**Python1-2**

**数字类型**

**1，数字**

**• int:有符号整数**

**• bool:布尔值 – True:1 – False:0**

**• float:浮点数 有小数点**

**• complex:复数 共轭复数**

**对于整数，没有任何前缀python默认识别为10进制数**

**数字以0o或0O开头表示为8进制数 权限的数值表示是八进制数**

**数字以0x或0X开头表示16进制数**

**数字以0b或0B开头表示2进制数**

**字符串 引号之间的字符集合 单双引号之间没有区别**

**三个引：连续三个引号之间可以用来包含特殊字符**

**• 使用索引运算符[ ]和切片运算符[ : ]可得到子字符串**

**• 第一个字符的索引是0,最后一个字符的索引是-1**

**• 子字符串包含切片中的起始下标,但不包含结束下标**

**下标为负数 表示从右开始取 最右边的下标为-1**

**切片 取一个片段**

**>>> py\_str[1:3] 包含起始下标 不包含结束下标**

**'yt'**

**>>> py\_str[1:] 结束下标不写表示取到结尾**

**'ython'**

**>>> py\_str[:] 开始下标不写表示从开头**

**'python'**

**>>> py\_str[1:60] 下标超出范围 切片不报错**

**'ython'**

**使用+号可以将多个字符串拼接在一起**

**使用\*号可以将一个字符串重复多次**

**>>> py\_str + ' is cool' 加号表示简单拼接**

**'python is cool'**

**>>> '\*' \* 30 \*号重复30遍**

**'\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*'**

**>>> py\_str \* 3 变量重复三次**

**'pythonpythonpython'**

**• 可以将列表当成普通的“数组”,它能保存任意数量任意类型的python对象**

**• 像字符串一样,列表也支持下标和切片操作**

**• 列表中的项目可以改变**

**• 使用in或not in判断成员关系**

**• 使用append方法向列表中追加元素**

**>>> 'y' in py\_str**

**True**

**>>> 'c' in py\_str**

**False**

**>>> py\_str[::2] 2表示步长值是2**

**'pto'**

**>>> py\_str[1::2] 从下标是1开始取 步长值是2**

**'yhn'**

**>>> py\_str[::]**

**'python'**

**>>> py\_str[::-1] 步长值为-1 表示从右向左开始取**

**'nohtyp'**

**定义列表**

**• 可以将列表当成普通的“数组”,它能保存任意数量任意类型的python对象**

**• 像字符串一样,列表也支持下标和切片操作**

**• 列表中的项目可以改变**

**列表 是一个容器 可以存放各种数据**

**>>> alist = [10,20,'niu','wang',[1.2]]**

**>>> len(alist)**

**5**

**>>> alist[2:4]**

**['niu', 'wang']**

**>>> 10 in alist**

**True**

**>>> alist \* 2**

**>>> alist.append(200) 向列表尾部增加一项**

**>>> alist**

**[10, 20, 'niu', 'wang', [1.2], 200]**

**>>> alist[-1] = 300 列表项可以重新赋值**

**>>> alist**

**[10, 20, 'niu', 'wang', [1.2], 200, 300]**

**元组 相当于是静态列表 项目不可变**

**>>> atuple = (10,20,'niu','wang')**

**>>> atuple[-1]**

**'wang'**

**>>> atuple[2::]**

**('niu', 'wang')**

**>>> atuple[0] = 100 报错 元组不能修改**

**• 字典是由键-值(key-value)对构成的映射数据类型 key不能重复**

**• 通过键取值,不支持下标操作 不支持切片 不支持 in类型**

**>>> adict={'name':'大锤','age':18}**

**>>> len(adict)**

**>>> 18 in adict 18是字典的key吗？**

**False**

**>>> 'age' in adict**

**True**

**>>> adict['sex']='female' #key不在字典中则增加一项**

**>>> adict['age']=20 #age已是字典的key 修改age的值**

**数据类型比较**

**• 按存储模型分类**

**– 标量类型:数值、字符串 只有一种类型 不能存储其它类型**

**– 容器类型:列表、元组、字典**

**• 按更新模型分类:**

**– 可变类型:列表、字典**

**– 不可变类型:数字、字符串、元组**

**• 按访问模型分类**

**– 直接访问:数字**

**– 顺序访问:字符串、列表、元组**

**– 映射访问:字典**

**• 标准if条件语句的语法**

**if expression:**

**if\_suite**

**else:**

**else\_suite**

**• 如果表达式的值非0或者为布尔值True, 则代码组if\_suite被执行;否则就去执行else\_suite**

**• 代码组是一个python术语,它由一条或多条语句组成,表示一个子代码块**

**if** -0.0: 任何值为0的数字都是False 非0为Ture  
 print(**'ok'**)

**if ' '**: 任何非空对象都是Ture 空为False  
 print(**'space is ture'**)

**if ''**: 无输入是空字串 False  
 print(**'空字串是Flase'**)

**if not None**:  
 print(**'None也是False 表示为空'**)

**1. 创建login2.py文件 2. 提示用户输入用户名和密码3. 获得到相关信息后,将其保存在变量中4. 如果用户输的用户名为bob,密码为123456,则输出Login successful,否则输出Login inorrect**

**import** getpass 导入模块getpass   
a = input (**'用户名：'**)  
b = getpass.getpass (**'密码：'**) getpass.getpass关闭终端显示  
**if** a == **'bob'and** b == **'123456'** :  
 print(**'login successful'**)  
**else**:  
 print(**'login inorrect'**)

