1. **Python语言和开发环境**
2. Python语言介绍(了解)

Python(蟒蛇)是一种,解释型,面向对象,动态数据类型的高级程序设计语言,由吉多-范罗苏姆,1989年底发明的,第一个发行版,在1991年.

Python Java C Mac DB

Python语言特点:

* 免费自由的软件
* 良好的平台性(可跨平台)
* 强大功能和可扩展性
* .....

Python应用领域:

* Google搜索引擎核心代码
* 网易游戏服务器
* 豆瓣网主体架构
* 无人驾驶(百度、小马、地平线...)
* ....

测试为什么学习Python:

* 了解开发原理和架构(LAMP),更好的发现缺陷
* 可以更好的和开发沟通
* 为自动化测试(功能、接口)做铺垫
* 面试

1. 开发环境搭建

(Python+PyCharm+DB+Linux+Gitee+Django+HTML)

(1)Python和PyCharm安装和配置

下载地址:

https://www.python.org/downloads/windows/

------------------------------------

https://www.jetbrains.com/pycharm/download/#section=windows

------------------------------------

安装和配置:

* 环境变量 --- Python
* 主题/字体/汉化 --- PyCharm

位置:文件->设置-> 外观和行为

编辑器

插件

(2)Python和PyCharm关系

Python --- 核心

PyCharm --- 外表

1. 开发项目思路
2. 项目名称 --- TSD2205
3. 编写代码 --- xxx.py
4. 执行代码
5. **数据类型**
6. Python数据类型(了解)

整数类型(int)

小数(浮点数)类型(float)

布尔类型(bool)

字符串(str) 列表(list)

元组(tuple) 集合(set)

字典(dict)

1. 变量

变化的量,存储在内存中,Python中的变量是动态数据类型的.

语法格式: 变量名 = 变量值

例如: a = 100 b = False

说明: =表示赋值符号, == 表示等号

语法格式表达的意思:将右边的变量值赋值给左边的变量名

1. 整数类型(int)

整数类型包括:正整数、负整数、零

123、243124、-999、0

例如:

x = 999 # 普通赋值

x = y = z = -999 # 链式赋值

x, y, z =100, -798, 0 # 多个赋值操作同时进行

* 扩展知识 --> 其他进制整数
* 10进制整数

例如: a = 10000

* 2进制整数(0B)

例如: a = 0B10010

* 8进制整数(0O)
* 16进制整数(0X)

例如: a = 0X520ABC

1. 注释

注释的作用:更容易阅读程序和后期维护

(1)单行注释

单行注释以#号开头,快捷键为:

Ctrl+？/

(2)多行注释

多行注释以3个单引号或者3个双引号开头和结尾

5、浮点数(小数)类型(float)

* 普通表示法

x = 3.14

* 科学计数法(E/e)

y = 2.34E3 相当于 2.34 \* 103

z = 4.56e-5 相当于 4.56 \* 10 -5

* 特殊情况
* 整数部分为零,可以省略

t = .98 相当于 t = 0.98

* 小数部分为零,可以省略

r = 19. 相当于 r = 19.0

6、布尔类型(bool)

布尔类型一般表示逻辑值,包括:True(真)和False(假)

例如: flag = False

1. **标识符(规则)**

标识符用来表示,变量名、函数、类、包、...等名称

标识符命名规则:

* 标识符第一个字符,必须是字母或下划线,其余字符可以是数字、字母、下划线
* 标识符不能包含空格以及下划线以外的字符
* 不能Python保留关键字
* 严格区分大小写
* 一般避免使用,以双下划线开头和结尾的名称

------------------------------------

变量名尽量达到见名知意,可以用中文作为变量名

1. **字符串--初级**

字符串是一个有序的数据集合,主要作用:存储和表示文本,可以使用单引号、双引号、3个单引号、3个双引号,括起来的字符,表示字符串

1. **输入和输出函数**
2. 输出函数

print() 表示输出函数,具体表示:输出处理结果并换行

stu\_id = 1001  
stu\_name = 'rose'  
stu\_age = 18  
stu\_address = "北京市朝阳区南沙滩66号院1-2-3达内教育"  
print(stu\_id)  
print(stu\_name, stu\_age, stu\_address)

1. 输入函数

input() 表示输入函数,具体表示:获取键盘输入的数据,返回的是字符串

3、判断类型函数

type() 表示判断类型函数,具体表示:判断某个变量名是什么类型

例如:

age = 18

num = 3.14

---------------------------------------

sid = '10010' *# --- str*num = 10010 *# --- int*

print(sid, type(sid))  
print(num, type(num))

