CS305 Lab1
Basic network commands & Brief introduction to Python(1)

Dept. Computer Science and Engineering
Southern University of Science and Technology

lizz@sustech.edu.cn;
wangw6@sustech.edu.cn;

SUSTech
Southern University
of Science and Technology

LAB information

- Lab website
  - Blackboard: Computer Networks Spring 2024
  - QQ group: 721468005
- Grading criteria
  - Attendance & practice(s) on each lab class
  - Lab assignments
  - Project
- regulations on plagiarism
  - The first time the score of the assignment will be 0
  - The second time the score of the course will be 0

Topics

- Introduction to network commands(1).
  - **ipconfig**
  - **arp**
  - **nslookup**

  TIPS: some commands may vary across different operating systems

- Introduction to Python(1)
  - **Basic syntax of Python**
  - tools

Part. A

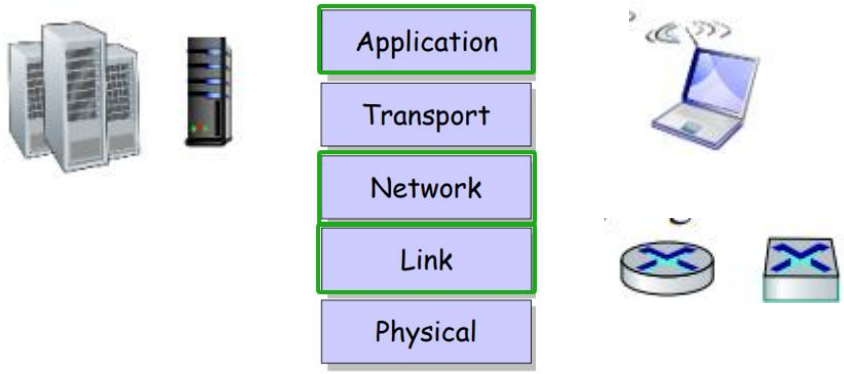
Basic network commands(1)
ipconfig, arp, nslookup

Identification of Network device(1)

- Domain name (Application Layer)
  - www.sustech.edu.cn
  - www.baidu.cn

- IP address (Network Layer)
  - IPv4
    - 32bit, dotted decimal notation
      - 157.148.69.80
      - 192.168.1.1
  - IPv6
    - 128bit, hexadecimal notation
      - 2002::4ab8:7b76
      - fe80::d2aa:775b:4ab8:7b76

- Physical/MAC address (Link Layer)
  - 48bit, hexadecimal notation
    - 8a-69-0c-51-98-66



Identification of Network device(2)

- IP address and Subnet mask(1)
  - **IP address** includes two identification codes (IDs), namely **network ID** and host ID. All hosts on the same physical network use the same network ID, and a host on the network has a corresponding **host ID**.
  - **Subnet mask** consists of 1 and 0, and 1 and 0 are consecutive, with network bits on the left, represented by the binary number "1". **The number of 1 is equal to the length of the network bits; On the right is the host bit, represented by the binary number "0"**, where the number of 0s is equal to the length of the host bit.

IP address:
1100 0000. 1010 1000. 0000 0001. 1010 1000
192.168.1.104

Subnet mask:
1111 1111. 1111 1111. 1111 1111. 0000 0000
255.255.255.0

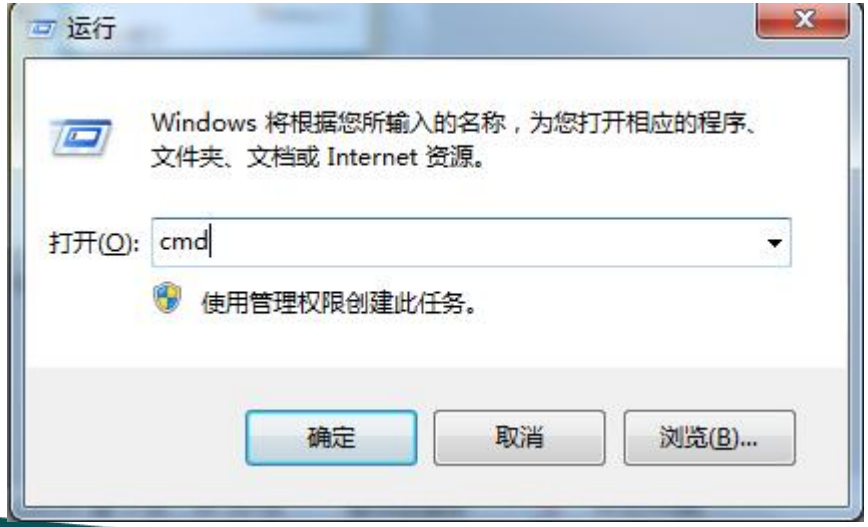
Identification of Network device(3)

- IP address and Subnet mask(2)
  - **Calculation:**
    - **The network address is obtained by performing a bitwise logical AND operation on a 32-bit subnet mask and an IP address in binary form.**
    - **The subnet mask is then bitwise reversed, and the logical AND operation is performed on the IP address binary to obtain the host address.**
    - **For example:**

IP address:	**192.168.1.104**
Subnet mask:	**255.255.255.0**
network ID:	**192.168.1.0**
host ID:	**104**

Experimental environment

- DOS terminal on Windows 10
  - Click ‘start’ on desktop -> choose ‘run’ ->input ‘cmd’ to invoke the DOS terminal on windows
  - Or Windows + R



1. ipconfig (1)

- "**ipconfig**" is usually used to show the configuration on network adapter.
  - "***ipconfig***" can display the IP address, gateway, network mask of network adapter .
  - Followed by different parameters can display different information.
  - "***ipconfig /all***" or "***ipconfig -all***" can display more information.
  - "***ipconfig /?***" or "***ipconfig -?***" can display its help information.

