**一、什么是自动化测试**

通过测试工具或其他手段，按照**测试人员的预定计划对软件产品进行自动化测试**，能**快速、全面**对软件进行测试，提升软件质量，节省经费，缩短软件发布周期

**二、了解为什么要进行自动化测试**

1. 缩短测试周期
2. 避免人为出错
3. 测试信息存储，交接方便
4. 轻易获取覆盖率
5. 可以自动或定时执行

**三、自动化测试的分类**

1.整体分类

* 自动化功能测试
* 自动化性能测试

2.自动化功能测试的分类

* 单元测试（开发人员）
* 功能测试
* 接口测试

大中型项目或长期项目可以采用自动化测试

3.性能测试：主要是使用测试工具Loadrunner、Jmeter等，对软件进行压力测试、负载测试、强度测试等

为了测试一个web站点的服务能力，需要模拟上千上万的请求(比如打开浏览器访问站点)，人的速度是远远达不到这样的操作的

**四、使用条件和场景**

1.使用自动化的前提条件

* 手动测试已经完成，后期不影响进度的情况下逐渐实现自动化
* 项目周期长，重复性工作交给机器实现
* 需求稳定，项目变动不大
* 自动化测试脚本复杂度比较低
* 可重复利用

2.使用自动化测试的场景

* 频繁的回归测试
* 冒烟测试
* 互联网迭代频繁
* 传统行业需求变化不大，应用频繁
* 性能测试

**五、自动化测试工具**

1.QTP（收费）：记录对应用的操作并进行回放

2.**Selenium（开源）**：一个测试框架，只需编写程序，跨平台且支持更多编程语言

3.RFT（收费）：主要用于功能测试和回归测试

4.Watir

5.Sahi

**六、元素的定位**

计算机通过一系列计数手段找到元素（按钮、输入框、模拟键盘等）

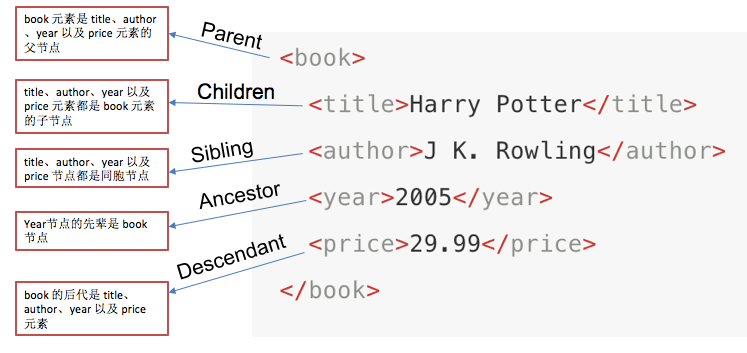
**元素定位的工具**

css选择器、xpath

自定义安装firefox，安装维护服务不勾选

xpath即为XML路径语言，它是一种用来（标准通用标记语言的子集）在HTML／XML文档中查找信息的语言

**节点**：每个XML/HTML的标签我们都称之为节点

XPath 使用**路径表达式**来选取 XML 文档中的节点或者节点集



写法：//meta[@name='author']

查找某个特定的节点或者包含某个指定的值的节点

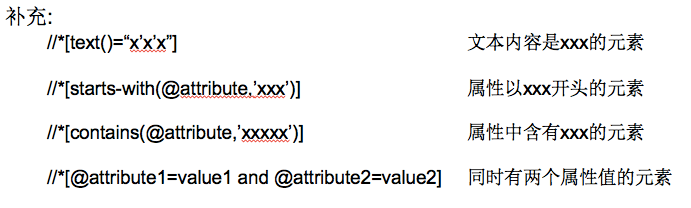


选择未知节点，通配符：统一匹配任意字符



选取若干路径





**CSS选择器**

最常用的选择器

1.通过伪类名、id、标签名定位



2.通过元素之间嵌套关系



**3.通过属性**





4.通过父子关系

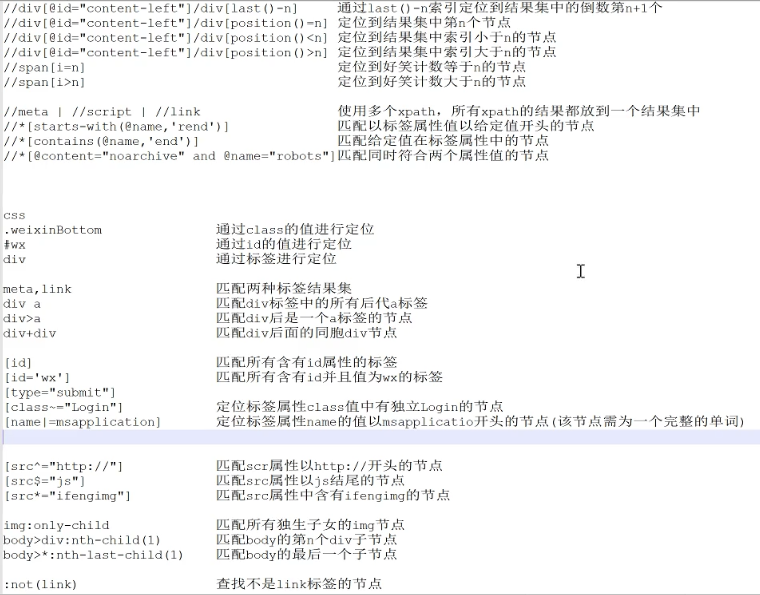


5.元素状态



6.元素状态





例：

1. //\*[@id="csdnc-alevel-14"]/path[3]
2. csdnc-alevel-14 > path:nth-child(3)

**七、Selenium自动化测试框架**

框架是**整个或部分系统的可重用设计**，表现为一组抽象构件及构件实例间交互的方法；另一种定义认为，框架是可被应用开发者定制的应用骨架。前者是从应用方面而后者是从目的方面给出的定义

**概述**：Selenium是ThoughtWorks公司的一个强大的开源Web功能测试工具系列，采用Javascript来管理整个测试过程，包括读入测试套件、执行测试和记录测试结果。

**特点**：

1. 采用Javascript单元测试工具JSUnit为核心，只要在测试用例中把预期的用户行为与结果都描述出来，我们就得到了一个可以自动化运行的功能测试套件
2. Selenium测试直接运行在浏览器中，就像真正的用户在操作一样

**Selenium API基础**

# 导入webdriver

from selenium import webdriver

# 创建一个浏览器对象

driver = webdriver.Chrome()

#设置全屏

driver.maximize\_window()

# 获取当前浏览器的尺寸

size = driver.get\_window\_size()

print(size)

# 设置浏览器的尺寸

driver.set\_window\_size(500,500)

# 文件夹

# print((dir(driver)))

# 获取浏览器的位置

position = driver.get\_window\_position()

print((position))

# 设置浏览器的位置，x轴=100,y轴=30

driver.set\_window\_position(100,30)

# 关闭当前标签／窗口

driver.close()

# 关闭所有标签／窗口

driver.quit()

from selenium import webdriver

import time

wd = webdriver.Chrome()

wd.get("https://www.baidu.com") # 打开百度浏览器

wd.find\_element\_by\_id("kw").send\_keys("selenium") # 定位输入框并输入关键字

wd.find\_element\_by\_id("su").click() # 点击【百度一下】搜索

time.sleep(3) # 在页面停留3秒

wd.quit() #关闭浏览器

**断言**

表示为一些布尔表达式，程序员相信在程序中的某个特定点该表达式值为真，可以在任何时候启用和禁用断言验证，因此可以在测试时启用断言而在部署时禁用断言

1.获取断言信息的操作

current\_url 获取当前访问页面

url title 获取当前浏览器标题

get\_screenshot\_as\_png() 保存图片

get\_screenshot\_as\_file(file) 直接保存快照

page\_source 网页源码

#需求：得到qq网页源码

from selenium import webdriver

driver = webdriver.Chrome()

url = 'http://www.qq.com'

driver.get(url)

data = driver.page\_source

print(type(data)) # 确认data是str类型

# 以二进制类型写入文件

with open('qq.html','wb') as f :

   f.write(data.encode())

# 将str类型数据（data）转换为bytes类型（二进制类型）

# bytes\_data = data\_encode()

# 将bytes类型（二进制类型）转换为str类型数据（data）

# str\_data = bytes\_data.decode()

driver.close()

**元素的定位**

#需求：得到qq网页源码

# 1.第1种写法（推荐）

from selenium import webdriver

# 定义time，设定页面停留时间

import time

driver = webdriver.Chrome()

url = 'http://www.baidu.com'

driver.get(url)

time.sleep(3)

# 定位到搜索框元素

el = driver.find\_element\_by\_id('kw')

# print(type(el))

# 向输入框中输入数据

el.send\_keys('百度贴吧')

time.sleep(3)

# 定位到搜索按钮

el\_click = driver.find\_element\_by\_id('su')

el\_click.click()

time.sleep(3)

driver.close()

# 2.第2种写法（selenium3会推广）

from selenium import webdriver

from selenium.webdriver.common.by import By

import time

driver = webdriver.Chrome()

# 获取网址

url = 'http://www.baidu.com'

driver.get(url)

# 定位输入框元素

el = driver.find\_element(By.ID,'kw')

el.send\_keys('百度网盘')

# 定位到搜索按钮，点击

el\_sub = driver.find\_element(By.ID,'su')

el\_sub.click()

time.sleep(3)

# 关闭网页

driver.close()

**id定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://www.qunar.com'

driver.get(url)

el = driver.find\_element\_by\_id('\_\_link\_tuan\_\_')

el.click()

time.sleep(3)

el1 = driver.find\_element\_by\_id('\_\_link\_piao\_\_')

el1.click()

time.sleep(3)

driver.quit()

**tag\_name定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://cn.bing.com'

driver.get(url)

# 通过标签名进行定位，该元素要么是第一个，要么唯一

el = driver.find\_element\_by\_tag\_name('input')

el.send\_keys('selenium')

time.sleep(3)

# 通过id定位到搜索按钮

el1 = driver.find\_element\_by\_id('sb\_form\_go')

el1.click()

time.sleep(3)

driver.close()

