



**Feimax**



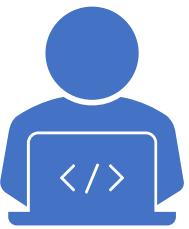
# **Lab 3**

# **Introduction**

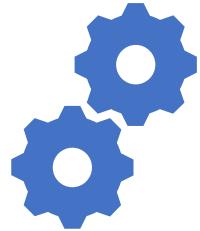
# **to Python**

CAN201 – Introduction to Networking

# Lab 3



How to use the PyCharm on labs' computer and install python and PyCharm for your computer.



Write a code using raw python and build a virtual Python runtime environment using *Virtualenv*



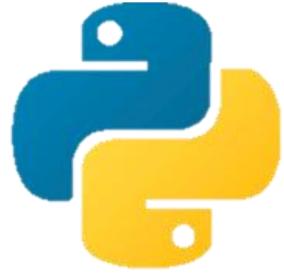
Basic Python Syntax:  
Variables, if, loop...

# Python

Sep 2024	Sep 2023	Change	Programming Language	Ratings	Change
1	1		 Python	20.17%	+6.01%
2	3		 C++	10.75%	+0.09%
3	4		 Java	9.45%	-0.04%
4	2		 C	8.89%	-2.38%
5	5		 C#	6.08%	-1.22%
6	6		 JavaScript	3.92%	+0.62%
7	7		 Visual Basic	2.70%	+0.48%
8	12		 Go	2.35%	+1.16%
9	10		 SQL	1.94%	+0.50%
10	11		 Fortran	1.78%	+0.49%

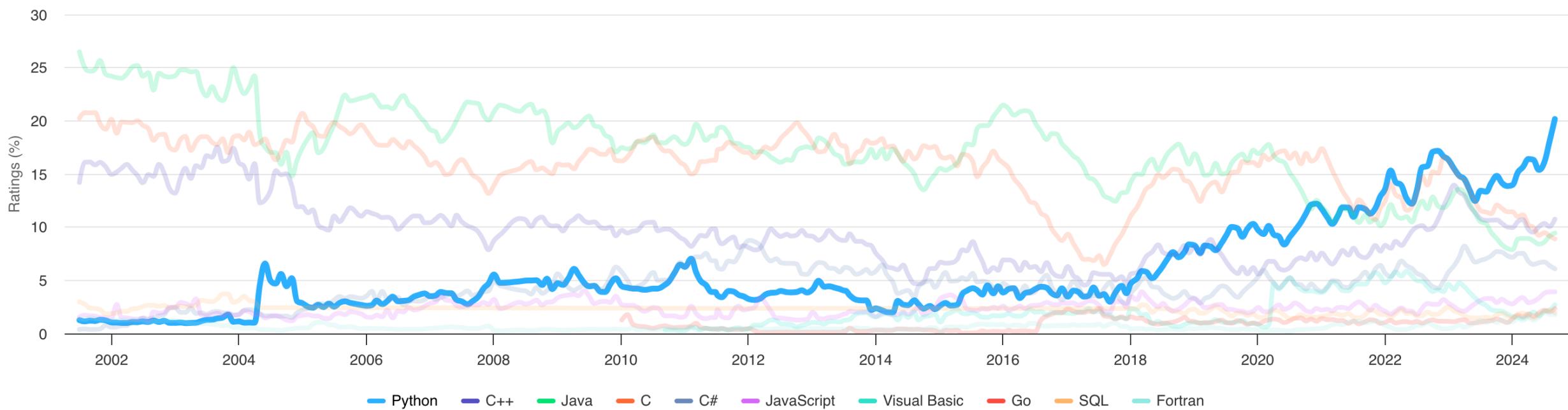
TIOBE September 2024

# Python



TIOBE Programming Community Index

Source: [www.tiobe.com](http://www.tiobe.com)



September 2024

# Python

- Python is a programming language that lets you work more **quickly** and integrate your systems more **effectively**.

**Python is powerful... and fast;  
plays well with others;  
runs everywhere;  
is friendly & easy to learn;  
is Open.**

These are some of the reasons people who use Python would rather not use anything else.



# What exactly can Python do?

What can python do?

- Almost everything
- AI (+ Tensorflow, PyTorch, Scikit-Learn)
- Data mining (+ Numpy, Pandas, Matplotlib, Jupyter Notebook)
- Scientific Computing (+ Scipy)
- Computer vision (+ OpenCV)
- Web and network (+Django, web.py, flask, socket, requests)
- Web Crawler (+ beautifulsoap, selenium)
- To exe (+ Pyinstaller)
- GUI (+ tkinter, wxPython)
- ...



