Input/Output

```
# 输出变量类型

print(type(first_name))

# 变量相乘

silly = "frog" * 7

# 输入

var = input("请输入")

# Python 不能把字串直接加数字

var = var + str(123)

# 转义字符串

print("Thursday July 30 at 7.15pm: \"Inside Out\"")

# \t是tab, \n是回车, print()自己本身就有一个回车

# 字串格式化，后面可以是字串可以是 数字、字串

print("Hi {0} {1}!".format(fname, lname))
print("{1} {2} is {0} years old".format(20, fname, lname))

# 字串对齐

print("{0:<15}{1:<10}{2:^25}{3:>15}".format("Element", "Symbol", "Atomic
number", "Atomic weight)")

# /是除法，输出是精确数值的小数

10 / 2 = 5.0
10 / 4 = 2.5
10 / 2.0 = 5.0
10.0 / 1.2 = 8.3333

# //是整数除法，根据输入是否是浮点数决定输出格式，输出一定是整数输出

10 // 2 = 5
10 // 4 = 2
10 // 2.0 = 5.0
10.0 // 1.2 = 8.0

# 取余是%，找的是最后一个正整数

# 2的幂次方

2 ** 2
```

If-Else

```
# 输入选择的一个范例

print("Choose a language: (I)talian (W)elsh (Z)ulu")
language_option = input("Enter language selection: ")

if (language_option == "I"):
    print("Ciao mondo.")
    print("Come stai?")
elif (language_option == "W"):
    print("Helo Byd.")
    print("Sut wyt ti?")
elif (language_option == "Z"):
    print("Sawubona Mhlaba.")
    print("Unjani?")
else:
    print("Hello World.")
    print("How are you?")

# 范例选择条件，注意双等号

if ((age > 40) and (student_type == “Domestic”)):

# and or not

if (not (number1 == 1000)):

# 常见输入方式：

item_input = int(input("Enter the quantity: "))
```

Loop

```
# 输出 [0,10): 0-9

for i in range(0, 10):
    print(i)

# 指定输出某个数的1-10的循环乘法

number_input = input("Enter a number: ")
number = int(number_input)

for i in range(1, 10):
    print("{0} x {1} = {2}".format(i, number, number * i))

# Friend of 10 table

for i in range(0, 11):
    print("{0:>2} + {1:>2} = {2:>2}".format(i, 10 - i, 10))

# 输出 0, 1, ... 10. 这种序列
```

```python
for i in range(0, 11):
    if (i < 10):
        trailing = ", "
    else:
        trailing = "."
    print(i, end="")
    print(trailing, end="")

# 累加1-10：

result = 0

for i in range(1, 11):
    result = result + i
print("The sum of 1 to 10 is {0}".format(result))

# 输出：
# 2 1
# 4 3 2 1
# 6 5 4 3 2 1
# 8 7 6 5 4 3 2 1
# 10 9 8 7 6 5 4 3 2 1

for i in range(1, 6):
    start_number = 2 * i
    for j in range(0, start_number):
        number = start_number - j
        print(number, end=" ")
    print()

# break 跳出循环

# 字串大小写

name = "John Smith"
name_uppercase = name.upper()
print(name_uppercase)

name = "John Smith"
name_lowercase = name.lower()
print(name_lowercase)

# 查找字子串

name = "Alexandra"
index = name.find("exa")
print(index)

# 字串长度len()

greeting = "Hi there!"
greeting_length = len(greeting)

# 截取字串

sentence = "Python is cool!"
sub_sentence1 = sentence[1:4]  # "yth" 1-4不包括4
sub_sentence2 = sentence[1:]  # "ython is cool!" 1-最后包括1
```

```
sub_sentence3 = sentence[:4]  # "Pyth" 4之前不包括4

# 输出字串每个字符

greeting = "Hi there!"