**通过link\_text定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://www.58.com'

driver.get(url)

el = driver.find\_element\_by\_link\_text('摩托卖场')

el.click()

driver.quit()

**通过class\_name定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://www.douyu.com/directory/all'

driver.get(url)

# 循环翻到下一页，操作10次

for i in range(10):

# 通过class属性对应的值定位到下一页

el = driver.find\_element\_by\_class\_name('dy-Pagination-item-link')

el.click()

time.sleep(3)

driver.quit()

**通过partial\_link\_text定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://www.baidu.com'

driver.get(url)

# 根据超链接中的部分文本进行定位，跳转网页

el = driver.find\_element\_by\_partial\_link\_text('hao')

el.click()

time.sleep(3)

driver.close()

**通过xpath定位**

from selenium import webdriver

driver = webdriver.Firefox()

url = 'https://movie.douban.com/'

driver.get(url)

el = driver.find\_element\_by\_xpath(".//\*[@id='screening']/div[1]/h2/span[2]/a")

el.click()

driver.close()

**通过class\_name定位**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://www.douyu.com/directory/all'

driver.get(url)

# 循环翻到下一页，操作10次

#for i in range(10):

# 通过class属性对应的值定位到下一页

el = driver.find\_element\_by\_class\_name('dy-Pagination-item-custom')

time.sleep(3)

driver.quit()

**获取元素信息**

from selenium import webdriver

driver = webdriver.Chrome()

url = 'https://movie.douban.com/'

driver.get(url)

el =driver.find\_elements\_by\_class\_name("list-item")

print(el)

# 对列表返回的结果进行遍历

for el1 in el:

# 得到标题和链接

print('标题：','el1\_text','链接：','el1.get\_attribute("href")')

driver.close()

**多标签切换**

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'https://sz.58.com/'

driver.get(url)

el = driver.find\_element\_by\_link\_text('深圳房产')

print('点击之前的身份证列表：',driver.window\_handles)

print('点击之前的url:',driver.current\_url)

el.click()

print('点击之后的身份证列表',driver.window\_handles)

print('点击之后的url:',driver.current\_url)

time.sleep(3)

# 进入第2个窗口

# 保存句柄列表

handle\_list = driver.window\_handles

# 通过句柄（身份证）索引进入相关的窗口

driver.switch\_to.window(handle\_list[1])

print('切换后的标题：',driver.title)

print(driver.current\_url)

time.sleep(3)

driver.quit()

**多表单、多框架切换（登录界面常用）**

# 在表单中，尤其是 登录 界面，经常会出现表单嵌套的情况，此时需要用到frame框架

from selenium import webdriver

driver = webdriver.Chrome()

url = 'http://www.126.com'

driver.get(url)

# 定位到表单

el\_frame = driver.find\_element\_by\_id('appLoginTab')

# 切换表单

driver.switch\_to\_frame(el\_frame)

# 定位到账号登录框

el = driver.find\_element\_by\_name('email')

el.send\_keys('15xxx5')

# 定位到密码输入框

el = driver.find\_element\_by\_name('password')

el.send\_keys('1xxx')

**鼠标右击操作**

from selenium import webdriver

from selenium.webdriver import ActionChains

driver = webdriver.Chrome()

url = 'http://www.baidu.com'

driver.get(url)

el = driver.find\_element\_by\_class\_name('qrcode-img')

# 鼠标右击操作，操作元素前，需要将操作'ActionChains(driver)'的元素定位出来

# 并且传入相应的动作'context\_click(el)'中，执行操作需要调用'perform()'

ActionChains(driver).context\_click(el).perform()

driver.quit()

**ActionChains—鼠标悬停操作**

from selenium import webdriver

from selenium.webdriver import ActionChains

import time

driver = webdriver.Chrome()

url = 'https://www.taobao.com/?spm=a2e15.8261149.1581860521.1.1a9129b4Eiv36l'

driver.get(url)

# 获取分类组元素

el = driver.find\_elements\_by\_css\_selector('body > div.screen-outer.clearfix > div.main > div > div.tbh-service.J\_Module > div.service.J\_Service > ul > li')

# 遍历每个元素，并执行悬停操作查看信息

for el1 in el :

ActionChains(driver).move\_to\_element(el1).perform()

driver.quit()

**Key类—键盘操作**

from selenium import webdriver

# 导入Key类，Key类中包含很多键盘按钮的操作

from selenium.webdriver.common.keys import Keys

import time

driver = webdriver.Chrome()

url = 'http://bing.com/'

driver.get(url)

el = driver.find\_element\_by\_id('sb\_form\_q')

el.send\_keys('羽绒服')

# 全选

el.send\_keys(Keys.CONTROL,'a')

time.sleep(2)

# 复制

el.send\_keys(Keys.CONTROL,'c')

time.sleep(2)

# 剪切

el.send\_keys(Keys.CONTROL,'x')

time.sleep(2)

# 粘贴

el.send\_keys(Keys.CONTROL,'v')

time.sleep(2)

# 回删一个字符

el.send\_keys(Keys.BACK\_SPACE)

time.sleep(2)

# 清空输入框

el.clear()

driver.quit()

**警告框设置**

from selenium import webdriver

import time

driver = webdriver.Chrome()

url = 'https://www.baidu.com/index.php?tn=monline\_3\_dg'

driver.get(url)

el = driver.find\_element\_by\_link\_text('设置')

el.click()

time.sleep(2)

el1 = driver.find\_element\_by\_link\_text('搜索设置')

el1.click()

time.sleep(2)

el2 = driver.find\_element\_by\_css\_selector('#gxszButton > a.prefpanelgo')

el2.click()

time.sleep(2)

# 进入警告框中，并且点击接受（确定）

# driver.switch\_to.alert.accept()

# 进入警告框中，并且点击解散（退出警告框）

driver.switch\_to.alert.dismiss()

# driver.quit()

**下拉框**

# 能定位的元素一定是在select标签下,多出现于有多个地区、省市下拉菜单等情况

from selenium import webdriver

# 导入Select类

from selenium.webdriver.support.select import Select

import time

driver = webdriver.Chrome()

url = 'http://www.baidu.com'

driver.get(url)

# 定位到'设置'元素并点击

el = driver.find\_element\_by\_link\_text('设置')

el.click()

# 定位到'搜索设置'并点击

el2 = driver.find\_element\_by\_link\_text('搜索设置')

el2.click()

time.sleep(2)

# 定位到下拉框元素

el3 = driver.find\_element\_by\_xpath('//\*[@id="nr"]')

# 创建下拉框对象

selobj = Select(el3)

time.sleep(2)

# 1.通过选项的索引进行选择，默认从0开始计数

selobj.select\_by\_index(0)

time.sleep(1)

selobj.select\_by\_index(1)

time.sleep(1)

selobj.select\_by\_index(2)

time.sleep(1)

"""

2.通过通过选项的Value进行选择,即option标签的一个属性值

selobj.select\_by\_value('10')

time.sleep(3)

selobj.select\_by\_value('20')

time.sleep(3)

selobj.select\_by\_value('50')

time.sleep(3)

3.通过文本选择(下拉框的值)

selobj.select\_by\_visible\_text('每页显示10条')

time.sleep(3)

selobj.select\_by\_visible\_text('每页显示20条')

time.sleep(3)

selobj.select\_by\_visible\_text('每页显示50条')

time.sleep(3)

"""

#查看第一个已选

print(selobj.first\_selected\_option.text)

# 查看所有已选

print(selobj.all\_selected\_options)

# 查看选项元素列表

print(selobj.options)

# 取消选择,将三者的select改为deselect即可

selobj.deselect\_by\_index(0)

driver.quit()

**执行js**

# scrollBy内部调用了scrollTo，它是基于当前位置的相对滑动,而scrollTo是绝对滑动

# 因此如果利用相同输入参数多次调用scrollTo()方法,由于View初始位置是不变只会出现一次View滚动的效果而不是多次

from selenium import webdriver

driver = webdriver.Chrome()

import time

url = 'http://news.baidu.com/guonei'

driver.get(url)

driver.maximize\_window()

time.sleep(3)

# 0代表x轴，1000代表y轴，正常网页打开不需要改变x轴，而需要往下拉查看y轴，单位是px（像素）

js1 = 'window.scrollBy(0,1000)'

driver.execute\_script(js1)

time.sleep(3)

# n为从顶部往下移动滚动,2000代表y轴的值，单位是px（像素）

js2 = 'var q=document.body.scrollTop=2000'

driver.execute\_script(js2)

time.sleep(3)

# 循环下拉,每次下拉100px

for i in range(100) :

js3 = 'window.scrollBy(0,%s)'%(i\*100)

driver.execute\_script(js3)

time.sleep(3)

driver.quit()

**隐式等待**

from selenium import webdriver

# 导入By

from selenium.webdriver.common.by import By

# 导入webdriver等待类

from selenium.webdriver.support.ui import WebDriverWait

# 导入预期条件设置类

from selenium.webdriver.support import expected\_conditions as EC

driver = webdriver.Chrome()

url = 'http://www.baidu.com'

driver.get(url)

# 定位元素,等待10s，每0.5s询问一次，超过时间检测不到则抛出异常

WebDriverWait(driver,10,0.5).until(EC.presence\_of\_element\_located((By.ID,'lg')))

driver.close()

# 隐式等待（优先使用）

# n的单位为秒,n为最大值，在这个最大值内只要元素定位到就结束等待

driver.implicitly\_wait(n)

**简单封装**

# 封装开启和关闭浏览器

from selenium import webdriver

import time

# command+alt+l 代码格式化

class Common(object):

# 初始化

def \_\_init\_\_(self):

# 创建浏览器

self.driver = webdriver.Chrome()

# 浏览器最大化

self.driver.maximize\_window()

# 导入网址链接

def open\_url(self, url):

self.driver.get(url)

# 设置最大等待时间为10s

self.driver.implicitly\_wait(10)

# 退出窗口

def close\_driver(self):

time.sleep(3)

self.driver.quit()