# Features



Simple and rich syntax – list, tuple, dict ...



Without RAM management;



A rich library of module packages (pypi) -  
numpy, scipy



C/C++ connection – glue language



.....

# How to Start

- Difficulty in starting - so many ways to start!
  - Raw Python
  - Virtualenv
  - Anaconda
  - IPython: Interactive shell for python
  - Jupyter: GUI for Ipython
- Spyder
- PyCharm
- VsCode
- IDLE: idle...



The image shows a person's hand pointing towards a computer screen. The screen displays a block of Python code. The code appears to be part of a larger script, likely for a 3D modeling application like Blender, given the context of the slide title. The visible code includes logic for mirroring objects across different axes (X, Y, Z) and handling selection operations.

```
mirror_mod = modifier_ob
# set mirror object to mirror
mirror_mod.mirror_object = ob
operation == "MIRROR_X":
    mirror_mod.use_x = True
    mirror_mod.use_y = False
    mirror_mod.use_z = False
operation == "MIRROR_Y":
    mirror_mod.use_x = False
    mirror_mod.use_y = True
    mirror_mod.use_z = False
operation == "MIRROR_Z":
    mirror_mod.use_x = False
    mirror_mod.use_y = False
    mirror_mod.use_z = True

#selection at the end -add
ob.select= 1
ier_ob.select=1
ntext.scene.objects.active = ("Selected" + str(modifier))
iror_ob.select = 0
bpy.context.selected_objects = data.objects[one.name].sel
int("please select exactly one object")
-- OPERATOR CLASSES ----
types.Operator):
    X mirror to the selected object.mirror_mirror_x"
    or X"
context):
    ext.active_object is not ob
```

# Run Python using different ways

- On Labs Ubuntu
  - RAW Python
  - Virtual environment
- On your computer(optional, strongly recommended)
  - RAW Python
  - Virtual environment