for i in range(0, len(greeting)):
    letter = greeting[i]
    print(letter)

# 替换部分字符生成密码

username = "SomeString"
password = ""
for i in range(0, len(username)):
    letter = username[i]
    if (letter == "i") or (letter == "I"):
        password_letter = "1"
    elif (letter == "r") or (letter == "R"):
        password_letter = "7"
    elif (letter == "s") or (letter == "S"):
        password_letter = "5"
    elif (letter == "z") or (letter == "Z"):
        password_letter = "2"
    else:
        password_letter = letter
    password = password + password_letter

# 计算输入的奇数偶数个数

even_count = 0
odd_count = 0

while True:
    user_input = input("Enter an integer (or q to quit): ")
    if (user_input == "q"):
        break
    number = int(user_input)
    if (number % 2 == 0):
        even_count += 1
    else:
        odd_count += 1
print("You have entered {0} even numbers".format(even_count))
print("You have entered {0} odd numbers".format(odd_count))
```

Function

```
# return 返回多个值

def ask_name():
    first_name = "Finley"
    last_name = "Fish"
    return first_name, last_name


first_name, last_name = ask_name()
```

```python
# int()别忘记加了

def ask_input():
    word = input("Enter a word: ")
    user_input = input("Enter expand factor: ")
    multiplicity = int(user_input)  # int()别忘记加了
    return word, multiplicity


# 扩展字符串

def expand(word, multiplicity):
    result = ""
    for i in range(0, len(word)):
        letter = word[i]
        letter_multiply = letter * multiplicity
        result = result + letter_multiply
    return result


# 替换部分字串生成密码

def generate_password(username):
    password = ""
    for i in range(0, len(username)):
        username_letter = username[i]
        password_letter = transform_character(username_letter)  # 这边用到另一个函数
        password = password + password_letter
    return password


def transform_character(letter):
    if (letter == "i") or (letter == "I"):
        password_letter = "1"
    elif (letter == "r") or (letter == "R"):
        password_letter = "7"
    elif (letter == "s") or (letter == "S"):
        password_letter = "5"
    elif (letter == "z") or (letter == "Z"):
        password_letter = "2"
    else:
        password_letter = letter
    return password_letter


# 设置默认参数

def welcome(name, greeting="Hi"):  # 这边设置
    print("{0} {1}!".format(greeting, name))


welcome("John", "Hello")
welcome("Mary", greeting="It is nice to meet you")  # 这边指定参数名来做
welcome("Paul")


# Recursive functions: 递归函数
```

```python
# factorial(n) = n x factorial(n-1)
# 输出1-9的阶乘

def factorial(n):
    if (n == 1):
        return 1
    else:
        return n * factorial(n - 1)


for i in range(1, 10):
    print("{0}! = {1}".format(i, factorial(i)))

# 四舍五入函数：round

number = 28.30188679245283
rounded_number = round(number)  # 保留整数，四舍五入
rounded_number = round(number, 1)  # 保留1位小数，四舍五入
rounded_number = round(number, 2)

# max(), min() 函数既可以接受 几个直接放进来的元素，也可以接受 列表和集合

min_num = min(num1, num2, num3)

# 输出10个 1-6 之间的随机整数

import random

for i in range(0, 10):
    random_number = random.randint(1, 6)  # 这里是包含1和6的
    print("Dice result: {0}".format(random_number))
```

Class

```python
# Instance attribute: data belongs to individual object instance.
# Class attribute: data that is common to all objects.
# (Some classes do not have any class attributes.)

class Student:
    # 类属性
    email_domain = "solla.sollew.edu"
    student_dir = "/user/student"

    def __init__(self, id, first_name, last_name):
        self.id = id
        self.first_name = first_name
        self.last_name = last_name


# Instance method: Deal with a particular individual object instance
# Static / Class method:
# 1. Do NOT deal with individual object instance
# 2. Common to all object instances

# instance method can be invoked from an object
# Use class name to call static method:
```

```
staff2.update_employment_type("Casual")
length3 = tv_program2.get_length_in_minutes()
minute_count5 = tv_program5.time_left_in_minutes(now)


# special/dunder methods have the double underscores in the method name

class TV_Program:

    def __init__(self, channel, title, start_time):

    def __str__(self):

    def __repr__(self):