# 结束的时候清理窗口

def \_\_del\_\_(self):

time.sleep(3)

self.driver.quit()

if \_\_name\_\_ == '\_\_main\_\_' :

com = Common()

com.open\_url('http://www.baidu.com')

time.sleep(3)

com.open\_url('http://www.hao123.com')

**封装定位操作**

# 封装定位操作

from selenium import webdriver

import time

class Commonshare(object) :

def \_\_init\_\_(self) :

self.driver = webdriver.Chrome()

self.driver.implicitly\_wait(5)

self.driver.maximize\_window()

def open\_url(self, url):

self.driver.get(url)

time.sleep(3)

# 定义定位方式

def locateElement(self, locate\_type, value):

# 为了防止el不存在的情况，可先定义el为None

el = None

# 八种元素定位的方式，用来定位元素

if locate\_type == 'id':

el = self.driver.find\_element\_by\_id(value)

elif locate\_type == 'name':

el = self.driver.find\_element\_by\_name(value)

elif locate\_type == 'tag\_name':

el = self.driver.find\_element\_by\_tag\_name(value)

elif locate\_type == 'xpath':

el = self.driver.find\_element\_by\_xpath(value)

elif locate\_type == 'css\_selector':

el = self.driver.find\_element\_by\_css\_selector(value)

elif locate\_type == 'class\_name':

el = self.driver.find.element\_by\_class\_name(value)

elif locate\_type == 'link\_text':

el = self.driver.find\_element\_by\_link\_text(value)

elif locate\_type == 'parital\_link\_text':

el = self.driver.find\_element\_by\_partial\_link\_text(value)

# 做一个判断，返回定位到的元素

if el is not None:

return el

# 对点击元素进行封装

def click(self, locate\_type, value):

el = self.locateElement(locate\_type, value)

el.click()

# 对定位到的元素进行文本输入

def input\_data(self,locate\_type,value,data):

# 调用locateElement通过'locate\_type和value'来定位到元素

el = self.locateElement(locate\_type,value)

# 文本内容本不存在，加上data代表传入的数据

el.send\_keys(data)

# 获取定位到的元素中的文本内容（xxx），即<a>xxx</a>

def get\_text(self,locate\_type,value):

# 调用locateElement进行元素定位

el = self.locateElement(locate\_type,value)

# 执行点击操作，返回文本内容

return el.text

# 获取定位到的元素中的标签属性

def get\_attribute(self,locate\_type,value,data):

# 调用locateElement进行元素定位

el = self.locateElement(locate\_type,value)

# 返回得到的标签属性数据

return el.get\_attribute(data)

# 收尾清理页面

def \_\_del\_\_(self):

time.sleep(3)

self.driver.quit()

# 在百度搜索框中搜索'selenium'，得到结果

if \_\_name\_\_ == '\_\_main\_\_' :

com = Commonshare()

com.open\_url('http://baidu.com')

com.input\_data('id','kw','selenium')

com.click('id','su')

time.sleep(3)

**单元测试**

价值：提前介入测试，发现代码级别的bug

适用于：大型的、项目周期长，新系统、新团队，涉及财产、生命的项目

**UT开展**

**UT计划**

* 确定单元测试范围，依据“**二八原则**（20%代码中隐藏80%的缺陷）” **如何确定20%的代码**：使用频率高、复用性、开发人员的技术／业务背景、复杂度（7级以上）
* 确定评估标准：逻辑覆盖率

1）语句覆盖率（非分支判断）=被测的语句／全部的语句

2）分支覆盖率（if语句的两个分支）=被测的分支个数／全部的分支个数

3）条件覆盖率（结果为True或Flase）=被测条件数／总条件数

4）路径覆盖率（从开始到结束的过程）=被测路径数／全部路径数

5）分支-条件覆盖率（分支和条件的组和）=（被测分支数+被测条件数）／（全部分支数+全部条件数）

**UT设计**

测试策略

* 自顶向下：从最上层函数开始逐层开展测试，需要对函数中调用的函数打桩（模拟调用的函数）
* 自底向下：从最下层的函数开始逐层向上开展测试，测试周期长
* **孤立策略**：选择需要进行测试的函数即可

**UT实现**

1. 先将测试的代码转为流程图
2. 将流程图转化为流图，流图中只有圈（判定框中一条、多条的条件或者语句）和线（带有箭头指向的连接） 通过流图描述的是程序的复杂度（圈复杂度）；**路径的个数**（用例个数）描述的就是复杂度的级别，通常为**条件个数+1**
3. 确定测试路径
4. 编写测试用例

**UT执行**

借助单元测试框架：unittest

读取数据的两种方式：xml文件，csv文件

数据驱动框架：公共模块，数据模块，被测函数模块，测试用例模块，数据驱动框架

**Shell**

shell是一段应用程序，是用户和操作系统交互的一个桥梁，shell通常是就是shell脚本

shell是弱类型语言，在定义变量时**不需要指定变量类型**

**shell脚本规范**

1. shell脚本是以.sh结尾的文件
2. .sh文件的第一行内容为#!/bin/bash，表示使用shell进行编程
3. shell中的注释：#

**变量**

变量名=变量值（等号两边不能有空格）

变量命名规范：不能以数字开头，由字母、数字、下划线组成

输出变量：echo $变量名

释放变量：unset 变量名

特殊（中间有空格）变量的定义：变量名加上引号

1. 双引号：如果变量值中存在“$变量名”时，那么输出结果将是变量名对应的变量值；如果想原样输出，则需要在变量名前加上转义符\
2. 单引号：如果变量值中存在$变量名，将原样输出
3. 反单引号：将Linux命令定义为变量值

**向文件传递参数**

$1 传递第一个参数

$n 传递第n个参数

$0 显示的是文件名称

$@ 显示所有的参数

$# 显示参数的个数

**重定向：将内容输出到文件中**

正确命令：数字1表示，>覆盖，>>追加

**如：ls /tmp 1>a.txt**

错误命令：数字2表示，>覆盖，>>追加

**例1：使用Shell脚本，输出当前所在的目录**

1. vim新建.sh文件，所有的操作步骤都写在文件中
2. 进入编辑模式，在文件中编写shell脚本
3. 执行
   1. 方式1：sh 文件名.sh
   2. 方式2： 1)添加x权限：chmod +x 文件名.sh 2）执行：./文件名.sh

**例2:统计目录下有几个文件**

#命令行界面

vim etc\_num.sh

# 进入shell脚本编辑模式

#!/bin/bash

# 进入etc目录下

cd /etc

# 定义变量名， wc -l意为统计数量，即通过管道符将etc目录下的文件统计出来

filenum=`ls -l | wc -l`

# 输出内容，用双引号

echo "该目录下有$filenum个文件"

# 输入wq保存并退出

# 来到命令行界面，执行

sh etc\_num.sh

**if else分支**

# 判断任意目录是否为空

#!/bin/bash

read -p '请输入目录:' dir

d=`ls $dir`

if [ "$d" == "" ]

then

       echo "目录为空“

else

       echo "目录不为空“

fi

1. **字符串**的比较：==判断是否相等，！=判断是否不相等
2. 逻辑运算符：且-a 或者-o 非!
3. 数字比较运算符： -eq 判断是否相等 -ne 判断是否不相等 -gt 判断是否大于 -ge 判断是否大于等于 -lt 判断是否小于 -le 判断是否小于等于
4. 数字变量操作 ((i++1)) ((i+=1)) let i=i+1

#!/bin/bash

read -p "请输入数字:" num

if [ "$num" -gt 1 ]

then

      ((num--))

       echo $num

else

      ((num++))

       echo $num

fi

* 测试文件状态：test 参数 变量名 -e 表示文件是否存在 -f 表示文件是否存在且为文件 -d 表示文件名是否存在且为目录 -r 可读 -w 可写 -x 可执行

#!/bin/bash

read -p "请输入目录名称：" dir

if test -d $dir

then

       filename=`ls -l $dir | wc -l`

       echo "目录$dir下有$filename个文件"

else

       echo "该目录不存在"

fi

**if elif else分支**

#!/bin/bash

read -p "请输入分数:" score

if [ "$score" -ge 90 -a "$score" -le 100 ]

then

       echo "优秀"

elif [ "$score" -ge 80 -a "$score" -lt 90 ]

then

       echo "良好"

elif [ "$score" -ge 70 -a "$score" -lt 80 ]

then

       echo "尚可"

elif [ "$score" -ge 60 -a "$score" -lt 70 ]

then

       echo "刚及格"

else

       echo "不及格"

fi

**文件判断**

-r filename 文件是否存在且可读

-w filename 文件是否存在且可写

-x filename 文件是否存在且可执行

-d filename 文件是否存在且为目录

-f filename 文件是否存在且为普通文件

-e filename 文件是否存在

-n str1 字符串长度是否不等于0

-z str2 字符串长度是否等于0

#!/bin/bash

read -p "请输入文件名称：" file

if [ -z "$file" ]

then

echo "为空"

elif [ -d "$file" ]

then

echo "是目录"

elif [ -f "$file" ]

then

echo "是普通文件"

else

echo "错误信息"

fi

**case分支**

1. case这一行最后必须为in
2. ；；表示结束语句
3. \*表示默认
4. esac代表case语句的结束
5. 值可以为字符串或数字，也可为区间范围[0-9]

# 1.根据选项挑选物品

#!/bin/bash

read -p "请输入选项：" pro

case $pro in

"a")

       echo "笔记本";;

"b")

       echo "电脑";;

"c")

       echo "小台灯";;

esac

# 2.区间范围打分

#!/bin/bash

read -p "请给出评分:" com

case $com in

[0-3])

       echo "不满意";;

[4-6])

       echo "满意";;

[7-9])

       echo "非常满意";;

esac

**for循环**

#找到1-20中的所有奇数

#!bin/bash

# 1和20代表首位数字，2是步长，代表数字的间隔

for num in `seq 1 2 20`

do

       echo $num

done

**while 循环**

# 求1-100的累加和

#!/bin/bash

i=1

sum=0

while [ "$i" -le 100 ]

do

       sum=$(($sum+$i))