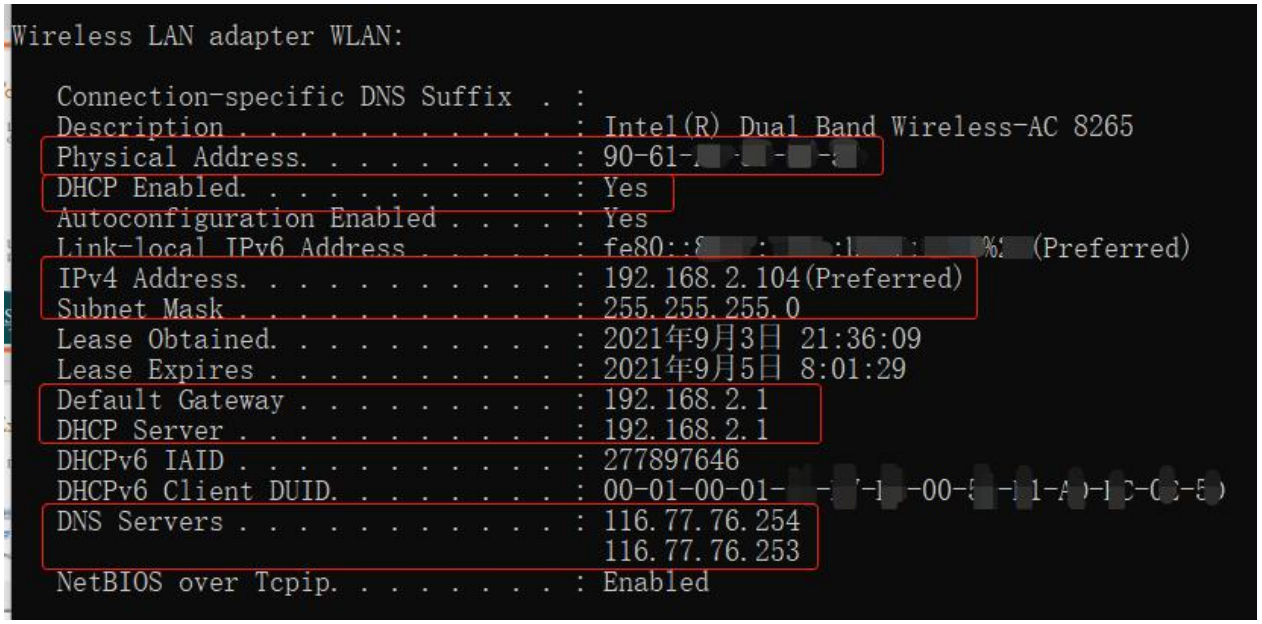
```
C:\Users\sustech>ipconfig /?

用法:
    ipconfig [/allcompartments] [/? | /all |
                                 /renew [adapter] | /release [adapter] |
                                 /renew6 [adapter] | /release6 [adapter] |
                                 /flushdns | /displaydns | /registerdns |
                                 /showclassid adapter |
                                 /setclassid adapter [classid] |
                                 /showclassid6 adapter |
                                 /setclassid6 adapter [classid] ]
```

1. ipconfig (2)

- Here is a part of information which is displayed while run the command "***ipconfig /all***"

  Tips:

  1. The Physical address has 48 bits, expressed in hexadecimal
  2. IPv4 address and Subnet Mask has 32 bits, expressed in dotted decimal

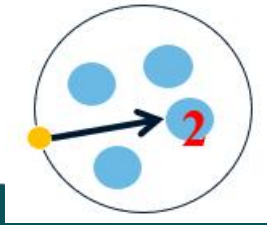


Practice 1.1

- Practise on " ***ipconfig*** " with option "***/all***" , what info will be shown by running this command?
- Are the **IP address**, **subnet mask** and **default gateway** of your PC same as those of your classmates? What are **the same**, What are **different**? Are your PCs **in the same subnet**?
- In the following pictures, PC1 and PC2 are in two differnent subnets, if PC1 needs to communicate with PC2, what's the usage of default gateway?(optional)

PC1
IP address: 192.168.**1**.104
Subnet mask: **255.255.255.0**
Default Gateway: 192.168.**1**.1

PC2
IP address: 192.168.**2**.104
Subnet mask: **255.255.255.0**
Default Gateway: 192.168.**2**.1



2. arp (1)

- "**arp**" is usually used to display or modify the address translation table (ARP cache, with the IP and MAC address pairs in it) which is used by ARP protocol.

- "**arp /?**" or "**arp -?**" can display its help information.

```
C:\Users\Administrator>arp /?

Displays and modifies the IP-to-Physical address translation tables used by
address resolution protocol (ARP).

ARP -s inet_addr eth_addr [if_addr]
ARP -d inet_addr [if_addr]
ARP -a [inet_addr] [-N if_addr] [-v]

  -a            Displays current ARP entries by interrogating the current
                protocol data.  If inet_addr is specified, the IP and Physical
                addresses for only the specified computer are displayed.  If
                more than one network interface uses ARP, entries for each ARP
                table are displayed.
  -g            Same as -a.
  -v            Displays current ARP entries in verbose mode.  All invalid
                entries and entries on the loop-back interface will be shown.
  inet_addr     Specifies an internet address.
  -N if_addr    Displays the ARP entries for the network interface specified
                by if_addr.
  -d            Deletes the host specified by inet_addr. inet_addr may be
                wildcarded with * to delete all hosts.
  -s            Adds the host and associates the Internet address inet_addr
                with the Physical address eth_addr.  The Physical address is
                given as 6 hexadecimal bytes separated by hyphens. The entry
                is permanent.
  eth_addr      Specifies a physical address.
  if_addr       If present, this specifies the Internet address of the
                interface whose address translation table should be modified.
                If not present, the first applicable interface will be used.
Example:
  > arp -s 157.55.85.212   00-aa-00-62-c6-09  .... Adds a static entry.
  > arp -a                                    .... Displays the arp table.
```