# Static / Class method:
# 1. Do NOT deal with an individual object instance
# 2. Common to all object instances
# static class method can be invoked from class name

# Use the decorator @staticmethod to define a static method:

contact_email = Student.admin_email()


# The first argument (cls) of a class method is always referred to the class

class Student:
    email_domain = "solla.sollew.edu"
    student_dir = "/user/student"

    @classmethod
    def admin_email(cls):  # 如果要用到cls里面的类属性，否则不用加
        return "admin@" + cls.email_domain

    @staticmethod
    def uni_website():
        return "http://www.solla.sollew.edu"


# 调用静态函数、类函数：用类名

# 静态方法是从地址上找东西，所以这里是呼唤Student类
print("Uni website: " + Student.uni_website())
print("Admin email: " + Student.admin_email())

# 静态方法 改 类方法
# staticmethod 改 classmethod
# () 改 (cls)

# 点成员、点函数

student1 = Student("0973427", "John", "Smith")
print(shark.name)
angelfish.address = "Uni duck pond"
```

```python
# 注释函数意思

class Student:
    """
    Class Student represents a student
    """

    def fullname(self):
        """
        Get student's full name
        """
        return self.first_name + " " + self.last_name


# 使用 help(Student) 获取帮助

print(help(Student))


# 定义 学生学号

class Student:
    def __init__(self, id, first_name, last_name):
        self.id = id
        self.first_name = first_name
        self.last_name = last_name
        # 生成学号
        self.username = first_name[0].lower() + last_name[0].lower() + id[0:3]

    def email_alias(self):
        """
        Get student's friendly-looking email:
        firstname.lastname.3IDdigits@domain
        """
        return self.first_name + "." + self.last_name + "." + self.id[0:3] + "@"
+ Student.email_domain

    def home_dir(self):
        """
        Get student's Unix home directory:
        studentDir/username
        """
        return Student.student_dir + "/" + self.username

    def fullname(self):
        """
        Get student's full name
        """
        return self.first_name + " " + self.last_name

    def email(self):
        """
        Get student's email: username@domain
        """
        return self.username + "@" + Student.email_domain  # 个人方法使用类属性
```

```
student2 = Student("1882845", "Mary", "Wilson")
print(student2.email())
print(student2.email_alias())


# 应用str()到类实例

class Student:
    def __str__(self):
        return "{0} ({1})".format(self.fullname(), self.id)


student2 = Student("1882845", "Mary", "Wilson")
print("Object student2 is " + str(student2))


# 应用repr()到实例

class Student:
    def __repr__(self):
        return "Student('{0}', '{1}', '{2}')". \
            format(self.id, self.first_name, self.last_name)


student2 = Student("1882845", "Mary", "Wilson")
print(repr(student2))


# 多类继承

# Python supports multiple class inheritance:
# a child class can inherit from multiple parent classes.
# Class inheritance allow child class:
# 1. To inherit all parent attributes and methods;
# 2. To override parent attributes;
# 3. To override parent methods.

# 继承，可以重写 类属性 和 类方法

class PostGradStudent(Student):
    # 重载类属性
    student_dir = "/user/poststudent"
    web_domain = "www.solla.sollew.edu"

    def __init__(self, id, first_name, last_name, thesis):  # 原来的+新的
        # 这句话很关键
        super().__init__(id, first_name, last_name)
        # 新的属性
        self.thesis = thesis

    # 增加新函数
    def web_address(self):
        # 这边使用 类属性
        return PostGradStudent.web_domain + "/" + self.username


# 重写分继承和不继承，不继承不用加self，继承要加self和使用super()
```

```
# super()是调用上一级的函数

class PostGradStudent(Student):

    def print_detail(self):
        super().print_detail()  # 这边要加东西
        print("Thesis: " + self.thesis)
        print("Web address: " + self.web_address())
```

List

```
# 两种遍历列表方式

for i in range(0, len(animal_list)):
    print(animal_list[i])

for animal in animal_list:
    print(animal)

# .append()尾加到列表

animal_list.append("kangaroo")
animal_list.append("emu")

# .insert()插入

animal_list.insert(1, "emu")  # 1号位，0开始

# del 删除某元素

del subject_list[1]

# .remove()按值删除第一个出现的数，没有的话会报错

random_numbers = [3, 12, 4, 5, 4, 3, 2, 6, 12]
random_numbers.remove(4)

# .index() 按值找最先出现的index

four_index = random_numbers.index(4)  # 找第一个数值4的出现位置

# .count() 计算列表中数值的出现次数

four_count = random_numbers.count(4)