      ((i++))

done

       echo $sum

**函数**

先定义函数，再调用

# 加法函数

#!/bin/bash

add(){

       s=$(($num+$num2))

       echo $s

}

read -p "请输入第一个数据：" num

read -p "请输入第二个数据：" num2

add

**函数的返回值**

函数正确时返回0

错误时返回1

查看返回值内容：$?

hello2(){

       echo "他手里有人质，我只能眼睁睁地看着"

# 此处填写错误信息，则返回1

cd xpp

}

hello2

echo $?   #函数正确时返回0

**例：创建文件函数**

#!/bin/bash

# 先创建一个函数判断文件是否存在，作为嵌套内容

createfile(){

       # 进入目录

       cd $1

       # 提示用户输入文件名

      read -p "请输入文件名称：" filename

       # 判断文件是否存在

       if [ -f "$filename" ]

       then

               echo "文件已存在"

       else

               echo "文件不存在"

               echo "开始创建文件……"

               touch $filename

               echo "文件创建成功"

       fi

}

# 提示用户输入

read -p "请输入目录：" dir

if [ -d "$dir" ]

then

       # 调用创建文件函数

      createfile $dir

else

       echo "目录不存在"

fi

**cut**

-b 按字节进行截取；一个空格／字母=1个字节，一个汉字=3个字节

-c 按字符进行截取；一个汉字／空格=1个字符

-前不写数字代表截取内容从第一个字节到该位置

-后不写数字代表截取内容从该位置到最后一个字节

截取相同内容时会自动去重

例：截取前五行内容的1至3个字节和第5个字节

head -5 passwd|cut -b 1-3,20-

按照域进行剪取

-d 后面跟剪取的分隔符

-f 后面跟取的域的位置

例：按照冒号剪取，取第一个域

head -5 passwd|cut -d ":" -f 1

**向文件中插入内容**：sed -i "行号a内容" 文件名

例：向文件.bash\_profile第10行中插入内容

sed -i "10aexport ORACLE\_BASE" .bash\_profile

**更新文件**：source ./.bash\_profile(文件路径)

#!/bin/bash

# 定义函数

creategroup(){

       # 创建管理组

      groupadd dba

       finddba=`grep "dba" /etc/group|cut -d ":" -f 1`

       # 判断

       if [ "$finddba" == "dba" ]

       then

               echo "创建管理组成功"

               #创建安装组

              groupadd oinstall

               # 查找组并定义变量

               findoin=`grep "oinstall" /etc/group|cut -d ":" -f 1`

               if [ "$findoin" == "oinstall" ]

               then

                       echo "安装组创建成功"

               else

                       echo "安装组创建失败"

               fi

       else

               echo "创建管理组失败"

       fi

}

# 判断当前用户是否为root

if [ "$USER" == "root" ]

then

       echo "当前用户是root"

       # 调用函数

      creategroup

else

       echo "当前用户不是root"

fi

**接口性能测试**

**接口测试**

定义：是测试**系统组件间接口**的一种测试，主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点，测试的重点是要检查**数据的交换、传递、控制管理过程**，以及系统间的相互**逻辑依赖**关系

接口分类：

1. 系统内部和系统外部接口 系统内部接口，模块间的相互调用 系统外部接口又包括软件接口和硬件接口
2. 软件接口又包括服务器接口和外部接口 **服务器接口**：是浏览器与服务器的接口，通过http协议来实现浏览器和服务器间的数据传递 **外部接口**：外部接口常见于通过第三方登录、第三方支付等，通过调用第三方接口并返回当前的系统

目前流行的软件接口测试原理：通过测试程序模拟客户端向服务端发送请求，服务器接收请求后做出相应的处理并将信息发送给客户端，客户端接收响应数据的过程

RESTful架构特点：

* 资源：网络实体，文本、图片、歌曲、服务等， JSON是现在最常用的资源表示格式
* 统一接口：HTTP方法中GET用来获取资源，POST用来新建资源（也可以用于更新资源），PUT用来更新资源，DELETE用来删除资源
* URI：统一资源定位符
* 无状态：即所有的资源，都可以通过URI定位，而且这个定位与其他资源无关，也不会因为其他资源的变化而改变

**http协议**

http请求由三部分组成，分别是：请求行、消息报头、请求正文

格式：Method Request-URI HTTP-Version CRLF

* GET 请求获取Request-URI所标识的资源
* POST 在Request-URI所标识的资源后附加新的数据
* HEAD 请求获取由Request-URI所标识的资源的响应消息报头
* PUT 请求服务器存储一个资源，并用Request-URI作为其标识
* DELETE 请求服务器删除Request-URI所标识的资源
* TRACE 请求服务器回送收到的请求信息，主要用于测试或诊断
* CONNECT 保留将来使用
* OPTIONS 请求查询服务器的性能，或者查询与资源相关的选项和需求

**HTTP响应**

也是由三个部分组成，分别是：状态行、消息报头、响应正文

格式：HTTP-Version Status-Code Reason-Phrase CRLF

HTTP-Version表示服务器HTTP协议的版本；Status-Code表示服务器发回的响应状态代码；Reason-Phrase表示状态代码的文本描述。

状态代码有三位数字组成，第一个数字定义了响应的类别，且有五种可能取值：

* 1xx：指示信息–表示请求已接收，继续处理
* 2xx：成功–表示请求已被成功接收、理解、接受
* 3xx：重定向–要完成请求必须进行更进一步的操作
* 4xx：客户端错误–请求有语法错误或请求无法实现
* 5xx：服务器端错误–服务器未能实现合法的请求

**常见状态码：**

* 200 OK //客户端请求成功
* 400 Bad Request //客户端请求有语法错误，不能被服务器所理解
* 401 Unauthorized //请求未经授权，这个状态代码必须和WWW-Authenticate报头域一起使用
* 403 Forbidden //服务器收到请求，但是拒绝提供服务
* 404 Not Found //请求资源不存在，eg：输入了错误的URL
* 500 Internal Server Error //服务器发生不可预期的错误
* 503 Server Unavailable //服务器当前不能处理客户端的请求，一段时间后可能恢复正常

eg：HTTP/1.1 200 OK （CRLF）

**聚合数据**

可用于测试接口：[www.juhe.cn](applewebdata://C9F08F54-20E1-44C0-BB7D-95B7421A9A45/www.juhe.cn)

先在终端中导入第三方库：pip3 install requests

pip3 install xlrd(easy\_install xlrd)

**GET请求**

# 导包

import requests

# 给接口地址定义变量名称

url = "http://v.juhe.cn/historyWeather/province"

para = {"id":"3","key":"0f16ea4fd8bd3707d32a1ca5854b7fb2"}

# 发送请求

r = requests.get(url,params=para)

print(r.status\_code)

# 获取json数据

print(r.json())

**Post请求**

# 导包

import requests

# 输入网址及数据

url = "http://v.juhe.cn/historyWeather/citys"

para = {"key":"0f16ea4fd8bd3707d32a1ca5854b7fb2","province\_id":"6"}

# 获取json数据

r = requests.post(url,data=para)

res = r.json()

# print(res)

print(res['data'][0]["context"])

**eg：城市天气案例**

# 导包

import unittest

import requests

# 定义类

class Test(unittest.TestCase):

   def setUp(self):

       print('begin')

   def tearDown(self):

       print('end')

   def test001(self):

       url = 'http://v.juhe.cn/historyWeather/citys'

       para = {'key':'0f16ea4fd8bd3707d32a1ca5854b7fb2','province\_id':'16'}

       # 使用post方法输出数据

       r = requests.post(url,data=para)

       res = r.json()

       # 与预期输出结果进行比较

       self.assertEqual(res['reason'],'查询成功')

       # 查询结果中是否包含某个内容,根据索引逐层定位结果中的字典、列表内容

       self.assertIn("已签收"，res['data'][0]['context'])

if \_\_name\_\_ == '\_\_main\_\_':

   unittest.main()

**代码与数据分离：excel表**

# 导包

import xlrd

# 打开excel表

book = xlrd.open\_workbook("../DataXML/data.xlsx")

# 定位sheet表

table = book.sheet\_by\_name("工作表1")

# 统计有多少行

print(table.nrows)

# 统计有多少列

print(table.ncols)

# 统计第一行的内容

print(table.row\_values(0))

**Jmeter实战**

启动：在bin目录下双击jmeter.bat即可

测试计划：所有的请求或其他操作都放在这里

线程组：所有的操作都属于一个线程

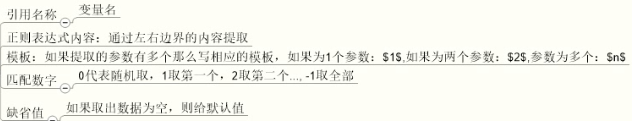
取样器sampler：存放的是请求

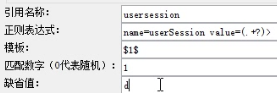
断言：对结果进行判断

监听器-断言结果：对结果进行收集

**接口间参数传递**

正则表达式提取器（在后置处理器中）





http请求中，名称为“userSession”，值为 “${变量名}”

**数据参数化**

**第一种方法**：txt／csv文件读取

1. 在桌面添加txt文件，分行添加数据，用逗号隔开
2. 线程数为1，用例数=循环次数
3. 添加“CSV数据文件设置”，填写文件**路径+变量名**
4. 将变量名添加到http请求**值**中，格式为：**${变量名}**，运行