2. arp (2)

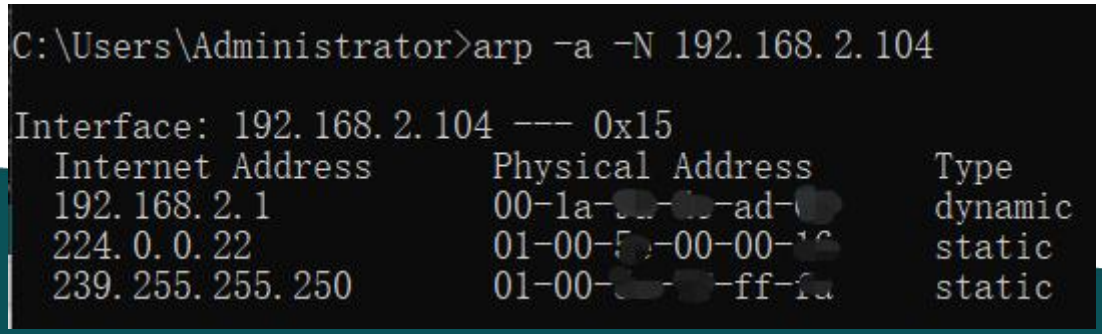
- **arp –a**
  - **Display** all ARP information, that is, the corresponding relationship between all activated IP addresses and physical addresses
- **arp –d**
  - **Delete** all ARP cache contents.
  - If the IP address is specified in the command, only the ARP cache information of the IP address is deleted.
- **arp –s**
  - **Adding** the corresponding relationship between IP address and physical address to ARP cache

Practice 1.2

- Run the "***arp - a***" command to display all the corresponding relationships in the "IP address to physical address" address translation table (ARP cache).

- You can try to solve the problem of IP address embezzlement in LAN by using "***arp -s***" command according to the format, and bundle the static IP address with the physical address of the network card. For example, "***arp -s 172.16.0.19 00-10-5C-BE-11-CC***".
-
- Run the command "***arp -s 192.168.2.222 00-11-22-33-44-YY***", could this mapping between two address be added to ARP cache? Why?

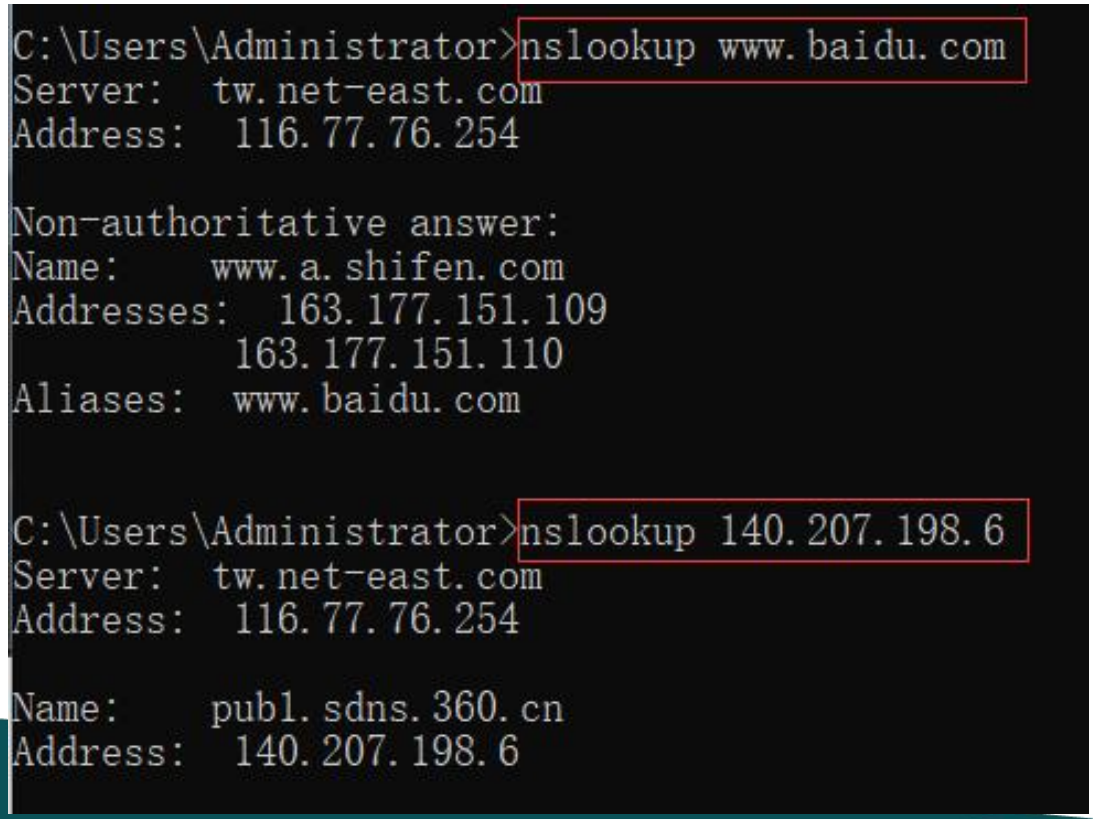
- In the following picture, "192.168.2.104" is the IP address of a wirelesscard, "192.168.2.1" is its default gateway, could this arp item related to "192.168.2.1" be deleted or changed from ARP cache?