1. **运算符和表达式**
2. 算数运算符

+、-、\*、/、\*\*(乘方)、//(整除)、%(取余)

1. 关系运算符

>、<、==(等于)、>=、<=、!=(不等于)

说明:通过关系运算符比较得到结果是True或者False,一般应用于条件的编写

1. 逻辑运算符

and(并且)、or(或者)、not(取反)

a = -30

print(a > 0 and 1/2) # False 短路重点

print(a < 0 or 1/2) # True 短路重点

1. 复合赋值运算符(了解)

+=、-=、\*=、/=、\*\*=、//=、%=

a = 10

a += 20 # a = a + 20

print(a)

**七、字符串-高级**

1、复习定义(创建、声明)字符串

2、字符串运算

(1)支持+运算 + 表示连接符

str00 = 'Hello'  
str09 = 'World'  
str11 = str00 + str09  
print(str11)

(2)支持\*运算

s1 = 'Hello'  
s2 = s1 \* 3  
print(s2)

说明:在字符串世界里,+是连接,\*是复制

3、转义字符

转义字符就是,在字符串中某些特定符号前,加一个斜杠(\),该字符将被解释成另外一种意思。

* \n 表示换行 \*\*\*
* \t 表示制表符
* \\ 输出\字符 \*\*\* 路径
* \' 输出单引号字符串中的单引号
* \" 输出双引号字符串中的双引号

4、格式化字符

格式化字符就是:在字符串中以%开头的字符,可以在程序运行(代码执行)的时候,可以改变字符串中的内容

* %d 表示10进制整数 \*\*\*
* %s 表示字符串 \*\*\*
* %f 表示浮点数 \*\*\*
* %c 表示单个字符
* %e 表示科学计数法形式
* %% 表示%

-------------------------------------------

还可以使用f实现字符串格式化,一般格式为f '{}', {}中添加的是变量名或表达式

x = int(input('请输入第1个数:'))  
y = int(input('请输入第2个数:'))

print(f'{x}+{y}={z}') print(f'{x}\*{y}={x\*y}')

1. 字符串切片

字符串切片就是,通过字符串的下标(位置),获取字符串中的每个字符

* 语法格式1: 变量名[i]

说明:获取字符串(str)中,第i号位置的字符,字符串中位置(下标)从0开始计算,还可以是负数下标,从-1开始

str01 = 'ABCDEFGHJK'  
s1 = str01[5]  
print(s1) *# F*print(str01[2]) *# C*print(str01[7]) *# H*print(str01[-3]) *# H*

* 语法格式2: 变量名[i:k]

说明:获取字符串(str)中,第i号位置,到k-1号位置的字符,i和k都可以省略

print(str01[1:5]) *# BCDE*print(str01[2:9]) *# CDEFGHJ*print(str01[:5]) *# ABCDE*print(str01[3:]) *# DEFGHJK*print(str01[:]) *# ABCDEFGHJK*

1. 字符串解包和比较

str02 = 'HELLO'  
print(str02) *# H-E-L-L-O*s5, s6, s7, s8, s9 = str02  
print(s5, s6, s7, s8, s9, sep='-')

-------------------------------------------

str04 = 'rose'  
str05 = 'rose'  
str06 = 'Rose'  
print(str04 == str05) *# True*print(str05 == str06) *# False*# 预期结果和实际结果是否相等 判断是否是缺陷

1. 字符串常见函数

(1)len() 获取字符串长度

name = input('请输入您的用户名:')  
str\_name\_len = len(name)  
print('用户名长度是:%d' % str\_name\_len)

(2)index()

查找一个字符串,在另外一个字符串中,首次出现的位置(下标)

str01 = 'abcdfgabcdfgabcdfg'  
str02 = 'fg'  
str\_index01 = str01.index(str02)  
print('位置是:', str\_index01)

(3)count()

一个字符串在另外一个字符串中,出现的次数

str01 = 'abcdabcdabcd'  
str02 = 'd'  
count = str01.count(str02)  
print('字符串出现的次数:', count)

(4)split()

指定分隔符,从字符串左边开始分隔,分成多个子字符串,以列表的形式显示

str03 = '西瓜,草莓,蓝莓,葡萄,桃子'  
s1 = str03.split(',')  
print(s1)  
s2 = str03.split('-')  
print(s2)  
s3 = str03.split(',', 2)  
print(s3)

