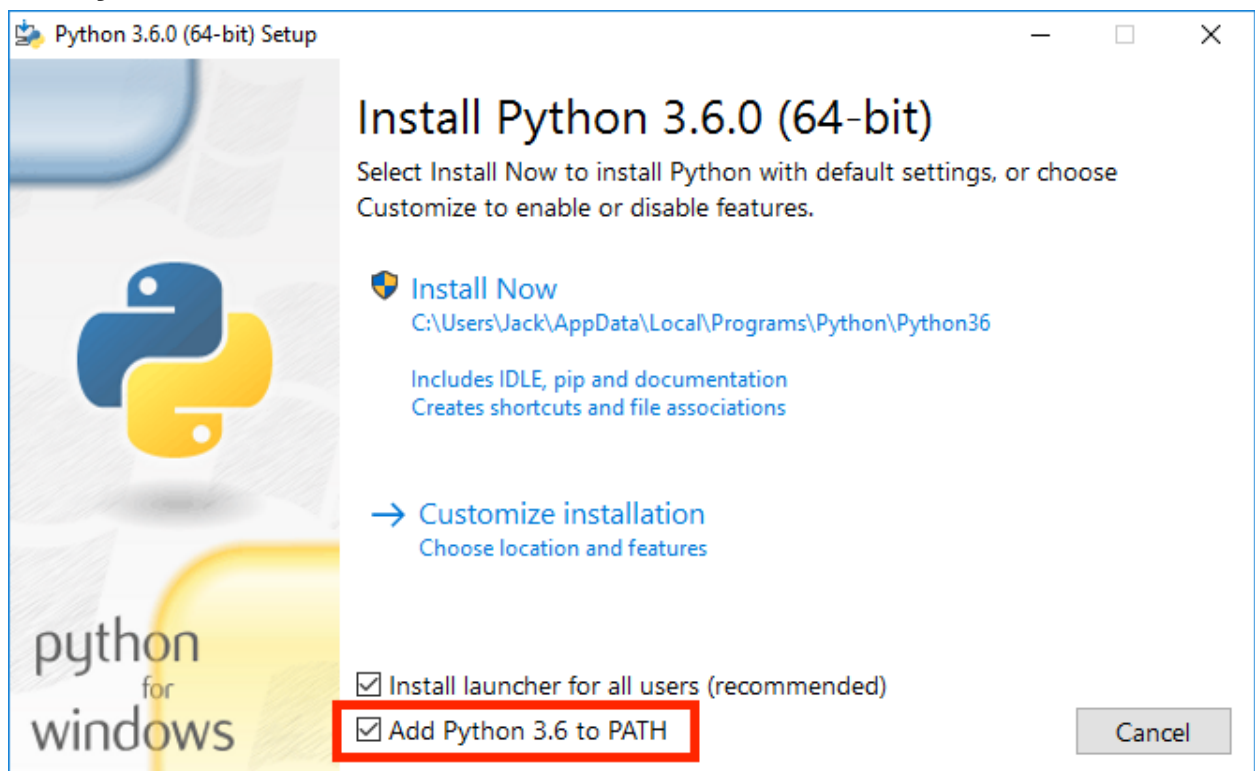
环境配置的视频教程 : <https://www.youtube.com/watch?v=YlothkvwsJo>