SUSTech
Southern University of Science and Technology

3. nslookup

- "**nslookup**" is usually used to find the corresponding IP through the host name, or find the corresponding host by specifying the IP. The former involves DNS services, while the latter involves reverse DNS (rDNS) services. Generally speaking, DNS services are available, but some websites may not support rDNS services.



Part. B

Introduction to Python(1)

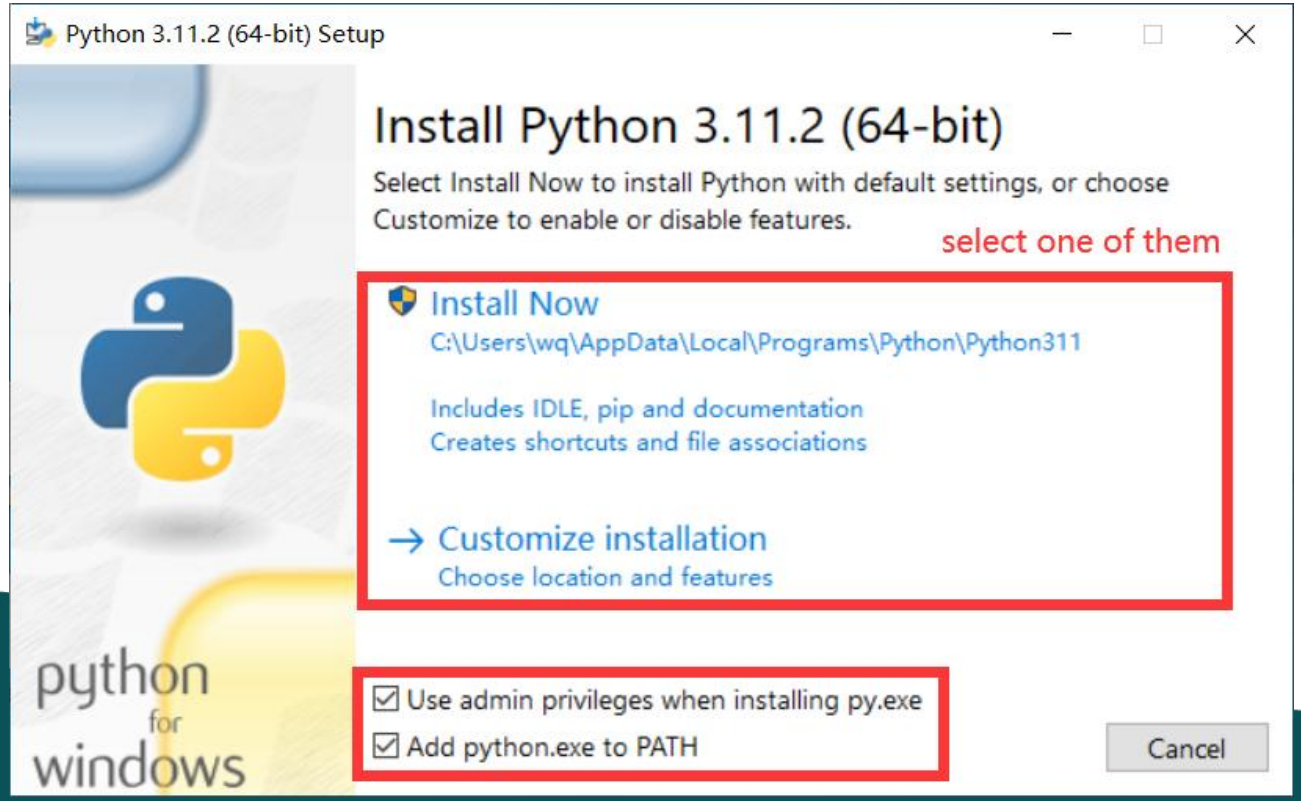
Thanks to HHQ. Zhang

Python

- Python is an interpreted high-level object-oriented programming language.
- First release in 1991.
- Official Tutorial: https://docs.python.org/3/tutorial/

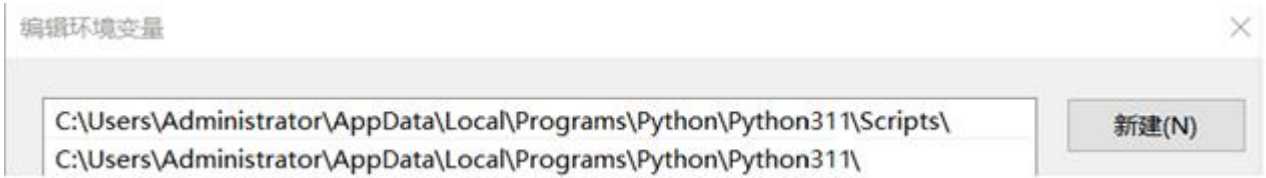
Install python(1)

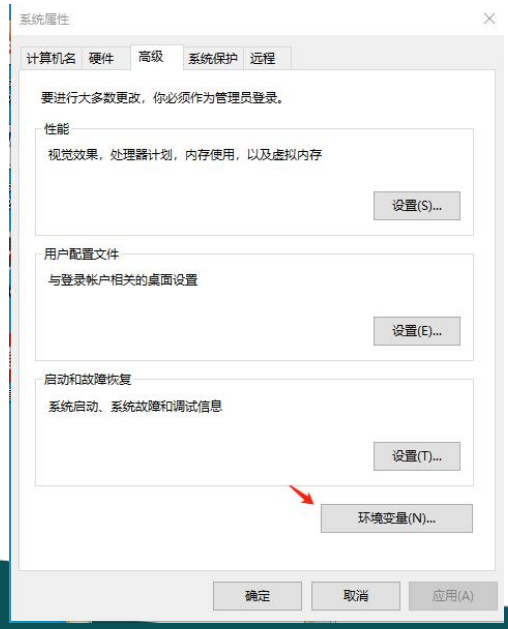
- The installation package can be got from **https://www.python.org/downloads/**
- You can choose install it by **default settings** or **customize installation**.
- It is highly recommend that choose 'Add python.exe to PATH', or you need to set PATH by hand as next page shows.