**第二种方法**：

通过“选项—函数文件助手—CSVRead函数”读取时，文件列号从0开始，用例数=线程数，修改循环次数为1

**第三种方法**：

“线程组—前置处理器—用户参数”，添加用户和变量，用例数=线程数，修改循环次数为1

**性能测试**

定义：基于协议模拟用户发出请求（业务的模拟），对服务器形成一定的负载，来测试服务器的性能指标是否满足要求

关注点：时间性能、空间性能，与界面无关

1. **负载测试**：通过在被测系统上不断加压，直到性能指标达到极限，例如“响应时间”超过预定指标或某种资源已经达到饱和状态 **配合性能调优使用**
2. **压力测试（强度测试）**：测试系统在一定饱和状态下，例如CPU、内存在饱和使用情况下，系统能处理的会话能力，以及是否会出现错误，**帮助系统的资源使用达到较高水平、测试系统的稳定性**
3. **并发测试**：通过模拟用户并发访问，测试**多用户并发访问**同一个应用、同一个模块或者数据记录时是否存在死锁或其他性能问题
4. 基准测试：性能测试出现问题时做基准测试
5. **配置测试**：通过对被测系统的**软硬件环境**的调整，了解各种不同系统的性能影响的程度，从而找到系统各项资源的最优分配原则
6. **可靠性测试**：在给系统加载一定业务压力的情况下，使系统运行一段时间（2—3天），以此检测系统是否稳定 **保证系统长期处于一个稳定的状态**

eg：**理发店模型**

**性能测试指标**

1. 单位时间内完成事务的数量tps(Transaction per second) 吞吐量：描述的是服务器的处理能力
2. 并发用户：由多个浏览器同时发送到服务器的任务
3. 平均事务响应时间：服务器处理一个浏览器请求的时间，**特点是随着并发用户的增加而增加**
4. 单位事务数：单位时间内服务程序完成客户端请求的数量，**随着并发用户的增加而增大，当并发用户数量达到一定量后不再增加**
5. 响应时间=网络传输时间+服务器处理时间，不包括前端页面的渲染时间，到浏览器收到请求后响应数据截止 原则：2-5-8
6. 资源利用率：在一定的负载情况下，服务器资源占用情况 1） cpu利用率：不允许超过70—80%，队列长度 2） Mem（内存）利用率：80%以下，页交换频率（虚拟内存） 3） 宽带利用率：100Mbps=12.5MB/s,1Byte=8bit 4） 如果资源利用率太小，造成资源浪费
7. 并发用户数：在同一时间向服务器发送请求的用户数量，**与每秒的并发请求数不同，一定要确认需求的目的是并发用户数还是并发请求数**

**2/8原则**：80%的用户请求，集中在20%的热点数据上，或时间段

eg:每天完成交易额2亿

1. 客单价为300，求每秒最大交易数200000000／300\* 0.8 ／（20%\*24）／3600=30
2. 每天8小时系统需支持500万用户访问，求每秒最大用户访问量 5000000\*0.8/（20% \*8）/3600=694

**性能测试流程**

**测试场景**

单一场景：登录、注册、搜索、添加购物车、下单、支付

混合场景：用户使用场景、系统使用场景

**测试计划**

测试目标

测试人员组织

测试进度安排

压力机：配置、要求、数量

风险

**测试工具**

loadrunner、jmeter

**测试环境**

数据库、服务器、架构设计，有条件的情况下尽量和生产环境一致

**用例设计**

基于脚本的用例、基于场景的用例

**测试执行**

脚本编写、场景监控设计、运行场景、监控场景、测试报告

**定位分析问题**

后端：代码、软件、硬件

前端

网络

**Loadrunner**

Virtual User Generator（虚拟用户产生器）：脚本编辑区，开发调试脚本

Loadrunner Controller（控制器）：设置需要使用的并发用户的数量，这些并发用户要向服务器请求什么功能（执行哪个脚本），对执行脚本的并发用户下命令，并产生数据

Loadrunner Analysis（分析器）：分析控制器在一段时间内记录的性能数据（转化为直观的的图，如折线图等）

根据以往项目探测项目属于什么协议，也可使用协议探测工具，浏览器<http://localhost:1080/webtours(>项目名称)，询问开发也可

**工作原理**：由Vugen生成脚本，并对脚本进行优化，然后由Controller下达指令由Load Generator执行脚本，执行脚本过程生成的数据由Analysis进行结果分析

**运行脚本时的设置**

1. Run logic：设置Action的迭代次数
2. pacing：迭代之间的步长
3. 日志log—Enable logging：无论是否有错误记录日志、只有出错时才记录日志
4. Think time：函数之间的思考时间

**函数**

* web\_url()：发送get请求
* web\_submit\_data()：发送get或post请求
* web\_custom\_request()：可以发送任何请求

**参数化：数据和代码分离**

1. 在需要定义的变量处点击p键，定义变量名，并输入变量值
2. 数据用{变量名}表示，参数化文件中的测试数据最后一行多加一个空行，参数的位置随意
3. lr\_output\_message("name")用来输出内容name
4. lr\_eval\_string("{user}")用来获取参数的值user
5. 9种取值策略，取值顺序为unique时，通常用来测多用户并发的情况，需要打开controller进行查看；此时需要考虑值被用光的策略（放弃、循环、使用最后一个值继续）；**注册功能只能用unique开头的策略**
6. atoi把字符串转换成整数，itoa把整数转换成字符串

Action()

{

 //1. 将变量i的结果输出到屏幕上

 for(int i=0,i<10;i++){

   lr\_output\_message("值为:%d",i)；

}

//2. 在脚本编辑区用到的变量必须放在首行，且定义变量名及类型，多个语句之间用分号隔开

int i;

for(i=1;i<=7;i++){

web\_url("点击","URL=https://tieba.baidu.com/f?kw={name}",LAST);

//3. 将变量结果输出在log中

lr\_output\_message(lr\_eval\_string("{name}"));

 //4. 日期datatime、随机数randomnumber、唯一数uniquenum（需要在controller中查看log）同理

 //   update value—each iteration：每次迭代值发生变化

 //   update value-each occurrence：每次出现值发生变化，跟迭代无关

 //   update value-once：一旦取到值不再发生变化

   lr\_output\_message(lr\_eval\_string("{datatime}"));

}

return 0;

}

**关联：获取实时值的方式**

输入用户名和密码进行登录，需要实时获取上一个请求的响应数据中的userSession对应的值，将其放到登录的请求数据中即可

注册型函数：web\_reg\_save\_param()

特点：哪个请求的响应数据有我们需要的数据，那么注册型函数就放在web\_url()请求函数的前面

Action()

{

 //session为关联值的变量名

 web\_reg\_save\_param("session",

 //LB 、RB即变量左右两边的内容，Ord代表选择第几个变量

"LB=name=userSession value=",

   "RB=>",

   "Ord=1"

   LAST)；

   //打开网站，reg函数与web\_url中间不能有其他函数

 web\_url("打开网站"，"URL=http://baidu.com/webtours/",LAST);

 web\_submit\_data("登录"，

 "Action=http://baidu.com/webtours/login.p1",

  ……

 ITEMDATA,

 //此处传入的是session的关联变量 ，需要加上大括号

 "Name=userSession","Value={session}",ENDITEM,

 ……

)

}

**自动关联**

1. 录制以后关联，必须先运行脚本一次
2. vuser菜单下，点击scan script for correlation选择扫描脚本，复制标黄内容右键选择create correlation即可

**高级关联：获取多个值—数组**

编写脚本，选择insert-函数（web\_reg\_save\_param），填写名称和左右边界，勾选instance并将1改为all，点击ok即可（名称作为下个链接的变量）

lr\_paramarr\_idx("fights",2)，返回指定位置的参数，类型是字符串，索引从1开始,需要进行转换

lr\_paramarr\_len("fights")，获取的是数组的长度

lr\_paramarr\_random("fights")，返回任意位置的参数，类型是字符串,需要进行转换

lr\_save\_string（“fights”,"fig"）：将字符串fights保存到参数fig中，fig作为变量写入关联函数的value值中

**举例**

1. 作为定义的变量应当先放在首行：char \* str;
2. 数组中有四个参数，选择数组第2个位置的内容： 1）str=lr\_paramarr\_idx("fights",2) 2）str=lr\_paramarr\_idx("fights",lr\_paramarr\_len("fight")-2)
3. 对字符串进行参数化转换，将字符串fights转换为参数f，f作为变量插入下一行value中：lr\_save\_string(lr\_paramarr\_idx/random("fights"),"f")



**事务**

**定义**：测试人员可以将一个或多个操作步骤定义为一个事务，在程序上，事务表现为被开始标记和结束标记圈定的一段代码区块。

Loadrunner根据事务的开头和结尾标记，计算事务响应时间、成功／失败的事务数

insert—Start transaction—End transaction，两个位置的名称应保持一致

**事务开始点函数**：lr\_start\_transaction

**事务结束点函数**：lr\_end\_transaction

**注册型检查点函数**： web\_reg\_find

结合检查点，看成功率和失败率

**思考时间 lr\_think\_time**

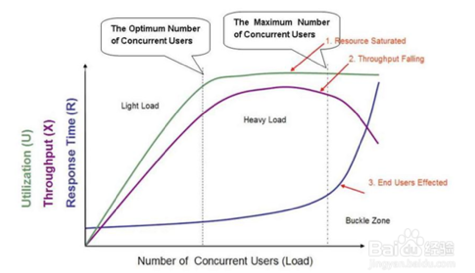
1. 实际情况下，获取真实场景的性能指标时一般加入思考时间1—2s
2. 做压力测试获取系统的峰值指标时，一般忽略思考时间
3. 思考时间不放入事务内，以免影响计算事务响应时间

**集合点 lr\_rendezvous**

帮助更好地**模拟多用户并发**操作，运行过程中多个用户可以在集合点等待到一定的条件后再一起发后续的请求

点击Controller—scenario—Rendezvous—policy添加

**压力曲线分析图**



1. Utilization，包括硬件资源和软件资源，表示资源的利用情况；Throughput，这里是指每秒事务数，表示服务器的吞吐量；Response Time即响应时间
2. 横轴从左到右表现了并发用户数（Number of Concurrent Users）的不断增长
3. 第一条虚线左边的区域为轻压力区，此时出现了第一个拐点，为最佳并发用户测试点
4. 第二条虚线左边的区域为重压力区，此时出现了第二个拐点，吞吐量开始放缓增长，为测试系统负载压力的最佳点
5. 第二条虚线右边的区域为崩溃区，此时用户响应时间越来越长，系统负载达到处理极限，用户排队，服务器宕机

**Controller**

1. initialize：什么时候运行vuser
2. Start Vusers：什么时候开始执行Action
3. Duration：选2表示持续多长时间；Action执行1次是否重复
4. Stop Vusers 1）Simultaneously—立即结束 2）每隔多长时间结束几个
5. Add action：添加action

**IP欺骗**

**企业环境搭建**

**虚拟机：**

在现有的操作系统上，借助指定的系统光盘镜像，通过软件模拟出来**具有完整硬件系统功能**的、一个运行在**完全隔离环境**中的、具有真实操作系统完全一样功能完整的计算机系统

**Linux（一切皆文件）**

内核：是运行程序、管理镜像磁盘和打印机等硬件设备的核心程序，它提供了一个在裸设备与应用程序之间的抽象层

操作系统是计算机的核心，而内核是操作系统的核心

Linux内核版本号由3个数字组成：r，x，y

1. r：目前发布的内核主版本
2. x：偶数表示稳定版本，奇数表示开发中的版本
3. y：错误修补的次数

**Linux基础命令**

操作系统中负责**管理和存储文件**的方法就是文件系统

**绝对路径**：从根目录开始（例：/Users/chenzhenyuan/pan）

**相对路径**：./pan（’.‘代表当前目录，‘..‘代表上级目录）

放大终端：command + shift+ '='

缩小终端：command +'-'

etc：系统配置文件存放的目录，不建议在此目录下存放文件

**pwd：查询当前所在位置**

**./文件名：代表执行（进入）这个文件**

**cd**

1.切换到用户自己的目录：cd 或者 cd ~

2.切换到上级目录：cd ..