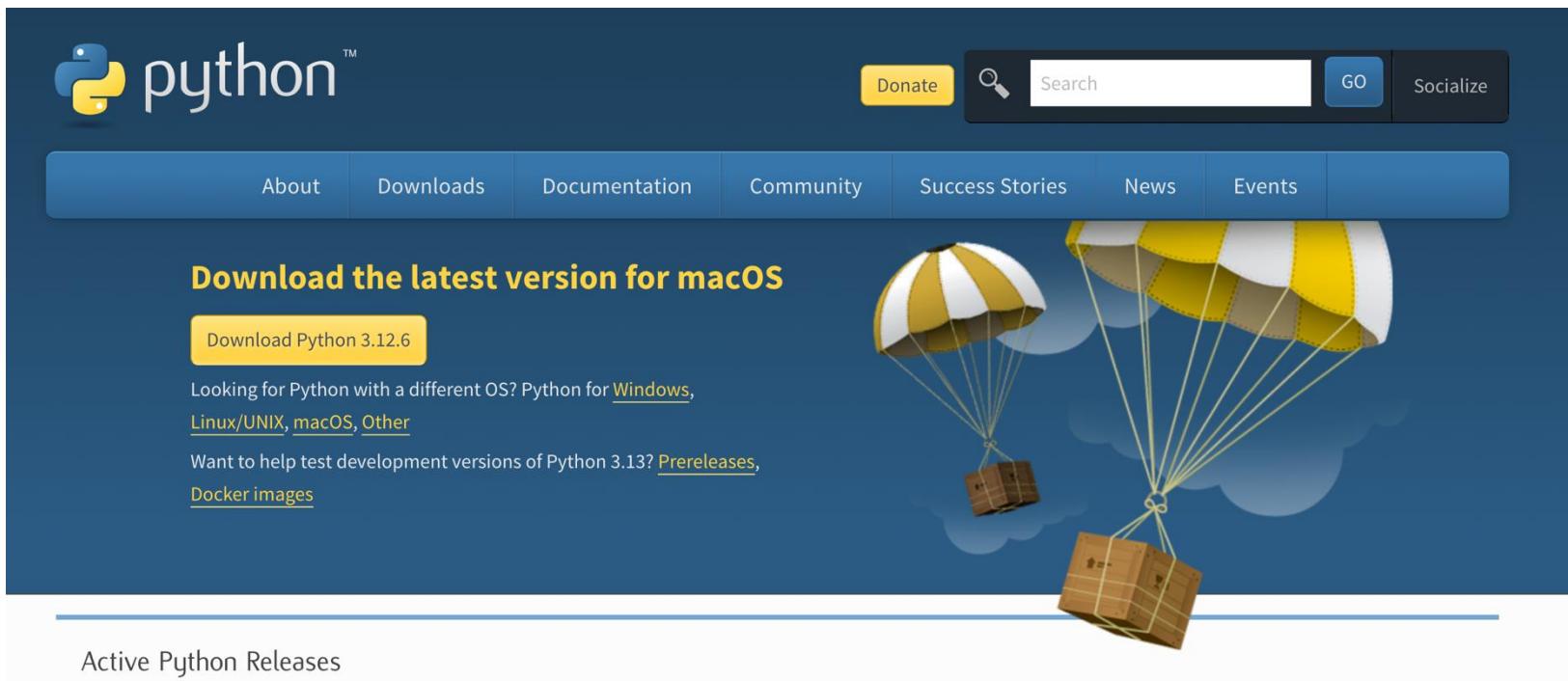
# DEMO

- On Labs Ubuntu
- <https://box.xjtu.edu.cn/f/2375185209654f3882fa/>



# How to get Python

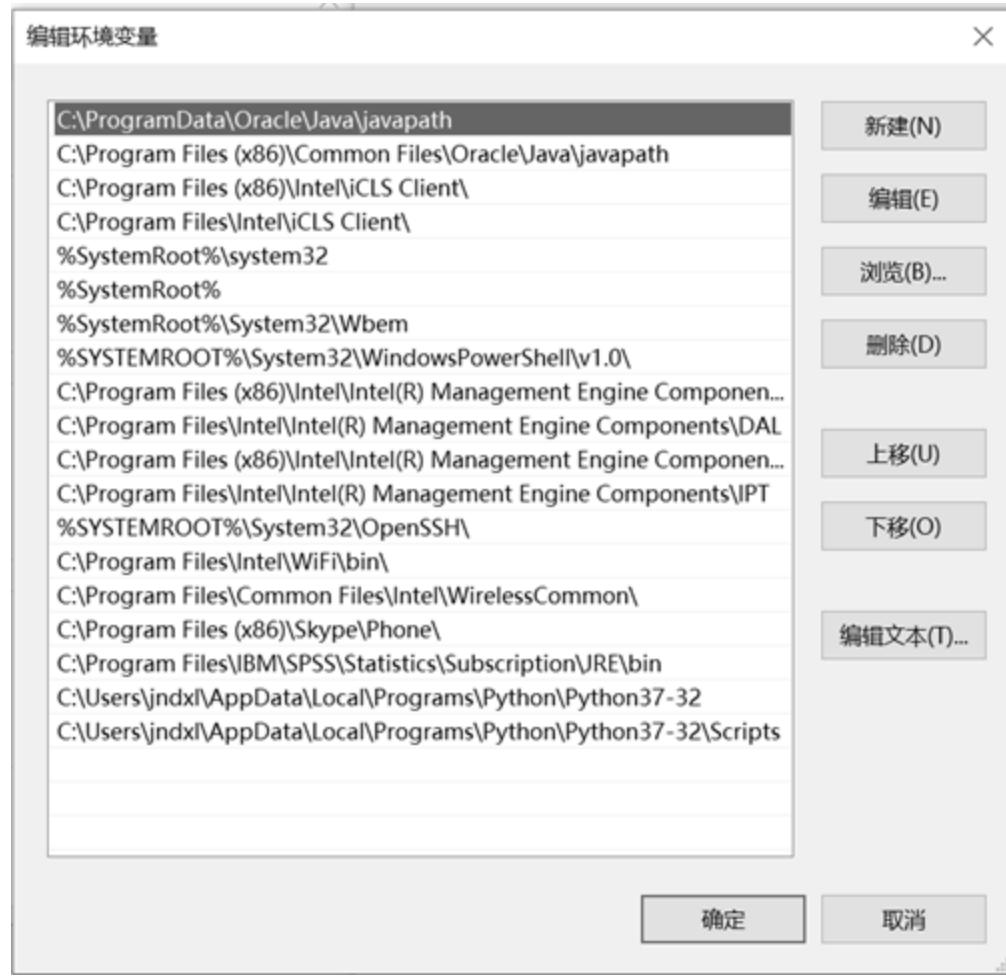
- <https://www.python.org/downloads/>



# Windows

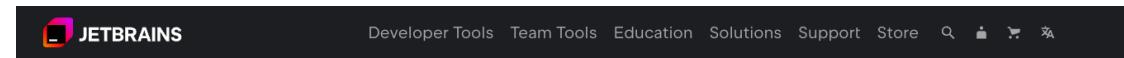
- Press Win key, select the gear icon on the left - Settings
- Search "Edit the system environment variables "
- Select " Edit the system environment variables "
- At the bottom right of the interface, select "Environment Variables"
- Find Path, in the list of Path
- Add the folder of python and pip
- Normally, path of python is
- C:\Users\xxx.xxx\AppData\Local\Programs\Python\Python37
- Path of pip:
- C:\Users\xxx.xxx\AppData\Local\Programs\Python\Python37\Scripts





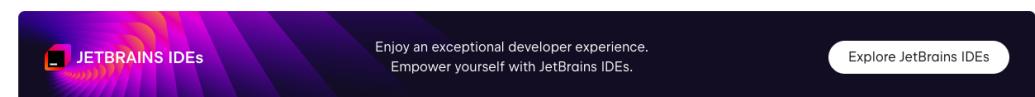
# How to get PyCharm

- <https://www.jetbrains.com/products/>



## Find the right tool

Whichever technologies you use, there's a JetBrains tool to match





**Fleet**  
More than a Code Editor  
[Explore Fleet](#)  
[Download](#)



**Toolbox App**  
2.4.2  
A control panel for your tools and projects  
[Explore Toolbox App](#)  
[Download](#)



**IntelliJ IDEA**  
2024.2.2  
The Leading Java and Kotlin IDE  
[Explore IntelliJ IDEA](#)  
[Download](#) [Buy](#)



**PyCharm**  
2024.2.2  
The Python IDE for data science and web development  
[Explore PyCharm](#)



**WebStorm**  
2024.2.2  
The JavaScript and TypeScript IDE  
[Explore WebStorm](#)



**PhpStorm**  
2024.2.2  
The PHP IDE  
[Explore PhpStorm](#)

**Filters**

Languages

Search

C

C++

C#

Dart

DSL

F#

Go

Groovy

HTML

Java

JavaScript, TypeScript

Kotlin

Objective-C

Technologies



Version: 2021.2.1  
Build: 212.5080.64  
27 August 2021

[System requirements](#)  
[Installation Instructions](#)  
[Other versions](#)  
[Third-party software](#)

## Download PyCharm

[Windows](#) [macOS](#) [Linux](#)

### Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