Install Python3

来到Python官网 : <https://www.python.org/downloads/windows/>

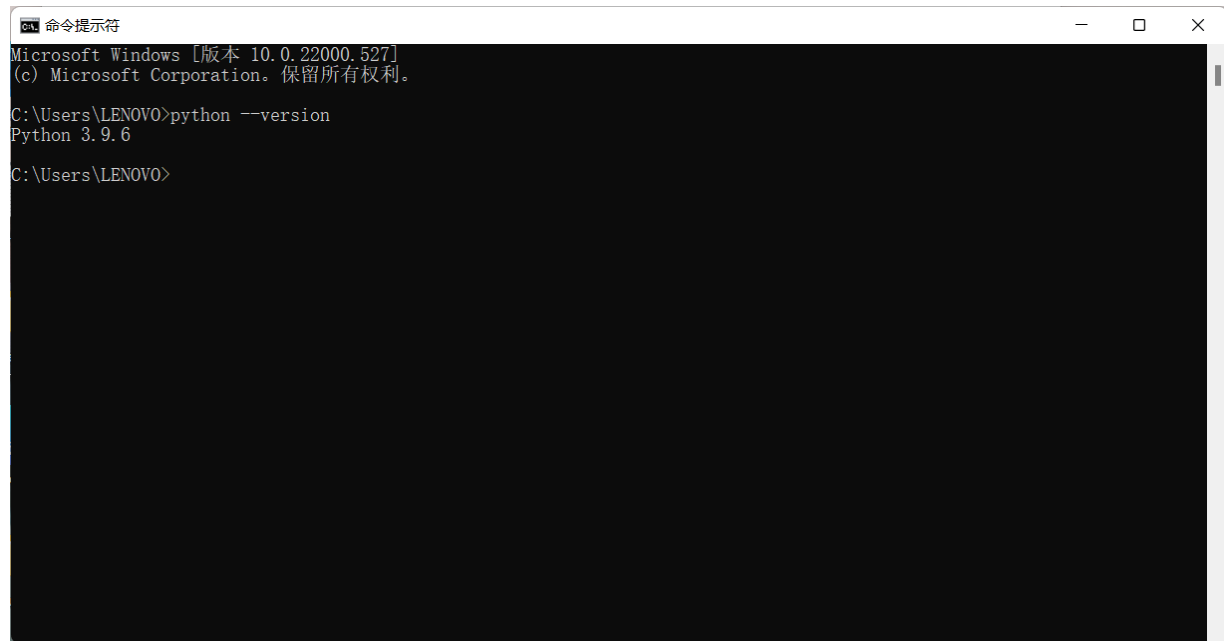


选择适合自己的版本(32bit / 64bit), 下载安装Python ,你可以使用默认选项 ,但是请选择 **Add Python 3.6 to PATH**



下载完成后打开windows cmd , 运行

```
python --version
```



```
命令提示符
Microsoft Windows [版本 10.0.22000.527]
(c) Microsoft Corporation。保留所有权利。

C:\Users\LENOVO>python --version
Python 3.9.6

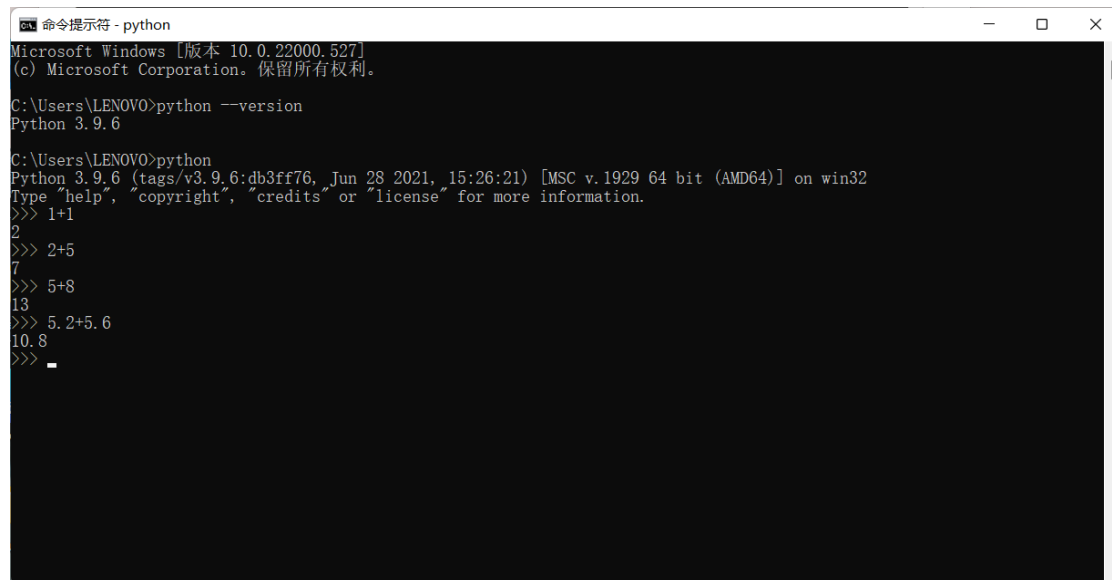
C:\Users\LENOVO>
```