**touch**

1.创建一个‘a.txt’文件：touch a.txt

2.创建多个文件：touch a.txt b.txt

3.创建以a开头的结尾为2—9的文件：touch a{2..9}

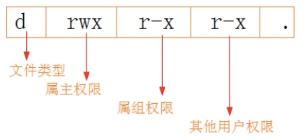
4.在文件（大括号，在此表示占位符）后追加任意内容：touch a{2..9}.txt

**ls**

1.展示当前目录：ls

2.展示当前目录包括隐藏目录：ls -a

3.展示当前文件详情（大小、位置）：ls -lh



d表示目录，此处的‘ - ’表示文件

r ：读权限 代表文件可以读

w：写权限 代表文件可以写

x：执行权限，若是目录，无x则不可进入

+：增加权限

-：减去权限

=：设定权限

u：属主，文件归谁所有

o：属组，文件属于哪个组

g：其他用户

**chmod**

给pan的组增加执行权限：chmod o+x pan

**chown（在root用户下）**

将root用户下的pan的权限改为admin用户中的admin组（修改属组权限要用R）：chown -R admin.admin pan

**mkdir**

1.创建一个名为‘nihau’的目录：mkdir nihao

2.创建多个目录：mkdir nihao1 nihao2

3.创建以a开头的结尾为2—9的目录：touch nihao{2..9}

4.在文件（大括号，在此表示占位符）后追加任意内容：mkdir a{2..9}.py

5.创建层级目录（新增包含ban的nihao目录）：mkdir nihao/ban -p

**vim**

选择一个文件进行编辑：vim a1.txt，进入文件后，'a'表示输入（转换为INSERT模式），‘esc键'表示退出编辑模式，‘shift+：’表示进入命令行模式，可在后面输入命令操作，删除冒号即可退回未编辑模式

**1.有文件则打开，无文件则创建一个新文件并编辑：vim a.txt**

**2.在命令行（冒号）模式下**

保存文件并退出：wq

不保存文件直接退出：q

强制退出：q!

撤销上一步操作：u

**3.esc键退出编辑模式后**

复制：yy (光标要放在该行)

粘贴：p

删除：d

返回行首：shift+6

返回行尾：shift+4

返回首行首字符：gg

返回最后一行首字符：G

搜索关键字：／yet(要找的内容)，会自动寻找光标所在行的内容

**cat**

1.把'a.txt'的文件内容打印到屏幕上：cat a.txt

2.显示a.txt文件中内容及每行内容：cat -n a.txt

**head**

查找文件指定行数内容：head -n 3 qg.txt（3代表前3行，不写默认查找前10行）

**tail**

查看文件末尾行内容：tail -n 3 qg.txt（3代表后3行，不写默认查找后10行）

**more**

分屏查看a.txt文件内容：more a.txt（根据终端屏幕大小显示内容，enter代表往下翻，q代表退出查看）

**mv**

1.移动a.txt文件到指定目录pan中：mv a.txt pan（若pan不存在，则创建）

2.修改目录名称ni为hao:mv ni haoIMG_256IMG_256(hao不存在)

**rm**

删除目录pan及其中所有文件：rm -rf pan（r代表递归，每一层，删除单个文件时可不加）

删除目录pan下的pp8文件：rm -rf pan/pp8

删除上一级目录中pan下的pp8文件：rm -rf ../pp8

删除大括号里的所有目录：rm -rf {a,b,nihao1,nihao2,pp1,pp2,pp9}

**cp**

拷贝文件ni到hao中：cp ni hao（当前目录下的拷贝的文件不能重名，hao不能存在）

拷贝目录ni到目录hao中：cp -r ni hao（目录拷贝必加-r）

**tar**

压缩：tar zcvf ppp.tar.gz ppx

tar为压缩命令，zcvf 为参数（v指在终端上显示解压过程，可不加），ppp.tar.gz为文件压缩后的名称，ppx为要压缩的文件

解压ppp.tar.gz文件：tar xf ppp.tar.gz（ppp文件不能与目录中的文件重名）

**zip**

将pix目录压缩为ppx（不需要加后缀）：zip -r ppx pix

解压ppx目录（文件夹中不能重名）：unzip ppx

**软连接文件**

对文件pan设置快捷键pan-1：ln -s pan-1 pan(删除原文件pan后快捷键不可用)

**wget**

下载文件：wget +网址（需要先确认VMware服务中联网）

**查找用户**

1.id 2.whoami 3.w

**切换用户**

切换用户：su - 用户名

创建新的普通用户：useradd 用户名（必须在root目录下）

设置新用户的密码：passwd 用户名

退出：exit

**find**

例：find 路径 -name "文件名（首尾可用\*来占位）"

查找pan目录：find / -name "\*pan"

**grep |(管道符)：与其他命令结合使用**

例1：命令1 | grep 关键字

命令1执行后的内容，通过管道符，传递给命令2使用

1）列出所有包含bit字符的文件：ls | grep bit

2）找到所有文件“aa"中包含box的文件：find /user -name "aa" | grep 'box'