**###############################################**

**import** random 调用模块random  
number = random.randint(1,100) 生成1-100间随机数字包含1.100  
print(**'number -> '**,number)  
answer = int(input(**'猜数1-100：'**)) 将用户的输入的字符转换成真正的数字  
**if** answer > number:  
 print(**'猜大了'**)  
**elif** answer < number:  
 print(**'猜小了'**)  
**else**:  
 print(**'猜对了'**)

###################猜对了结束运行###############################

**import** random  
number=random.randint(1,100)  
running = **True***#answer = int(input('猜数（1-100）:'))***while** running:  
 answer = int(input(**'猜数（1-100）:'**))  
 **if** answer > number:  
 print(**'猜大了'**)  
 **elif** answer < number:  
 print(**'猜小了'**)  
 **else**:  
 print(**'猜对了'**)  
 running=**False**

**条件表达式**

**• Python 在很长的一段时间里没有条件表达式(C ? X :Y),或称三元运算符,因为范·罗萨姆一直拒绝加入这样的功能**

**• 从Python 2.5集成的语法确定为: X if C else Y**

**>>> a=100**

**>>> b=80**

**>>> samller = a if a<b else b**

**>>> samller**

**80**

**简单猜拳**

**import** randomchoice = [**'石头'**,**'剪刀'**,**'布'**]  
pc = random.choice([**'石头'**,**'剪刀'**,**'布'**])  
ww = [[**'石头'**,**'剪刀'**],[**'剪刀'**,**'布'**],[**'布'**,**'石头'**]]  
pooo = **"""(0) 石头  
(1) 剪刀  
(2) 布  
请猜拳(0/1/2)"""**index = int(input(pooo))  
user2=choice[index]  
print (**'电脑猜拳：'**,pc,**'你的猜拳：'**,user2)  
**if** user2 == pc :  
 print (**'\033[31;1m平局！\033[0m'**)  
**elif** [user2,pc] **in** ww :  
 print (**'\033[31;1mYouWin!!!!\033[0m'**)  
**else**:  
 print (**'\033[31;1mYouLose!!!!\033[0m'**)

**• 一组被重复执行的语句称之为循环体,能否继续重复,决定循环的终止条件**

**• Python中的循环有while循环和for循环**

**• 循环次数未知的情况下,建议采用while循环**

**• 循环次数可以预知的情况下,建议采用for循环**

**• 当需要语句不断的重复执行时,可以使用while循环**

sum100=0  
counter = 1  
**while** counter <=100:  
 sum100 += counter  
 counter += 1  
print(sum100)

**• 语句while\_suite会被连续不断的循环执行,直到表达式的值变成0或False**

**import** random  
number=random.randint(1,100)  
running = **True***#answer = int(input('猜数（1-100）:'))***while** running:  
 answer = int(input(**'猜数（1-100）:'**))  
 **if** answer > number:  
 print(**'猜大了'**)  
 **elif** answer < number:  
 print(**'猜小了'**)  
 **else**:  
 print(**'猜对了'**)  
 running=**False**

**break语句**

**• break语句可以结束当前循环然后跳转到下条语句**

**• 写程序的时候,应尽量避免重复的代码,在这种情况下可以使用while-break结构**

**import** random  
number = random.randint(1,100)  
**while True**:  
 answer = int(input(**'猜数（1-100）：'**))  
 **if** answer > number:  
 print(**'猜大了'**)  
 **elif** answer < number:  
 print(**'猜小了'**)  
 **else**:  
 print(**'猜对了'**)  
 **break**

**•当遇到continue语句时,程序会终止当前循环,并忽略剩余的语句,然后回到循环的顶端**

**•如果仍然满足循环条件,循环体内语句继续执行,否则退出循环**

sum100 = 0  
counter = 0  
**while** counter < 100:  
 counter +=1  
 **if** counter % 2: *#结果只有0或1 0为False 1为True* **continue** sum100 += counter  
print(sum100)

**• python中的while语句也支持else子句**

**• else子句只在循环完成后执行**

**• break语句也会跳过else块**

**import** random  
number = random.randint(1,100)  
tries = 0  
**while** tries < 5 :  
 **if** acccc < number: acccc = int(input(**'请输入数字：'**))  
  
 print(**'\033[32;1m猜小了!\033[0m'**)  
 **elif** acccc > number:  
 print(**'\033[32;1m猜大了！\033[0m'**)  
 **else**:  
 print(**'\033[32;1m猜对了！!!!!\033[0m'**)  
 **break** tries += 1  
**else**:  
 print(**'正确的数字是： '**,number)

**猜拳三局两胜**

**import** random  
choice = [**'石头'**,**'剪刀'**,**'布'**]  
ww = [[**'石头'**,**'剪刀'**],[**'剪刀'**,**'布'**],[**'布'**,**'石头'**]]  
pooo = **"""(0) 石头  
(1) 剪刀  
(2) 布  
请猜拳(0/1/2)"""***#a =*b = 0  
c = 0  
**while** b<2 **and** c <2:  
 pc = random.choice([**'石头'**, **'剪刀'**, **'布'**])  
 index = int(input(pooo))  
 user2 = choice[index]  
 **while** user2 == pc:  
 **break  
 if** [user2,pc] **in** ww :  
 print(**'电脑猜拳：'**, pc, **'你的猜拳：'**, user2)  
 print (**'\033[31;1m正确\033[0m'**)  
 b += 1  
 *# continue* **else**:  
 print(**'电脑猜拳：'**, pc, **'你的猜拳：'**, user2)  
 print (**'\033[31;1m错误\033[0m'**)  
 c += 1  
 *# continue***while** b == 2:  
 print(**'\033[31;1mYOUWIN!!!!\033[0m'**)  
 **break  
while** c == 2:  
 print(**'\033[31;1mYOULOSE!!!!\033[0m'**)  
 **break**