输出版本号则说明安装成功

之后可以输入命令

```
python
```

运行python, 可以看到 >>> 提示符, 可以输入一些简单的数字进行运算



```
命令提示符 - python
Microsoft Windows [版本 10.0.22000.527]
(c) Microsoft Corporation。保留所有权利。

C:\Users\LENOVO>python --version
Python 3.9.6

C:\Users\LENOVO>python
Python 3.9.6 (tags/v3.9.6:db3ff76, Jun 28 2021, 15:26:21) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> 1+1
2
>>> 2+5
7
>>> 5+8
13
>>> 5.2+5.6
10.8
>>> _
```

(optional) 的安装

(虽然可能对新手不太友好但是还是推荐使用anaconda，防止以后出现冲突问题)

安装anaconda (群文件有, 网速慢的可以去清华镜像站

<https://mirrors.tuna.tsinghua.edu.cn/anaconda/archive/?C=M&O=A>)

配置环境变量 (参考https://blog.csdn.net/weixin_42403127/article/details/88363533)

初学者只要在 CMD 或

```
C:\Users\MBF>conda
usage: conda-script.py [-h] [-V] command ...

conda is a tool for managing and deploying applications, environments and packages.

Options:
  -h, --help            show this help message and exit
  -V, --version          display conda's version number and exit

positional arguments:
  command
  clean                Remove unused packages and caches.
  compare              Compare packages between conda environments.
  config               Modify configuration values in .condarc. This is modeled after the git config
                        command. Writes to the user .condarc file (C:\Users\MBF\.condarc) by default.
  create               Create a new conda environment from a list of specified packages.
  help                 Displays a list of available conda commands and their help strings.
  info                 Display information about current conda install.
  init                 Initialize conda for shell interaction. [Experimental]
  install              Installs a list of packages into a specified conda environment.
  list                 List linked packages in a conda environment.
  package              Low-level conda package utility. (EXPERIMENTAL)
  remove               Remove a list of packages from a specified conda environment.
  uninstall            Alias for conda remove.
  run                  Run an executable in a conda environment. [Experimental]
  search               Search for packages and display associated information. The input is a MatchSpec,
                        a query language for conda packages. See examples below.
  update               Updates conda packages to the latest compatible version.
  upgrade              Alias for conda update.
```