(5)lower()、upper()

str05 = 'abcdABCD'  
s1 = str05.lower() # lower 表示小写  
print(s1)  
s2 = str05.upper() # upper 表示大写  
print(s2)

(6)strip()

删除字符串2端的,空白字符(空格)

str01 = ' ABC ABC 123 '  
print('原字符串:', str01)  
s1 = str01.strip()  
print('删除后字符串:', s1)

(7)isalpha()、isdigit()

判断字符串是否全部都是字母、数字

str02 = 'ABCDEFG,123'  
res1 = str02.isalpha()

# isalpha() 表示字母print(res1)  
res2 = str02.isdigit()

# isdigit() 表示数字print(res2)

(8)in

检查一个值,是否包含在某个字符串(列表)中

str03 = '大鱼,小鱼,鲨鱼,鲸鱼,草鱼,巴沙鱼'  
str04 = input('请输入您要买的鱼:')  
print(str04 in str03)

(9)eval()

用来执行字符串的表达式,并返回表达式的值

x = 100  
z = eval('x \* 5')

# 'x \* 5' --> x \* 5  
print(z)

**八、高级(序列)数据类型**

8.1、列表(list)

列表是Python中使用最频繁的数据类型,列表是有序、可变的,列表中的元素,可以是不同的数据类型。

(1)定义列表

语法格式:

列表名 = [数据1,数据2,数据3,...数据n]

列表名 = list(可迭代对象)

列表名 = [ [数据1,数据2,...],

[数据1,数据2,... ],

[数据1,数据2,... ],...

[数据1,数据2,...]]

说明:列表是以[]括起来的数据集合,以逗号分隔数据,数据可以是任何数据类型。

list01 = [123, 345, 567, 890]

list03 = list()

list04 = [  
 [101, 'rose', 20, 6000, '北京'],  
 [102, 'jack', 25, 3456.78, '美国']]

(2)使用列表

* 通过列表名访问

print(列表名)

* 通过列表名+下标访问

print(列表名[下标])

print(列表名[下标:下标])

* 解包

cases = [  
 ['case\_01', 'admin', 'abc123', '登录成功'],  
 ['case\_02', 'admin', '123', '密码不正确'],  
 ['case\_03', '', 'abc123', '用户名不能为空']  
]  
case\_01, case\_02, case\_03 = cases  
print('第1条用例:', case\_01)  
print('第1条用例部分数据:', case\_01[3])

(3)特殊情况列表

* 空列表

list05 = []

list06 = list()

* 无名列表

list()

(4)删除列表

* 根据列表名删除

语法格式: del 列表名

list07 = [100, 200, 300, 'ABC']  
print('删除前数据:', list07)  
del list07  
print('删除后数据:', list07)

* 根据列表名+下标删除

语法格式: del 列表名[下标]

list08 = ['ABC', 100, 200, 'QQ']  
del list08[3]  
print('删除后数据:', list08)

(5)列表运算

* 支持+运算

list01 = [100, 200, 300]  
list02 = ['A', 'B', 'C']

res1 = list01 + list02  
print(res1)

* 支持\*运算

list01 = [100, 200, 300]

res2 = list01 \* 3  
print(res2)

(6)类型转换

* 字符串转列表

str01 = 'AB,CD,EF'  
list03 = str01.split(',')  
print(list03)  
print(type(str01), type(list03))

* 列表转字符串

list04 = ['ABC', 'DEF', 'ASD']  
str02 = str(list04)  
print(str02, type(str02))  
print('---------------------------')  
str03 = ''.join(list04)  
print(str03, type(str03))  
str04 = '-'.join(list04)  
print(str04, type(str04))

说明:使用str()方式转换成列表,会带有[]和逗号;join()方式转换成列表,列表中的元素必须都是字符串类型。

(7)列表的常见函数

* append()

向列表的尾部追加一个元素

list01 = [100, 200, 300]  
print('list01添加前数据:', list01)  
list01.append(400)  
print('list01添加后数据:', list01)

* insert()

向列表中,指定位置,插入数据

list02 = [100, 200, 300]  
list02.insert(1, 'ABC')  
print('list02添加后数据:', list02)

* len()

统计列表中,元素的个数

list03 = [100, 200, 300, 'ABC']  
len = len(list03)  
print('list03列表中元素个数:', len)