Install python(2)

- If the ‘Add python.exe to PATH’ is not set while installing, configure ‘Path’ manually according to the following steps after the installation.
  - Right click ‘my computer’ on the desktop
  - select ‘attribute’-> ‘advanced attribute’->environment variable
  - configure ‘Path’ with the path where python.exe belongs and its subdirectory ‘Scripts’





Read-Eval-Print Loop

- Python has an REPL playground.
- Type and get feedback.

```
C:\Users\Administrator>python
Python 3.8.6rc1 (tags/v3.8.6rc1:08bd63d, Sep  7 2020, 23:10:23) [MSC v.1927 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print('Hello World!')
Hello World!
>>>
```

Basic Types and Operations

- The following standard types are built in the interpreter:
  - **Numeric** Types — int, float, complex
  - **Boolean** Type — True, False
  - **Text Sequence** Type — str
  - **Sequence** Types — list, tuple, range
  - **Set** Type & **Dict** Type
  - **Binary Sequence** Types — bytes, byte array
- There are predefined operations on each type

- Ref: https://docs.python.org/3/library/stdtypes.html

Sequence Types

- **List**

  animals = **[**'dog', 'cat', 'bird'**]**

  animals[0] # => 'dog'

  animals[0] = 'puppy'

```
>>> animals = ['dog','cat','bird']
>>> animals[0]
'dog'
>>> animals[0]='puppy'
```

- **Tuple**

  animals = **(**'dog', 'cat', 'bird'**)**

  animals[0] # => 'dog'

  animals[0] = 'puppy'

  Traceback (most recent call last):

  File "<stdin>", line 1, in <module>

  TypeError: 'tuple' object does not support item assignment

```
>>> animals = ('dog','cat','bird')
>>> animals[0]
'dog'
>>> animals[0]='puppy'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item assignment
```

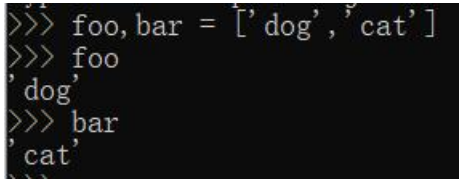
Unpacking from Sequence Types

- **List**

  foo, bar = **[**'dog', 'cat'**]**

  foo # => 'dog'

  bar # => 'cat'

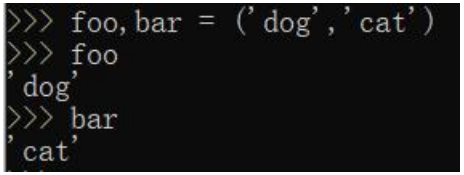


- **Tuple**

  foo, bar = **(**'dog', 'cat'**)**

  foo # => 'dog'

  bar # => 'cat'



Set & Dict

- **Set**

```
animals = set()
animals.add('dog')
animals  # => {'dog'}
```

```
>>> animals = set()
>>> animals.add('dog')
>>> animals
{'dog'}
```

- **Dict**

```
alias = dict()
alias['dog'] = 'puppy'
alias[['pig']] = ['hog']
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
TypeError: unhashable type: 'list'
```

```
>>> alias = dict()
>>> alias['dog']='putty'
>>> alias[['pig']] = ['hog']
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unhashable type: 'list'
```

Immutable & Mutable

- Mutable: it is possible to change its content
- **Immutable Type: Numeric, Boolean, str, tuple, bytes, etc.**
- **Mutable Type: list, dict, set, etc.**
- Example:

```
>>> cubes = [1, 8, 27, 65, 125]    # cubes here is a list
>>> cubes[3] = 64                  # replace the item whose index is 3
>>> cubes
[1, 8, 27, 64, 125]
```

- Only **Immutable types** can be **key of dict** or **member of set**.

Boolean Values

- Following values are treated as **False**:
  - **None**, **False**
  - **0**, **0.0**, **0j**, **Decimal(0)**, **Fraction(0, 1)**
  - **''**, **()**, **[]**, **{}**, **set()**, **range(0)**

- Otherwise they are **True**

```
>>> bool(None)
False
>>> bool(Fraction(0,2))
False
>>> bool('')
False
>>> bool(' ')
True
>>> bool(Fraction(1,2))
True
```

Flow Control — if

- Example:

```
foo = []
if foo:
    print(foo)
else:
    if foo == []:
        print('100% sure foo is empty')
    else:
        print('what hell?')
```

Flow Control — if

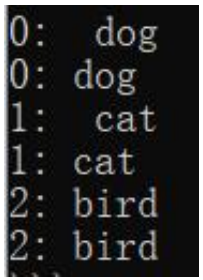
- Example:

```
foo = [1, 2, 3, 4]
if foo:
    print(foo)
else:
    if foo == []:
        print('100% sure foo is empty')
    else:
        print('what hell?')
```