者 Git Bash 输入 conda 回车后出现如下字样表示安装成功

输入 conda activate , 出现 (base) 字样即可正常使用python

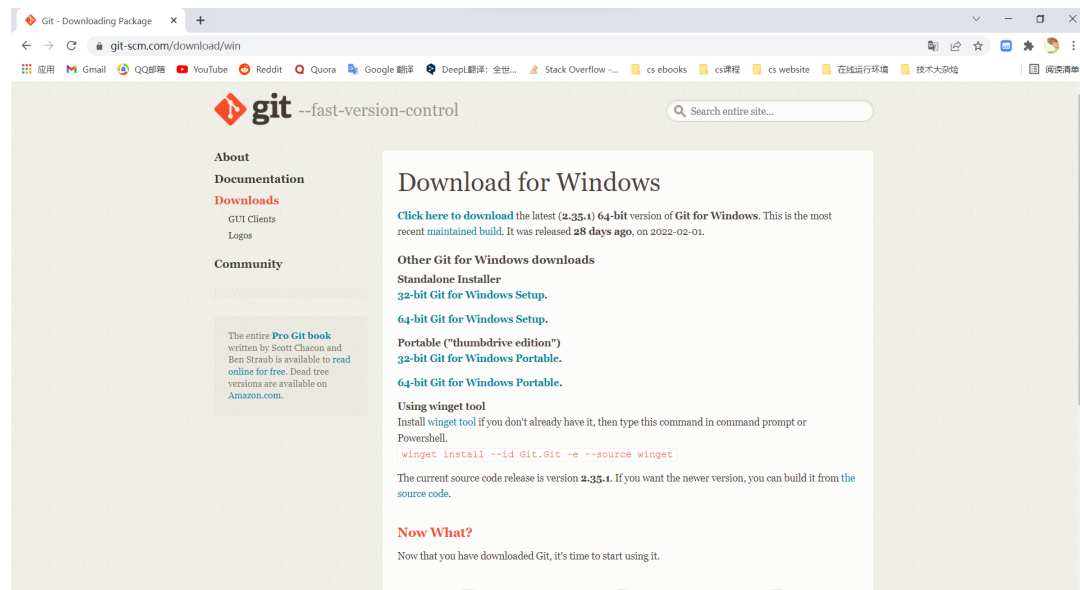
```
Microsoft Windows [版本 10.0.19043.1526]
(c) Microsoft Corporation。保留所有权利。

C:\Users\<img alt="user icon" data-bbox="210 165 245 180">>conda activate

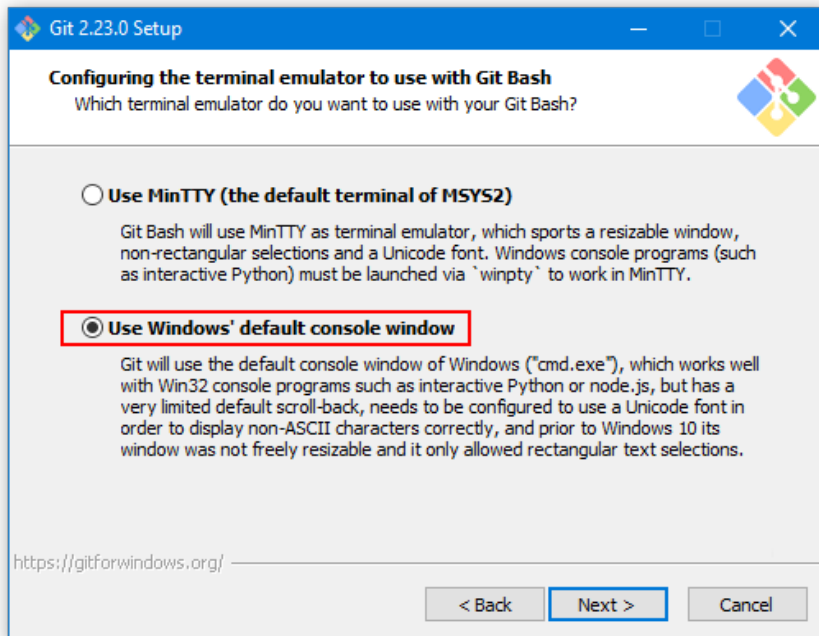
(base) C:\Users\<img alt="user icon" data-bbox="280 195 315 210">>
```

Install Git

进入Git官网：<https://git-scm.com/download/win>



选择合适的版本下载，你可以使用默认选项 但有一个例外：在 **Configuring the terminal emulator to use with Git Bash** 步骤 中选择 **Use Windows' default console window**，这个非常重要！如果你不选择此选项，你的终端将无法使用 Python！