# sorted() 返回一个排序后的新列表，原列表不变

sorted_numbers = sorted(random_numbers)

# .sort() 是列表内部自己的函数，对自己做了修改排序

random_numbers.sort()

# .reverse() 反转列表

random_numbers.reverse()
```

```
# .clear() 清空列表所有元素

random_numbers.clear()

# 列表可以直接相加

list12 = list1 + list2

# 列表也可以乘

list3 = [9, 8]
list4 = list3 * 3

# 截取列表/列表切片注意 这里不包含4

list1 = random_numbers[1:4]

# 列表可换元素，元组不可换元素

animal_tuple = ("dog", "cat", "frog")
animal_tuple[0] = "elephant"  # ERROR

# .title() 是字串首字母大写

# 元组不可变，只能全体重置，d="12"字串也是不可局部调出来变

d = (1, 2)
d[0] = 3  # error
d = (3, 2)  # OK

# 在 b = a 中 b与a 不是同一个Id
# 但是列表 等同是 引用同一个id的
# 即使事先声名，也一样应用同一个

# 使用.copy()可以新建一个列表

a = [1, 2, 3]
b = []
b = a
b[1] = 4
print(a)  # [1,4,3]

c = a.copy()
c[1] = 2
print(a) # a不受影响
```

Dict

```
# 注意字典左右两边都要加引号，右边除了数值

empty = {}
person = {
    "first_name": "Amanda",
    "last_name": "Smith",
    "age": 20
```

```
}

# 根据键获取值

first_name = person.get("first_name")

# 如果不存在会返回 None

email = person.get("email")
if (email is None):  # None用is做判断
    print("User has no email")
else:
    print("User email is " + email)

# 为获取不到的情况设置默认值

std_type = person.get("student_type", "N/A")
credit_point = person.get("credit_point", 0)

# 获取字典值或者更改值

person["first_name"] = "Mandy"
person["last_name"] = "Jones"
person["age"] = 24

# 删除键对值

del person["email"]

# 清除字典

person.clear()

# 使用范例

country = input("Enter country: ")
capital = capital_city.get(country)
if capital is None:
    print("Sorry I don't know the capital city of " + country)
else:
    print("Capital city of {0} is {1}".format(country, capital))

# 遍历键、值、键值对

spam = {
    "first_name": "Amanda",
    "last_name": "Smith",
    "age": 20
}

for v in spam.values():
    print(v)

for k in spam.keys():
    print(k)

for i in spam.items():
    print(i)
```

```
for k, v in spam.iems():
    print(k + '=' + str(v))

# 集合是可以自动去重的

a = {'a', 'b', 'a', 'c'}
b = set(a_list)

# 列表套字典

p0 = {'a', 'b', 'c'}
p1 = {'a', 'b', 'c'}
p2 = {'a', 'b', 'c'}
ls = [p0, p1, p2]
a = len(ls)  # 3
for p in ls:
    print(p)

# 字典套列表

p = {
    'name': 'some',
    'age': '18',
    'like': [
        'abc',
        'bcd'
    ]
}

print(f"{p['name']} like:")
for i in p['like']:
    print(i)
```

File

```
# 文件mode有：r、w(已经有的会被覆盖)、a(追加)、r+(读和写)

silly_file_path = "put/the/file/path/here/silly.txt"

# 写
with open(silly_file_path, "w") as silly_file:
    # f.write()函数
    silly_file.write("Hi! ")
    silly_file.write("I am Sam.\n")
    silly_file.write("Would you like green egg and ham?\n")

user_input = input("Enter a number to generate times table: ")
number = int(user_input)
file_path = input("Enter output file path: ")
with open(file_path, "w") as timestable_file:
    for i in range(1, 10):
        timestable_file.write("{0} x {1} = {2}\n".format(number, i, number * i))

# 读
with open(text_file_path) as silly_file:
```

```
    while True:
        line = silly_file.readline()
        if (line == ""):
            break
        print(line)

with open(text_file_path) as silly_file:
    for line in silly_file:
        print(line)


# CSV 格式

# stn,first_name,last_name
# 1111,John,Smith
# 2222,Lee,May
# 3333,Ye,Zhang

import csv

student_file_path = " put/the/file/path/here/student .csv"

with open(student_file_path, "w") as student_file:

    field_name_list = ["stn", "first_name", "last_name"]
    # 写dict的写法，先定义一个writer = csv.DictWriter(文件名，列名字段名一个列表)
    # x = csv.DictWriter(f_name,field_name)
    writer = csv.DictWriter(student_file, fieldnames=field_name_list)

    # 启动：先写入header
    writer.writeheader()
    # 再以 writerow 写入字典
    writer.writerow({"stn": "1111", "first_name": "John", "last_name": "Smith"})
    writer.writerow({"stn": "2222", "first_name": "Lee", "last_name": "May"})
    writer.writerow({"stn": "3333", "first_name": "Ye", "last_name": "Zhang"})

# 获取的是每个字典，读也是按照每个字典来读的

import csv

student_file_path = " put/the/file/path/here/student.csv"

with open(student_file_path) as student_file:

    # 先是读取文件，这边是直接读取了整个文件
    x = csv.DictReader(f)
    reader = csv.DictReader(student_file)

    for row in reader: # 用get的方式
        student_number = row.get("stn")
        fname = row.get("first_name")
        lname = row.get("last_name")
        print("{0:<10}{1:<10}{2:<10}".format(student_number, fname, lname))


# JSON 格式

# json.dump() 导入列表
```