[Download](#)[.dmg \(Intel\)](#)

### Community

For pure Python development

[Download](#)[.dmg \(Intel\)](#)

Free

[.dmg \(Intel\)](#)

Free, open-source



PyCharm

[.dmg \(Apple Silicon\)](#)

Intel and Apple Silicon



Get the Toolbox App to download PyCharm and its future updates with ease

# Student jetbrains account

- <https://www.jetbrains.com/community/education/#students>

The screenshot shows a dark-themed landing page for free educational licenses. At the top, the title "Free Educational Licenses" is displayed in large white font. Below it, a sub-headline reads "Learn or teach coding with best-in-class development tools from JetBrains!". Three navigation buttons are visible: "For students and teachers" (highlighted with a rounded rectangle), "For schools and universities", and "For non-academic educators". A section titled "Individual licenses for students and teachers" describes how users can get free access to JetBrains IDEs for personal use at school or home. It includes a link to "Who can get free individual licenses for education" and a note about accreditation requirements. At the bottom, there is a link to check the FAQ or read the full terms.

**Free Educational Licenses**

Learn or teach coding with best-in-class development tools from JetBrains!

For students and teachers   For schools and universities   For non-academic educators

**Individual licenses for students and teachers**

Get free access to all JetBrains IDEs for personal use at school or at home.

[Who can get free individual licenses for education](#)

Students and faculty from accredited educational institutions (high schools, colleges, and universities) are welcome to apply.

Students need to be enrolled in an accredited educational program that takes one or more years of full-time study to complete.

Not sure about the license terms? [Check out the FAQ](#) or read the full terms [here](#).