安装完毕后，鼠标右键选择"Git Bash Here"打开

```
git --version
```

查看git版本来验证是否安装成功，如果出现问题请重新安装

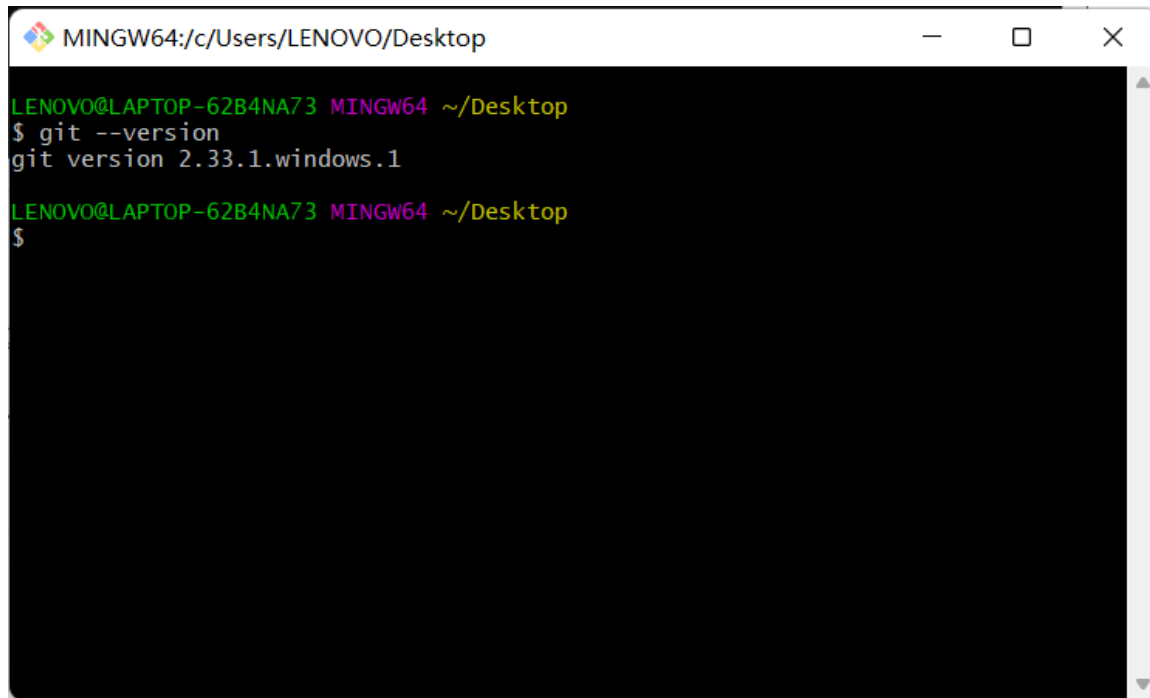
正常显示的结果应该类似这样

```
$ git --version
git version 2.29.2.windows.3
```

第一次使用Git时，系统会要求你注册一个账户

```
git config -global user.name 'xxxxx'
git config -global user.email 'xxx@xx.xxx'
```

将'xxxxx'换成自己的名字 'xxx@xx.xxx'换成自己邮箱即可，这里x的位数是随便打的

A screenshot of a Windows terminal window titled "MINGW64:/c/Users/LENOVO/Desktop". The terminal shows a user prompt "LENOVO@LAPTOP-62B4NA73 MINGW64 ~/Desktop" followed by the command "\$ git --version" and its output "git version 2.33.1.windows.1". The prompt is repeated below the output.