Flow Control — for

- Example:

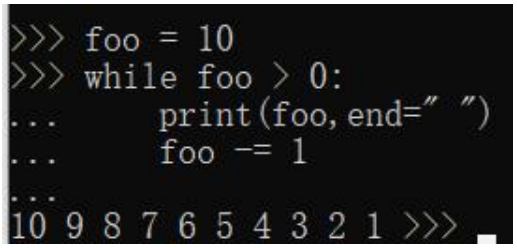
```
foo = ['dog', 'cat', 'bird']
for bar in foo:
    print(bar)
for index, value in enumerate(foo):
    print('%d: %4s' % (index, value))
    print('{0}: {1}'.format(index, value))
for i in range(10):
    print(i,end=" ")
```



Flow Control — while

- Example:

```
foo = 10
while foo > 0:
    print(foo, end=" ")
    foo -= 1
```



Defining Functions

- Example:

```
def fib(n):  # write Fibonacci series up to n
    a, b = 0, 1
    while a < n:
        print(a, end=' ')
        a, b = b, a+b
    print()
```

Defining Functions

- Example:

```
def fib2(n): # return Fibonacci series up to n
    result = []
    a, b = 0, 1
    while a < n:
        result.append(a)    # see below
        a, b = b, a+b
    return result
```

Defining Classes

```
class Animal:
    def __init__(self, name):
        self.name = name

class Duck(Animal):
    def __init__(self, name):
        super(Duck, self).__init__(name)
    def quack(self):
        print(self.name, ' Quack')
```

Module

- Save our fib functions(fib and fib2) into fibs.py

```
import fibs
fibs.fib(5) # => 0 1 1 2 3
result = fibs.fib2(5) # => [0, 1, 1, 2, 3]
```

Practise:

Add two functions fib3 and fib4 to fibs.py, the parameter of these two function is the number of first items of fibonacci sequence. fib3 print the specified number of first items of fibonacci sequence, fib4 returen a list which includes the specified number of first items of fibonacci sequence.

for example:

```
fibs.fib3(10) # => 0 1 1 2 3 5 8 13 21 34
result = fibs.fib3(10) # => [0, 1, 1, 2, 3,5,8,13,21,34]
```

Practice 2.1

- Determine whether the IPv4 address and subnet mask represented by dotted decimal are legal. If both of them are legal, calculate the network ID and host ID.
- Tips:
  - Both IPv4 address and subnet mask are 32bits, usually grouped in 8-bit units, represented in dotted decimal system.
  - The subnet mask consists of 1 and 0, and 1 and 0 are consecutive. The length of the subnet mask is also 32 bits, with network bits on the left, represented by the binary number "1". The number of 1 is equal to the length of the network bits; On the right is the host bit, represented by the binary number "0", where the number of 0s is equal to the length of the host bit.
- Requirements:
  - case1:
    - Input: 192.168.1.155 255.255.255.0
    - Output: network ID: 192.168.1.0, host ID: 155
  - case2:
    - Input: 192.168.1.355 255.255.255.0
    - Output: IP address illegal
  - case3:
    - Input: 192.168.1.155 255.253.255.0
    - Output: subnet mask illegal

Practice 2.2

- You are free tochoose one of the following two options to complete, with unlimited implementation methods (that is, you can directly execute the relevant tasks in sequence on the command line, or you can encapsulate the option task into a function or class to call and execute)

  - option1： Please create an ARP table that stores the mapping relationship between physical addresses and IP addresses. Each physical address can only have one IP address, and you can add, delete, and overwrite the table through command parameters

  - option2：Please create a mapping relationship between domain name and IP address. A domain name can correspond to multiple IP addresses, and this table supports addition and overwrite operations

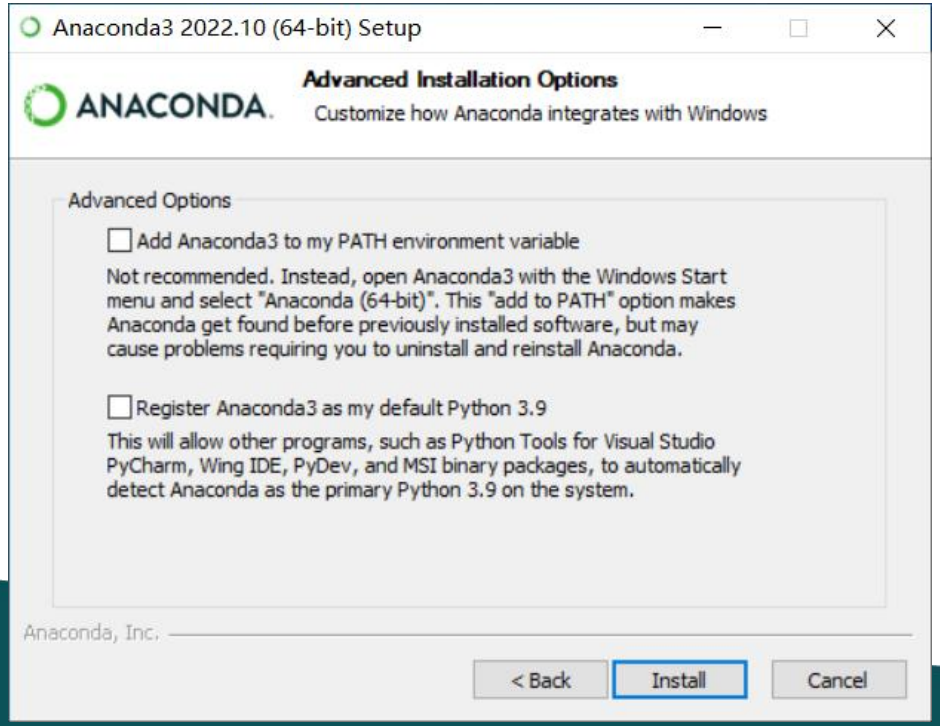
Install IDE: VS Code

- The installation package can be got from:
  - **https://code.visualstudio.com/Download/**
- Official Tutorial:
  - **https://code.visualstudio.com/learn**
- Quick Start Guide for Python in VS Code
  - **https://code.visualstudio.com/docs/python/python-quick-start**