```python
# json.dump(x,f) 里面两个位置不要搞错!

import json

numbers = [2, 3, 5, 7, 11, 13]
filename = 'numbers.json'
with open(filename, 'w') as f:
    json.dump(numbers, f)

# json.load() 读取列表

with open(filename) as f:
    numbers = json.load(f)

print(numbers)
```

Exception

```python
# 基本格式

try:

except ExceptionA as e:

except ExceptionB as e:

except ExceptionC as e:

except:  # 接受任意错误都执行

else:  # 前面所有意外都不发生才执行

finally:  # finally 表示一定会执行

# int()一个str：ValueError
# 除0错误：ZeroDivisionError
# 找不到模块错误：ModuleNotFoundError
# 未命名错误：NameError
# 列表取元素超范围错误：IndexError

# 类名不应以Error结尾

# BaseException
# Exception
# ValueError NameError ArithmeticError(ZeroDivisionError)


# Raise Error

try:
    user_input = input("Enter an integer: ")
    number = int(user_input)
    print("You have entered {0}".format(number))
except ValueError as e:
    print(e)  # e是原来错误信息输出
    print("You have entered an invalid number")  # 输出自定义错误信息
```

```python
# 范例是 ValueError 和 ZeroDivisionError

try:
    user_input = input("Enter the 1st integer: ")
    number1 = int(user_input)
    number2 = int(user_input)
    quotient = number1 / number2
    print("{0} / {1} = {2}".format(number1, number2, quotient))
except ValueError as e:
    print("You have entered an invalid number")
except ZeroDivisionError as e:
    print("Invalid division - cannot divide by 0")

# try-except 的嵌套

try:
    user_input = input("Enter a positive integer: ")
    try:
        number = int(user_input)
    except:
        raise ValueError("Invalid integer format ")  # 自定义错误信息
    if (number <= 0):
        raise ValueError("Input must be a positive number ")  # if 也可以抛错
    print("You have entered {0}".format(number))
except ValueError as e:  # 这边用来接收，并指定输出格式
    print("Error: " + str(e))

# 范例：不断要求输入直到正确输入正整数

while True:
    try:
        user_input = input("Enter a positive integer: ")
        try:
            number = int(user_input)
        except:
            raise ValueError(" Invalid integer format ")
        if (number <= 0):
            raise ValueError(" Input must be a positive number ")
        print("You have entered {0}".format(number))
        break
    except ValueError as e:
        print("Error: " + str(e))


# 自定义错误类

class BadInputError(Exception):
    def __init__(self, message):
        self.message = message


try:
    print("1) Green eggs and ham")
    print("2) Red breads with jam")
    print("3) Blue salad with lamb chops")
    food_option = input("Enter your selection (1/2/3): ")
    if food_option == "1":
```

```
        food = "green eggs and ham"
    elif food_option == "2":
        food = "red breads with jam"
    elif food_option == "3":
        food = "blue salad with lamb chops"
    else:
        raise BadInputError("You have to choose 1, 2 or 3.")

    print("Drink size:")
    print("S) Small")
    print("M) Medium")
    print("L) Large")
    drink_option = input("Enter your selection (S/M/L): ")
    if drink_option == "S":
        drink = "small drink"
    elif drink_option == "M":
        drink = "medium drink"
    elif drink_option == "L":
        drink = "large drink"
    else:
        raise BadInputError("You have to choose S, M or L.")

except BadInputError as e:
    print(e.message)


# 追朔式报错

def spam():
    bacon()


def bacon():
    raise Exception("This is error info")  # Exception()这边可以直接塞错误信息


spam()

# 把错误写入文件

import traceback

try:
    raise Exception('This is the error message')
except:
    errorFile = open('error_info.txt', 'w')
    errorFile.write(traceback.format_exc())
    errorFile.close()
    print("错误已写入文件")

# 断言：Assertion

ages = [28, 32, 97, 72]
ages.sort()
# 这边直接认定（判断）这个条件，条件符合继续，条件不符合报错右边信息
assert ages[0] < ages[-1], "The first should <= the last"

# logging 控制台输出信息
```

```python
import logging

# 设置logging 输出格式
logging.basicConfig(level=logging.DEBUG, format='%(asctime)s - %(levelname)s - %(message)s')

# 开始运作
logging.debug('Start Debug')


def factorial(n):
    logging.debug("Start of fac(%s)" % n)
    total = 1
    for i in range(1, n + 1):
        total *= i
        logging.debug(f'i is {i}, total is {total}')
    logging.debug('End of fac(%s)' % n)
    return total


print(factorial(5))
logging.debug("End")
```