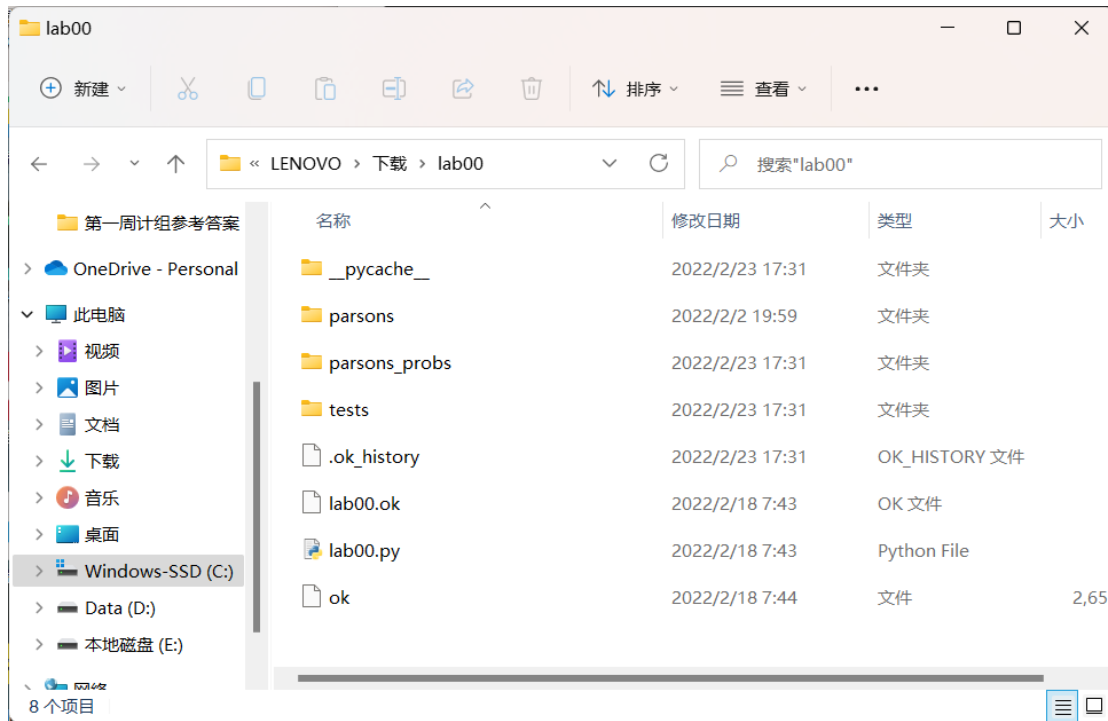
```
MINGW64:/c/Users/LENOVO/Desktop  
LENOVO@LAPTOP-62B4NA73 MINGW64 ~/Desktop  
$ git --version  
git version 2.33.1.windows.1  
LENOVO@LAPTOP-62B4NA73 MINGW64 ~/Desktop  
$
```

完成配置

Code for Lab

- Python Basics

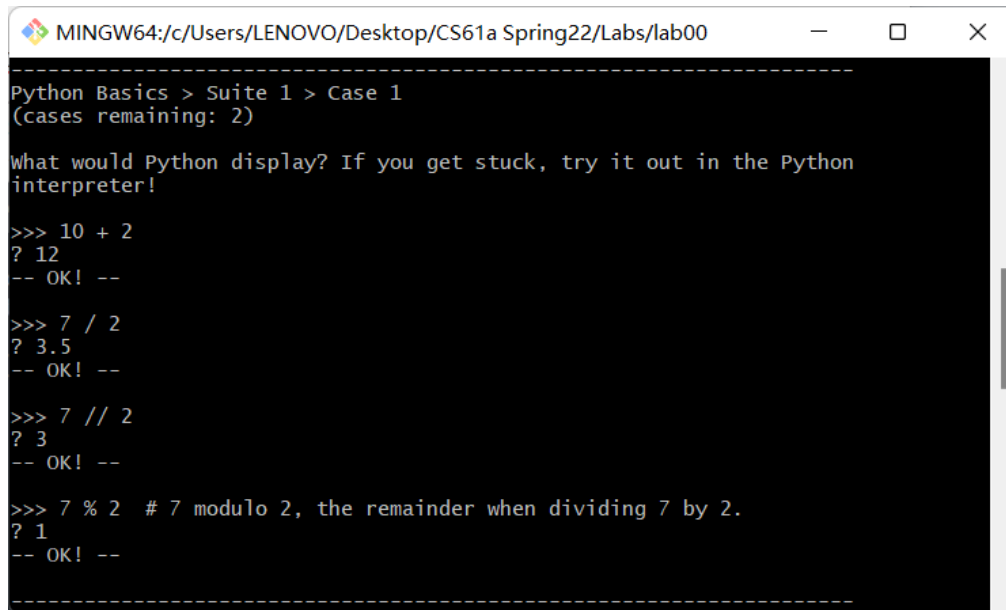
进入课程官网下载[lab00.zip](#), 在本地创建一个新文件夹用以保存课程的labs,Hw等等, 将下载好的文件解压至本地文件夹, 你应该可以看到以下画面:



右键选择"GitBash Here", 打开GitBash, 输入

```
python ok -q python-basics -u --local
```

开始基本的python语法测试:



```
MINGW64:/c:/Users/LENOVO/Desktop/CS61a Spring22/Labs/lab00
-----
Python Basics > Suite 1 > Case 1
(cases remaining: 2)

What would Python display? If you get stuck, try it out in the Python
interpreter!

>>> 10 + 2
? 12
-- OK! --

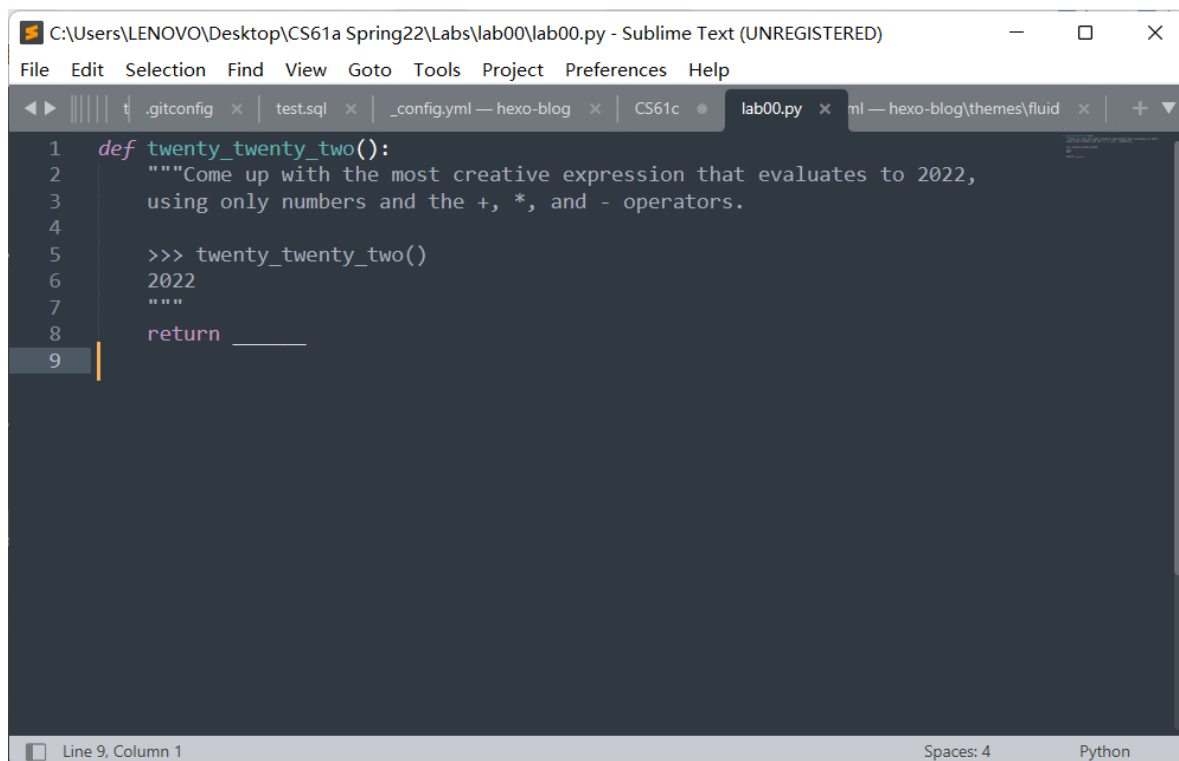
>>> 7 / 2
? 3.5
-- OK! --

>>> 7 // 2
? 3
-- OK! --

>>> 7 % 2 # 7 modulo 2, the remainder when dividing 7 by 2.
? 1
-- OK! --
-----
```