SUSTech
Southern University of Science and Technology

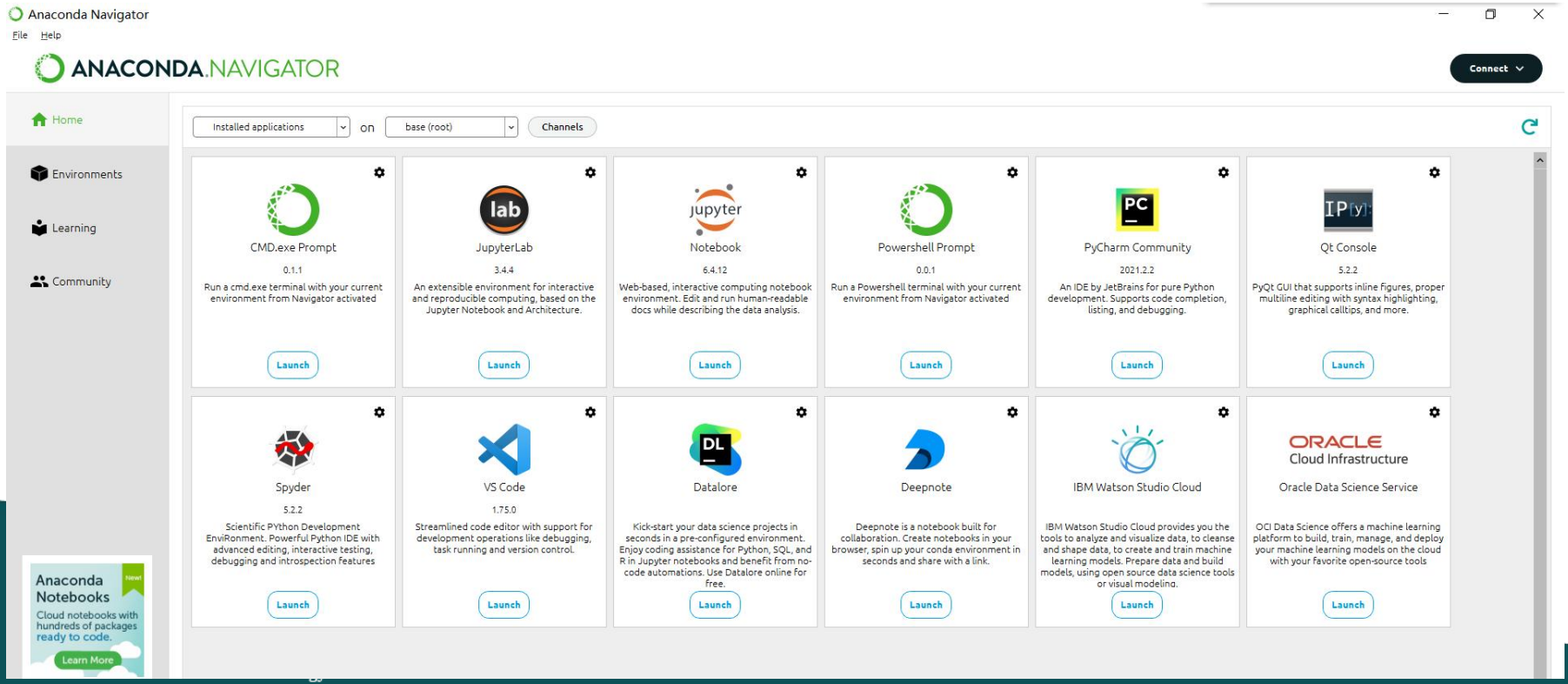
Install Anaconda

- Open source package management system and environment management system.
- Anaconda offers the easiest way to perform Python/R data science and machine learning on a single machine.
- The installation package can be got from **https://www.anaconda.com/**