* pop()

删除指定位置上的元素

list04 = [101, 'mary', '女', 90.5]  
list04.pop(3)  
print('删除后列表元素:', list04)

* clear()

删除列表中的全部元素

list05 = [101, 'mary', '女', 90.5]  
print('原列表数据:', list05)  
list05.clear()  
print('删除后列表数据:', list05)

* count()

统计列表中,某个元素出现的次数

list06 = [100, 200, 100, 300, 100]  
count = list06.count(100)  
print('元素出现的次数:', count)

* index() \*\*\*

计算某个元素,在列表中,首次出现的位置

list07 = [100, 'A', 200, 'A', 300]  
index = list07.index('A')  
print('元素出现的位置是:', index)

8.2、元组(tuple)

元组是有序、不可变的数据类型,一旦定义,不可以修改,称为常量列表,访问和处理数据的速度快、安全,一般由于数据库方向。

(1)定义元组

元组名 = (数据1,数据2,...,数据n)

元组名 = tuple(可迭代对象)

说明:元组是以()括起来的数据集合,以逗号分隔,元组中可以包含任何数据类型。

tuple01 = (100, 200, 300)  
tuple02 = (101, 'rose', 60.5, True, ['语文', '数学', '英语'])  
tuple03 = tuple()

(2)使用元组

* 通过元组名访问元组

print(tuple01)  
print(tuple02)

* 通过元组名+下标访问元组

print(tuple01[1])  
print(tuple02[4])

print(tuple02[1:3])

* 解包

tuple04 = ('A', 100, 'C')  
t1, t2, t3 = tuple04  
print(t2)

-------------------------------------------

tuple05 = ('username','admin')  
sql = ("select \* from emp where

%s=%s" % tuple05)  
print(sql)

(3)特殊情况元组

* 空元组

t5 = ()  
t6 = tuple()

* 无名元组

tuple()

* 省略圆括号 \*\*\*

x2 = 100, 200, 300

(4)删除元组

语法格式: del 元组名

tuple06 = (100, 200, 300)

del tuple06  
print('删除后的元组元素:', tuple06)

(5)元组运算

t7 = 100, 200, 300  
t8 = 'ABC', 'BCD'

* 支持+运算

res1 = t7 + t8  
print(res1)

* 支持\*运算

res2 = t8 \* 2  
print(res2)

8.3、字典(dict)

字典是Python中非常重要、无序、可变数据类型之一,以键值对形式进行存储,键名不可以重复,可以通过键名来,访问字典中的值

(1)定义字典

语法格式:

字典名 = {键1:值1,键2:值2,...键n:值n}

字典名 = dict(键1=值1,键2=值2,...键n=值n)

说明:字典是以{}括起来的数据集合,以逗号分隔,以键值对形式存储,键不可以重复,值可以重复,可以是不同数据类型

dict01 = {'id':101, 'name':'rose', 'age':20, 'address':'北京市朝阳区'}  
dict02 = dict(id=101, name='rose', age=20, address='北京市朝阳区')

(2)使用字典

* 通过字典名访问

print(字典名)

print(dict01)  
print(dict02)

* 通过字典名+键访问

print(字典名[键])

print(dict01['address'])  
print(dict02['name'])

* 解包

d1, d2, d3, d4 = dict01  
print(d4)

(3)特殊情况的字典

* 空字典

d5 = {}  
d6 = dict()

* 无名字典

dict()

(4)删除字典

* 通过字典名删除

del dict01  
# print(dict01)  
# NameError: name 'dict01' is not defined

* 通过字典名+键删除

print('原字典元素:', dict02)  
del dict02['name']  
print('删除后字典元素:', dict02)

(5)增加和修改字典元素

dict03 = {'A':100, 'B':200, 'C':300}  
print('原字典元素:', dict03)  
# 增加元素  
dict03['D'] = 400  
print('增加后字典元素:', dict03)  
# 修改元素  
dict03['B'] = 500  
print('修改后字典元素:', dict03)

(6)类型转换

* 字典转换字符串

dict04 = {'name':'rose', 'age':20, 'sal':7000}  
str01 = '%(name)s%(age)d%(sal)d' % dict04  
print(dict04, type(dict04))  
print(str01, type(str01))

* 字符串转字典

str02 = "{'name':'rose', 'age':20, 'sal':7000}"  
dict05 = eval(str02)  
print(str02, type(str02))  
print(dict05, type(dict05))