- **Lab00**

之后，回到文件夹，使用编辑器 (随你喜好，VS Code, Sublime Text, Pycharm,甚至记事本均可) 打开lab00.py



```
C:\Users\LENOVO\Desktop\CS61a Spring22\Labs\lab00\lab00.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
. .gitconfig x test.sql x _config.yml — hexo-blog x CS61c lab00.py x ml — hexo-blog\themes\fluid x + v
1 def twenty_twenty_two():
2     """Come up with the most creative expression that evaluates to 2022,
3     using only numbers and the +, *, and - operators.
4
5     >>> twenty_twenty_two()
6     2022
7     """
8     return _____
9
```

可以看到该题目的要求，使用加减乘除，以最具创造力的形式返回值 2022，这里我们就简单地返回 2022

```
def twenty_twenty_two():  
    """Come up with the most creative expression that evaluates to 2022,  
    using only numbers and the +, *, and - operators.  
  
    >>> twenty_twenty_two()  
    2022  
    """  
    return 2022
```

然后保存文件，继续在Git Bash中，执行

```
python ok --local
```



The screenshot shows a Windows command prompt window with the title bar "MINGW64:/c/Users/LENOVO/Desktop/CS61a Spring22/Labs/lab00". The window contains the following text:

```
Running tests  
  
-----  
Doctests for ilove61a  
  
>>> from parsons_probs.ilove61a import *  
>>> ilove61a() # .Case 1  
  
# Error: expected  
#      'I love CS 61A!'  
# but got  
  
:( Test Case 1 failed  
  
-----  
  
Test summary  
  3 test cases passed before encountering first failed test case  
Cannot backup when running ok with --local.  
LENOVO@LAPTOP-62B4NA73 MINGW64 ~/Desktop/CS61a Spring22/Labs/lab00  
$
```

出现以下提示

```
Test summary  
  3 test cases passed before encountering first failed test case
```

则说明完成lab00

3. Additional Summary

- 总结做题步骤:
官网下载lab，解压到本地，使用编辑器写代码，完成后打开GitBash，运行

```
python ok --local
```

- Note 1: 如果你看到课程网站上提示你使用 **python3**，但是在GitBash中请使用 **python**
- Note 2: 本地评测即 **ok** 命令，务必添加 **--local** 以保证在本地进行测试，如果没有该option，很肯会要求你输入Berkeley邮箱，或是打开自动化网上测评系统 <https://okpy.org/> 需要Enroll该门课程的学生才能进行网上评测，本地测试与网上一样。
- Note 3: **-u** option是解锁本地测试的意思
- 关于Parsons问题：需要在ok.org上进行评测，如果你执行 `python parsons` 则会要求你输入Berkeley的邮箱：
如果您不打算远程访问实验室计算机或打印机，则不需要它们，本地足够

How To Find A Solution

当你完成Assignments, 请到课程网站处寻找

| Calendar | | Start |
|----------|--|--------------|
| | Disc 00: Getting Started | |
| | Solutions | |
| | Lab 00: Getting Started (Optional) | Due Thu 1/27 |
| | Solutions | |
| | | H D S |
| | Lab 01: Variables & Functions, Control | |
| | Due Wed 1/26 | |
| | Solutions | |
| | Disc 01: Control, Environment Diagrams | |
| | Solutions | |
| | Exam Prep 01: Variables & Functions, Control | H D S |
| | Recording Notes | |
| | Lab 02: Higher-Order Functions, Lambda Expressions | H |
| | Due Wed 2/2 | |
| | Solutions | |
| | Disc 02: Environment Diagrams, Higher-Order Functions | |
| | Solutions | |
| | Exam Prep 02: Higher-Order Functions, Environment Diagrams | |
| | Recording Notes | |

A GoodNote Recommendation

<https://rigelj.github.io/tags/CS61A/>

最后,令人兴奋的是, CS61a的四个Project里面有三个都是做游戏, 你一定很感兴趣, 开始你的CS61A之旅吧!