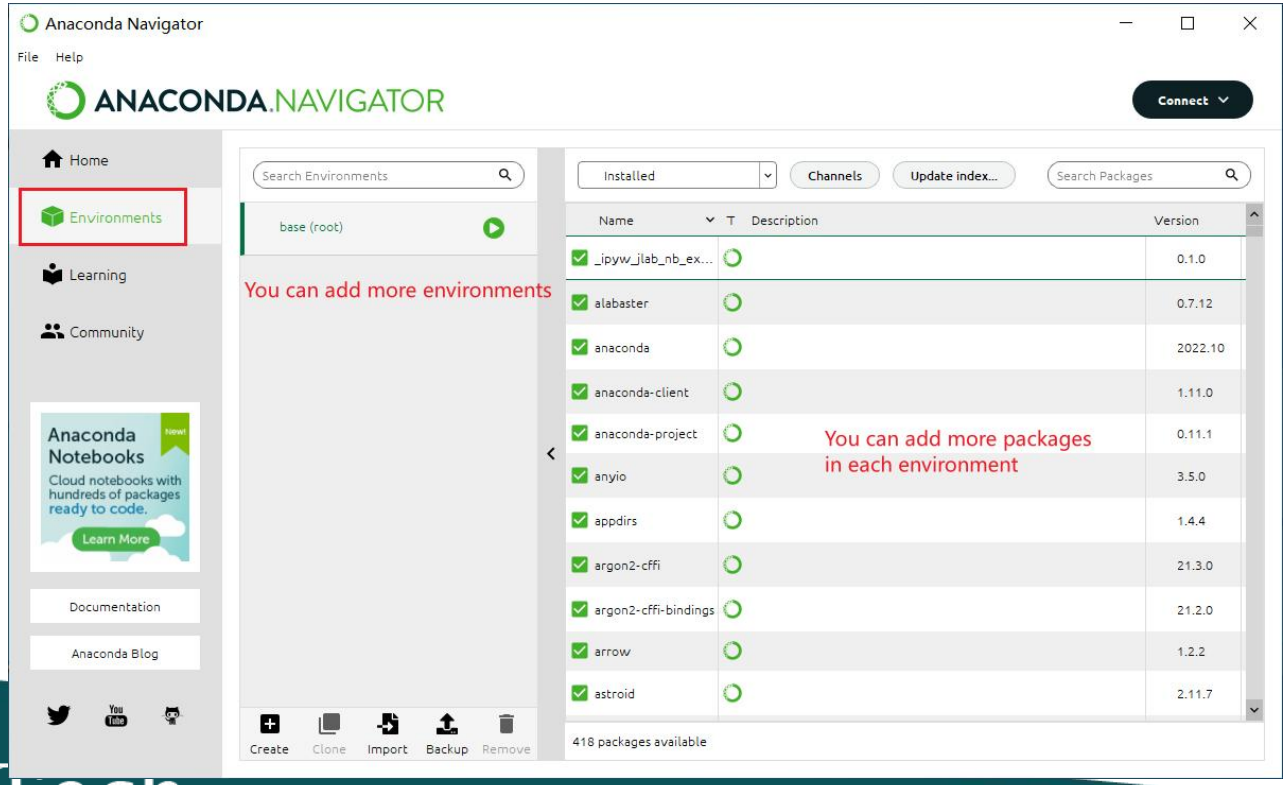
Anaconda usage

- Anaconda comes with lots of tools, including editor, interpreter, and IDE.
- Anaconda includes more than 180 scientific packages and their dependencies, including conda and Python.



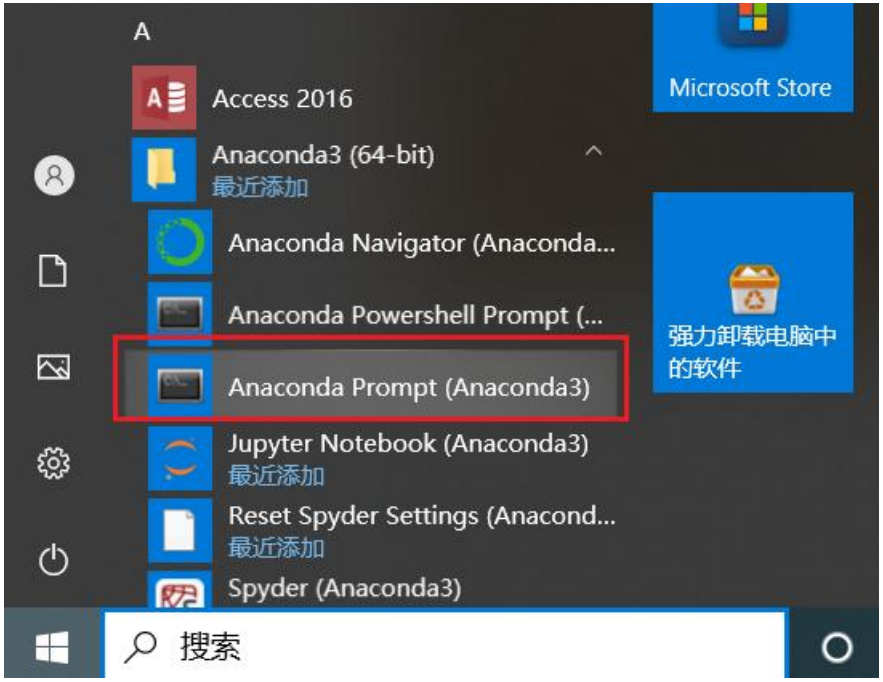
Anaconda environment

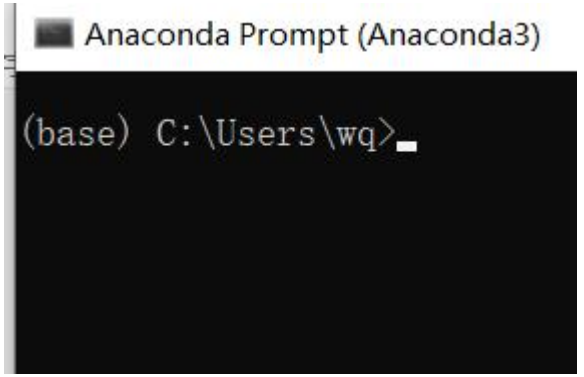
- Anaconda can create, save, load and switch environments for different projects conveniently.



Anaconda environment

- Anaconda command line.





Frequently-used conda commands

- check help information: conda –h
- check conda version: conda --version
- install package: conda install *** (for example, django)
- list all the packages: conda list
- update the package: conda update ***
- remove the package: conda remove ***
- create a new environment: conda create –n *** python = version
- activate the environment: activate ***
- quit the environment: conda deactivate ***
- delete an environment: conda remove –n *** --all
- list all environments: conda env list