* 列表转字典

list01 = [  
 ['name', 'rose'],  
 ['age', 20],  
 ['sal', 9000]  
]  
dict06 = dict(list01)  
print(dict06, type(dict06))

(7)字典常见函数

* len() 获取字典元素的个数(长度)

dict01 = {'A':100, 'B':200, 'C':300}  
len = len(dict01)  
print('字典中元素的个数是:%d' % len)

* keys() 获取字典中全部的键

dict02 = {'A':100, 'B':200, 'C':300}  
keys = dict02.keys()  
print('字典中全部的键:', keys)

* values() 获取字典中全部的值

dict03 = {'A':100, 'B':200, 'C':300}  
values = dict02.values()  
print('字典中全部的值:', values)

* get() 获取字典中键对应的值

dict04 = {'A':100, 'B':200, 'C':300}  
values = dict04.get('B')  
print('键对应的值是:', values)

* items() 以键值形式获取字典中的元素

dict05 = {'A':100, 'B':200, 'C':300}  
key\_values = dict05.items()  
print('键值形式显示:', key\_values)

* pop() 根据键,删除字典中的元素

dict06 = {'A':100, 'B':200, 'C':300}  
print('删除前字典元素:', dict06)  
values = dict06.pop('C')  
print('删除后字典元素:', dict06)  
print('字典中删除键对应的值是:', values)

* update() 将一个字典中的元素,更新到另外一个字典中(存在就覆盖,不存在就添加)

dict07 = {'id':101, 'name':'mary', 'age':20}  
dict08 = {'A':100, 'B':200, 'id':101, 'age':30}  
dict07.update(dict08)  
print('新字典元素是:', dict07)

8.4、集合(set)

集合是无序、可变的数据类型之一,同一个集合中元素的是不可重复(唯一)的

(1)定义集合

集合名 = {数据1,数据2,...,数据n}

集合名 = set()

说明:集合是以{}括起来的数据集合,逗号分隔,数据不能重复,可以是不同数据类型

set01 = {100, 'abc', 200, 300, 'def'}  
set02 = set()  
set03 = {100, 200, 300, 100, 200, 300}

(2)使用集合

* 通过集合名访问

print(set01)  
print(set03)

* 解包

set04 = {100, 200, 300}  
s1, s2, s3 = set04  
print(s1)

(3)特殊情况集合

* 空集合 \*\*\*

set05 = {} # 空字典  
set06 = set() # 空集合  
print(type(set05))  
print(type(set06))

* 无名集合

set()

(4)删除集合

set07 = {100, '200', 300}  
del set07  
# print(set07)  
# NameError: name 'set07' is not defined

(5)集合运算

set08 = {1, 2, 3, 4, 5, 6, 7, 8, 9}  
set09 = {0, 2, 4, 6, 8, 10}

* 并集(|)

res1 = set08 | set09  
print('新集合元素:', res1)

* 交集(&)

res2 = set08 & set09  
print('交集结果是:', res2)

* 差集(-)

res3 = set08 - set09  
print('差集结果是:', res3)

(6)集合常见函数

* add() 向集合中添加元素

set01 = {100, 200, 300}  
set01.add('abc')  
print('添加后集合元素:', set01)

* update() 将一个集合中的元素,更新到另外一个集合中

set02 = {100, 200, 300, 400}  
set03 = {100, 'abc', 'def', 300, 500}  
set02.update(set03)  
print('更新后集合元素:', set02)

* pop() 删除集合中的一个元素

set04 = {100, 'abc', 'def', 300, 500}  
values = set04.pop()  
print('删除后集合元素:', set04)  
print('删除的元素是:', values)

* discard() 删除集合中的一个元素

set05 = {100, 'abc', 'def', 300, 500}  
set05.discard('def')  
print('删除后集合元素:', set05)

* clear() 清空集合

set06 = {100, 'abc', 'def', 300, 500}  
set06.clear()  
print(set06) *# set()*

1. **流程控制语句**

在一个程序执行过程中,各条语句的执行顺序,对程序的结果有直接的影响,需要通过流程控制语句,来控制语句(代码)的走向,来实现要完成的功能,在Python中分为3中基本结构:顺序结构、选择(分支)结构、循环结构

9.1 顺序结构

顺序结构就是:程序中语句的执行顺序,就是按照语句(代码)出现的位置,进行执行的,按照自上而下,自左而右

9.2 选择(分支)结构