# How to receive emails from JetBrains?

- Make sure your email account is right:  
xxx.xxx22@student.xjtu.edu.cn
- If you cannot find emails from JetBrains, please try to find emails from <https://spam.xjtu.edu.cn/>

# How to start to learn computer language?

1. Install a beautiful IDE
  2. Variables: how to define? Type?
  3. How to print “hello world”?
  4. How to input a number and a string? Know 1 and ‘1’ are different.
  5. Know “if” conditional statement.
  6. Know “for” loop and “while” statements.
  7. Use functions
  8. Try to adapt to the non-GUI environment.
  9. How to deal with a large number of data? (Array and other data structure)
  10. OOP if the language supports.
- .....

# Basic Python Syntax

- Variables: numbers, strings, list
- Identifier: naming rule, python keywords
- Important syntax 1: line, indentation
- Flow Control: if...elif...else; ternary operator; for and while loop

# Variables

- Variables in Python can be used without declaration.
  - Simple, fast
  - Ambiguity
- Get type of a variable
  - `type(v)`
  - `isinstance(v, type/class...)`

# Variables - String

- Many ways to have a string
  - `'foo'`, `"bar"`, `'''foobar'''`
- String prefix
  - `b`: byte array
  - `r`: raw (avoid char escaping)
  - `f`: formatted
  - `u`: unicode
- Print string
  - `print()`
  - `pprint()` # a pretty printer

# Variables - Numbers

- int:
  - Binary, Decimal, Octal, Hexadecimal:
    - `int('1001', n)` # make a base "n" number to a decimal number
    - # bin(d), oct(d), hex(d) convert a decimal number to b / o / x
  - Rounding
    - # `int(f)`, `round(f)`, `math.ceil(f)`, `math.floor(f)`
- float:
  - `float('0.15')`
- Variables precision
  - Try *numpy*

# Variables

DEMO

<https://box.xjtu.edu.cn/f/558783dd99e546dfb49c/>



# "Variables" - List

- List: a **super** array
- Different type for each item!
- Easily nest

```
[ 'a', 'ny', item, [1,2,3], [True, False, {'a': 1}]]
```

# "Variables" - List

- Define:
  - `l = []`
- Sort
  - `l.sort([key=func])`
  - unique type / multiple type
- Index
  - `l.count()`
  - `l.index(item, [start, [, end]])`
- An item exist
  - `item in l` # a boolean value

# "Variables" - List

- Put an item
  - `l.append(item)`
  - `l.extend(another_list)`
- Take away an item
  - `l.pop(position) # get and remove this item`

# List

DEMO



<https://box.xjtu.edu.cn/f/d37d529002ac42be8286/>

# Identifier

- Definition:
  - identifier ::= (letter | "\_") (letter | digit | "\_")\*
  - letter ::= "a" ... "z" | "A"..."Z"
  - digit ::= "0"..."9"
- Name of:
  - variables, functions, class, python keywords

} **BNF: Backus-Naur form**

<https://docs.python.org/3/reference/>

# Coding Style - Name

- Name your classes and functions consistently; the convention is to use UpperCamelCase for classes and lowercase\_with\_underscores for functions and methods.
- Required by PEP-8: <https://www.python.org/dev/peps/pep-0008/> (Python Enhancement Proposals)

# Identifier

- Python keywords (33):
  - `['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']`
- How to know python keywords

```
import keyword  
print(keyword.kwlist)
```

# Important syntax 1

- Line:
  - Explicit line joining: \
  - Implicit line joining: parentheses (), square brackets [] or curly braces {} can be split over more than one physical line without using backslashes
- Blank lines: spaces, tables, a comment (only)
- `pass` # not a blank line

# Important syntax 1

- Indentation:
  - Leading spaces and ~~tabs~~
  - Indentation is used to represent program blocks / scope
- Coding style:
  - Use 4-space indentation, and no tabs.
  - Wrap lines so that they don't exceed 79 characters

# Flow Control - if

- `if ...`
- `if ... else ...`
- `if ... elif ... elif ... else ...`
- Condition: support `a < x <= c`
- ternary operator: `z = x if condition else y`
- No select or switch in Python

if

# DEMO



<https://box.xjtlu.edu.cn/f/fdec60d9eb264bdcaba5/>

# Flow Control - loop

- `for item in generator:`  
`do_sth()`
- `while condition:`  
`do_sth()`
- `break` and `continue`: the soul of loops
- No `do... while`: how to simulate?

# loop

## DEMO



<https://box.xjtu.edu.cn/f/cf35c865af3b4c9c87e7/>

Thanks ”