例2：grep 关键字 文件名

1）过滤掉文件1.txt中包含a的内容：grep a 1.txt

2）过滤掉文件1.txt中不包含a的内容(v表示取反，n显示行数）：grep -vn a 1.txt

过滤的内容会在屏幕上显示

**echo:将后面的内容输出到当前屏幕**

将"she is a good girl"放到aa.txt文件中，若无aa.txt则创建：echo "she is a good girl" > aa.txt

**name -a/-r：查询系统内核（linux）版本号**

**ifconfig：查找本机网络信息包括IP地址**

**网络基础知识**

常见通信方式：网线、蓝牙、wifi

如何通信：通过ip和端口进行通信

ip地址的格式：192.168.8.14（每组数字在0—255之间）

当前局域网的ip地址段：192.168.x.x

当前系统的本地回环地址：127.0.0.1

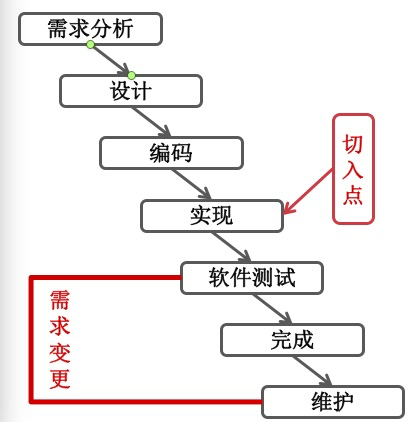
**yum**

安装软件：yum install 软件名

删除软件：yum remove 软件名

查看软件信息：yum info 软件名

**瀑布模型**



1.是线性模型的一种，是其他模型的基础，很重要

2.每个阶段执行一次，按线性顺序进行软件开发

**一、测试的切入点：**

测试阶段必须处于软件实现后，预留充分时间进行软件测试

**二、优点：**

开发阶段清晰、强调早点计划和需求调查、**适合稳定**的产品需求

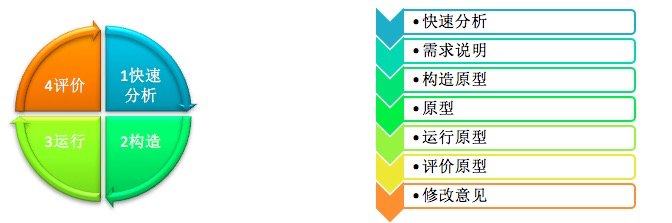
**三、缺点：**

单一过程不可逆、问题到后期才暴露，失去及早纠正的机会

**四、改良**

沿用瀑布模型的线性思想，细化各个阶段，在重要阶段掺入迭代的思想，**尽早介入**

**快速原型模型**

****

**一、定义**

先构建一个快速原型，实现用户与系统的交互，用户对原型进行评价，进一步细化需求，根据用户需求逐步调整

**二、优点**

克服了瀑布模型的缺点，更好地满足用户的需求并减少由于软件需求不明确带来的项目开发风险。**适合预先不能确切定义需求的软件系统开发**

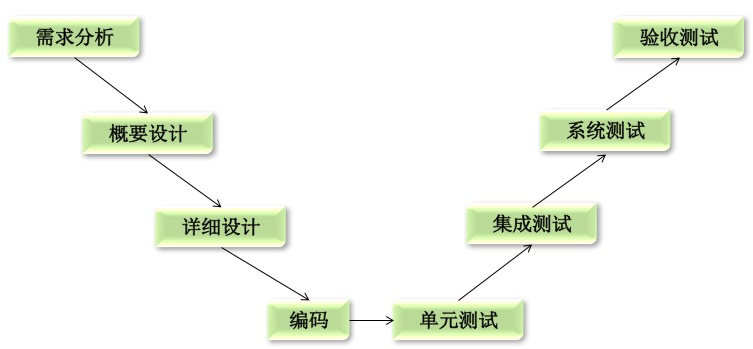
**三、缺点**

**不适合大型系统的开发（多用于小型、灵活性高的系统）**前提是要有一个可展示的产品原型，会限制开发人员的创新

**测试模型**

常见： **V模型**、**W模型**、**H模型**

**V模型示意图**



1、需求分析：用户需求、业务需求、需求规格说明书

2、概要设计：系统架构、模块分析、模块与模块之间的接口

3、详细设计：模块内部实现的逻辑和方法

4、编码：用代码形式呈现设计

5、单元测试：检测代码的开发是否符合详细社戏的要求

6、集成测试：检测此前测试过的各部分是否能结合

7、系统测试：检测已集成的产品是否符合规格说明书

8、验收测试：检测产品是否符合用户最终需求

**一、优点**

1、包含了底层测试（**检验源代码质量-单元测试**、**检验整个系统的需要-系统测试）**

2、清楚地标示了软件开发的各阶段

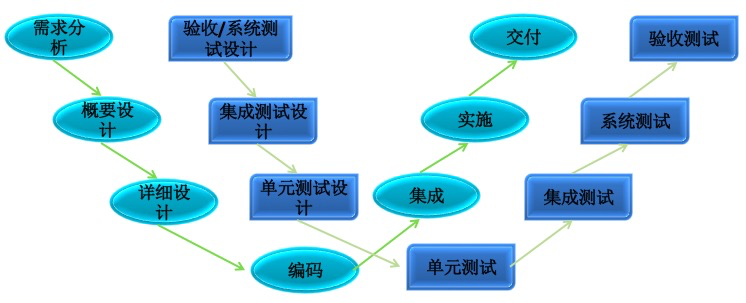
3、线性模式，阶段工作明确便于控制开发过程

**二、缺点**

1、强调顺序，前期的错误到后期才能发现

2、实际开发过程中，用户需求多变，返工量大，模型灵活度较低

**W模型**



开发一个V，测试一个V，组合成W模型

**一、优点**

1、 测试伴随着整个软件开发周期，并且测试的对象不仅仅是**程序，需求和设计**同样要测试

2、更早介入测试，更早发现开发初期缺陷

3、便于控制项目过程

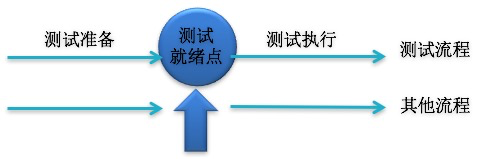
**二、缺点**

1、线性关系，无法支持迭代、自发性和需求变更调整

2、许多项目在实际执行过程中不产生文档，难以适用

3、实际使用技术复杂度（需求和设计的测试要求）高，实践困难

**H模型**



**一、概念**

测试流程：

1. 测试准备：所有测试执行活动的准备，判断是否可以开始执行测试的条件
2. 测试就绪点：测试准入准则，即是否可以开始测试的条件
3. 测试执行：具体执行测试的程序
4. 开发中的其他流程，如：设计流程

**二、优点**

* 展示了软件测试的其他工作
* 测试过程完全独立，贯穿整个生命周期，且可与其他流程并发进行
* 测试可尽早准备，尽早执行，灵活性高
* 软件测试可根据被测物的不同而分层次、分阶段、分次序执行，可以迭代

**三、缺点**

* 管理要求高：必须定义清晰的规则和管理制度
* 技能要求高：需要明确每个迭代的规模，不能过大也不能过小
* 测试就绪点分析困难：无法确认测试准备工作是否合适，对后续测试就绪点的启动不明
* 对整个项目组的人员要求非常高，要保证在规范的制度下高效工作

**一、测试用例：**

**测什么**？**怎么测？**

**定义：**是为了特定的目的而设计的一组测试输入、执行条件和预期的结果，以便测试是否满足某个特定需求。通过大量的测试用例来检验软件的运行效果，是指导测试工作进行的依据

**举例：**

1. 买手机
2. 新浪注册页面

**二、等价类划分法：**

定义：将所有用户可能输入的数据划分为若干份（子集），然后从每一个子集中选取少数具有代表性的数据作为测试用例

作用：常用的**黑盒测试**方法，只需要考虑程序的输入规格，将**不能穷举的测试过程**进行合理分类，保证测试用例的**完整性和代表性**

**1.有效等价类：**

符合《需求规格说明书》，输入合理的数据集合

**2.等价类思考步骤：**

1）先确定有效和无效等价类

2）有效等价类就是题目条件（两端的极／边界值必须判断，中间随意一个值也要判断）

3）无效等价类先划分与条件相反的情况，再找到特殊情况：

• 整数（在范围内的整数-99到99）取最大、最小、中间

• 小数（在范围内的整数-99到99）

• 符号（+-=/*，。、！@#￥%……&*（））

• 汉字

• 空格

• 不输入

**3.0—100的两位数加法：**



**注意**：不会两个框都输入错误的值，以便确定到底是哪个框出现错误

**4.QQ登录账号**

**要求：**

测试QQ账号，账号的要求是 6---10位正整数。

**有效的等价类：**长度在6—10位之间的整数

**无效等价类：**

1:长度小于6

2:长度大于10.

3：负数

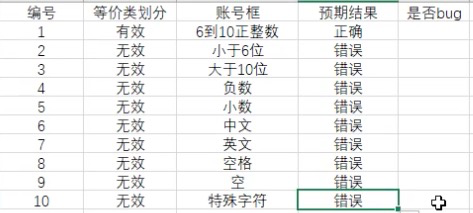
4：小数

5：英文字母

6：中文

7：空格

8：特殊字符



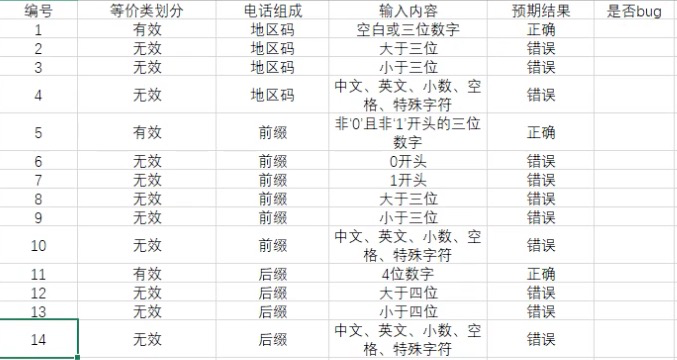
**5.电话号码**

某城市电话号码由三部分组成，分别是

**地区码：**空白或是3位数字

**前缀：**非‘0’且非‘1’开头的三位数字

**后缀：**4位数字



**6.注册登录界面**

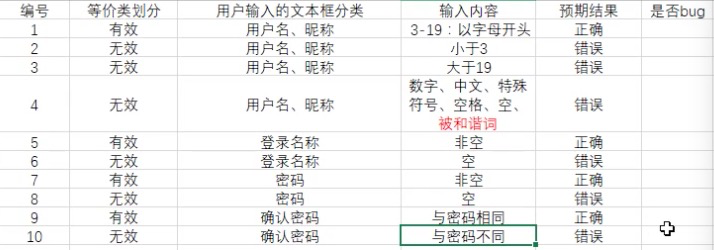
**要求：**

用户名（昵称）长度为 3-19：以字母开头

登录名称：非空

密码： 非空

确认密码： 值和密码相同



**等价类分类注意点：**

1.文本框要求输入的长度

2.输入的类型

3.组成规则

4.是否为空

5.是否重复

6.是否去除空格

7.区分大小写

**三、边界值**

* 定义：指对于输入等价类和输出等价类而言，稍高或低于其边界值的特定情况，大量的错误发生在**输入或输出范围的边界**上
* 有效数据和无效数据的边界的分界点往往是测试重点
* 一般情况下，需要对边界值（0和100）以及边界值两边的数（-1、1、99、101）分别进行测试

**边界值方法练习**

输入一个学生成绩n，判断是否及格（0到100整数）

思路：

* 画流程图Axure(产品经理用于制作流程图)
* 确定有效区域和无效区域
* 临界点：0、60、100
* 取值：-1、0、1、59、60、61、99、100、101
* 具体撰写测试用例



**方法小结：**

* 如果输入条件规定了值的范围，则应取刚到这个范围的边界值，以及刚刚超越这个范围边界的值
* 输入条件规定了值的个数，如姓名要求1—20个字符，则需要测试0、1、2、19、20、21
* 边界值与等价类的区别：边界值分析不是从某等价类中随便挑一个作为代表，而是这个等价类的每个边界都要作为测试条件

**常见边界值**

* 文本框接收字符个数，比如用户名长度、密码长度等
* 报表的第一行和最后一行
* 数值元素的第1个和最后1个
* 循环的第1次、2次和倒数第1次、2次

**四、因果图法**

**概念**

因（ci）：输入条件

果(ei)：输出条件、输出结果

各节点表示状态，**0**表示状态**不出现**，**1**表示状态**出现**

适用于**输入条件之间**有相互**制约及组合**、**输出条件对输入条件**相互**依赖**的情况

**因果图中的符号**

1. 恒等：若c1=1，则e1=1；若c1=0，则e1=0（取钱、打印凭条）
2. 非（～）：若c1=1，则e1=0；若c1=0，则e1=1（搜索联系人，如果有就不提示错误，如果没有就提示错误）
3. 或（v）：若c1=1或c2=1或c3=1，则e1=1；若c1=0或c2=0或c3=0，则e1=0（聪明、努力、听话是好孩子）
4. 与（^）：若c1=1且c2=1，则e1=1；若c1=0且c2=0，则e1=0（自称乡村教师且为淘宝前老板的人是马云）

**约束条件（非重点）**

1. 互斥：最多有一个可能成立
2. 包含：至少有一个必须成立
3. 屏蔽：a成立时b不成立；a不成立时b的值不一定（如元旦不上班，不是元旦则不一定上班）
4. 唯一：三个条件中有且只有一个成立
5. 要求：一个出现另一个一定出现（如银行取钱）

**因果图法基本步骤**

1. 找出所有的原因，即输入条件或输入条件的等价类
2. 找出所有的结果，即输出条件
3. 明确所有输入条件之间的制约关系及组合关系
4. 明确所有输出条件之间的制约关系和组合关系
5. 找出什么样的输入条件组成会产生哪种输出结果
6. 把因果图转换成判定表／决策表
7. 为判定表的每一列表示的情况设计测试用例

**五、判定表法**

**定义：**根据因果图来制作判定表（因果图可以不画）

**组成**

1. 条件桩：问题的所有条件
2. 动作桩：问题的所有输出
3. 条件项：针对条件桩的取值
4. 动作项：条件项的各种取值情况下的输出结果

**书写步骤**

1. 列出所有条件和动作桩
2. 填写条件和动作桩中的项目
3. 简化判定表

**六、场景法**

模拟用户操作软件时的场景，主要用来测试业务流程：分为基本流（正确流程）和备选流（错误流程）

注意：需要覆盖系统用例中的**主成功场景和扩展场景，还需要补充异常情况**

在冒烟测试中主要采用场景法来测试

**七、流程分析法**

适用于有**先后顺序**的测试：常用于业务流程、安装流程，从白盒测试设计方法中的路径覆盖分析法借鉴而来

* 在白盒测试中，路径就是指函数代码的某个**分支组合**，路径覆盖法需要构造足够的用例覆盖函数的所有代码路径。
* 在黑盒测试中，若将软件系统的某个流程看成路径的话，则可以针对该路径使用路径分析的方法设计测试用例

每个流程就是一条测试用例，只是测试整体流程是否正确，具体的细节和功能还需要使用等价类、边界值等方法来完善

**八、错误推测法**

指利用**直觉和经验**猜测出错的可能类型，有针对性列举程序中所有可能的错误和容易发生错误的情况

以一个攻击者的态度对待程序，才能够设计出比较完善的测试用例

**正交表和软件缺陷**

**一、正交表**

根据正交性从全面试验中挑选出有代表性的点进行测试，点具备**均匀分散、整齐可比**的特点，是一种高效率、快速、经济的方法

一般正交表记为:

1. n是表的行数，也就是需要测试组合的次数
2. K是表的列数，表示控件的个数（因素的个数，或因子个数）
3. m是每个控件包含的取值个数（各因素的水平数，即各因素的状态数）

如： 

* 有4个控件
* 每个控件有3个取值
* 9为需要测试的组合个数
* 叫4因素3水平

**查找正交表：**

<http://support.sas.com/techsup/technote/ts723_Designs.txt>

**正交表使用方法**

1、根据所测程序中控件的个数（因素）以及每个控件的取值个数（水平），选取一个合适的正交排列表

2、把控件及其取值列举出来，并对其进行编号

3、把控件及其取值映射到正交排列表中

* 把正交排列表中的ABCD（因子）分别替换成4个控件
* 把每列中的1,2,3（状态）分别换成这个控件的3个取值（水平），排列顺序要按照表中给出的顺序

4、根据映射好的正交排列表编写测试用例

**案例：字符属性设置程序**



**混合正交表工具**

用于因素和水平不对应的情况下

* 制作取值表（不需要编号，列出数据即可）
* 复制表格中的数据放在一个新建的txt文本文档中，保存到allpairs文件夹中（例如：文件命名为test.txt）
* 把文本文档（test.txt）放在allpairs文件夹中
* Win+r在输入cmd进入控制台界面
* 使用控制台代码进入allpairs文件夹（cd+目录名）
* 在控制台中输入allpairs.exe test.txt>test2.txt（test2.txt是自己起的名字，用于存放生成的组合用例，可以自动生成）

**二、测试用例方法的选择**

* 如果测试**功能和流程**，使用场景法（冒烟测试）
* 需要**输入数据**的地方，要使用等价类划分法，注意配合边界值法来做详细测试
* 程序的功能说明中含有**输入条件的组合**情况，使用因果图和判定表法
* 对于**参数配置类**的软件，需要考虑参数之间的组合情况，要使用正交表来科学选择测试用例
* 如果未达到覆盖标准，就要增加测试用例
* 用错误推断法—依靠经验追加测试用例

**三、软件缺陷**

**定义：**是指产品中存在的问题，最终表现为用户所需要的**功能没有完全实现，没有满足用户的需求**

**哪些情况属于缺陷？**

1. 软件未达到需求规格说明书表明的功能
2. 软件出现了需求规格说明书指明不会出现的错误
3. 软件的功能超出了需求规格说明书指明的范围
4. 软件未达到需求规格说明书虽未指明而应该达到的目标
5. 软件测试人员认为软件难以理解、不易使用、运行速度慢、或者最终用户认为不好

**缺陷的表现形式**

1. 功能、特性没有实现或部分实现
2. 设计不合理，功能特性不明确，逻辑不清楚或存在矛盾
3. 产品实际结果和所期望的结果不一致
4. 没有达到需求规格说明书所规定的性能指标等
5. 运行出错，包括运行中断、系统崩溃、界面混乱等
6. 数据不正确、精度不够、不完整或格式不统一
7. 用户不能接受的其他问题，如存取时间过长、界面不美观
8. 硬件或系统软件上存在的其他问题

**软件缺陷的状态**

|  |  |  |
| --- | --- | --- |
| **编号** | **缺陷状态** | **描述** |
| 1 | 提交（Submited） | 已提交的缺陷 |
| 2 | 打开（Open） | 确认“提交的缺陷”，等待处理 |
| 3 | 拒绝（Rejected） | 拒绝“提交的缺陷”，不需要修复或不是缺陷、重复缺陷、无法重现 |
| 4 | 修复（Resolved） | 缺陷被修复 |
| 5 | 关闭（Closed） | 回归测试后确认修复的缺陷，将其关闭 |
| 6 | 推迟（Later） | 可在以后解决，但要确定修复日期或版本 |

**软件缺陷的信息**

|  |  |  |
| --- | --- | --- |
| **编号** | **属性名称** | **描述** |
| 1 | 缺陷ID | 唯一的缺陷ID，可以根据该ID追踪缺陷 |
| 2 | 缺陷状态 | 缺陷状态指缺陷通过一个跟踪修复过程的进展情况 |
| 3 | **缺陷标题** | 描述缺陷的标题 |
| 4 | **缺陷的严重程度** | 对软件产品的影响程度，分致命、较严重、严重、一般、低 |
| 5 | **缺陷的优先级** | 缺陷修复的先后顺序，即哪些缺陷优先修正，哪些稍后修正 |
| 6 | **缺陷所属模块** | 缺陷所属的项目和模块，要能较精确的定位至模块 |
| 7 | 缺陷记录者 | 提交缺陷的人员姓名 |
| 8 | 缺陷提交时间 | 缺陷提交的时间 |
| 9 | 缺陷处理人 | 处理缺陷的处理人 |
| 10 | 处理结果描述 | 对处理结果的描述，描述处理情况和代码修改说明 |
| 11 | 缺陷处理时间 | 缺陷处理的时间 |
| 12 | 缺陷验证人 | 对被处理缺陷验证的验证人（回测者） |
| 13 | 验证结果描述 | 对验证结果的描述（通过、不通过） |
| 14 | **缺陷详细描述** | 缺陷的重现步骤 |
| 15 | 缺陷环境说明 | 对测试环境的描述 |
| 16 | 必要的附件 | 如涉及到附件的或错误现象的图片等。 |

**软件缺陷严重程度**

|  |  |
| --- | --- |
| **严重等级** | **描述** |
| 5-Critical | 系统瘫痪、异常退出、死循环、严重的计算错误等 |
| 4-VeryHig | 频繁的死机，系统大部分功能不可用 |
| 3-High | a.功能点没有实现，或不符合用户需求 b.数据丢失 |
| 2-Medium | a.影响一个相对独立的功能 b.仅仅在特定条件上发生 c.与产品需求定义不一致　d.断断续续的出现问题 |
| 1-Low | 表面性错误（如错别字） |

**软件测试的优先级**

|  |  |
| --- | --- |
| **优先级别** | **描述** |
| 5-Urgent | 最高优先级。在这个错误影响下，系统几乎不可用 |
| 4-VeryHigh | 高优先级。错误对这套系统的能力产生严重的影响 |
| 3-High | 中优先级。如果这个错误存在与系统中，会制约开发和测试的活动的进行，如果先前没有修复它，那么需要在发布前修复它 |
| 2-Medium | 低优先级。不会延迟发布，但是会在以后修正这个错误 |
| 1-Low | 最低优先级。时间和资源允许时修正 |

1. **严重程度高不一定优先级高，看用户需求和软件设计目标**

**软件缺陷分类**

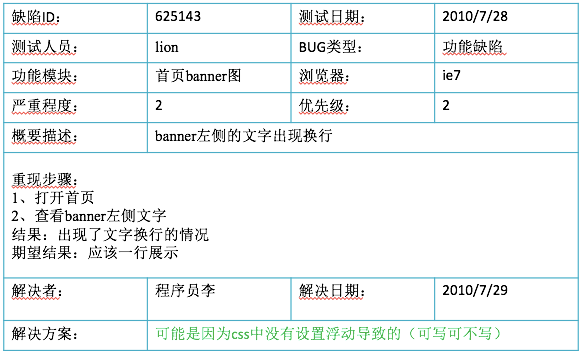
|  |  |  |
| --- | --- | --- |
| **缺陷类型** | **内容说明** | **备注** |
| 系统缺陷 | 1.由于程序所引起的死机，异常退出 2.程序死循环 3.程序错误，不能执行正常工作或重 要功能，使系统崩溃或资源不足 | 不能执行正常工作或重要功能，使系统崩溃或资源不足 |
| 数据缺陷 | 1.数据计算错误 2.数据约束错误 3.数据输入、输出错误 | 严重地影响系统要求或基本功能的实现，且没有办法更正（重新安装或重新启不属更正方法） |
| 数据库缺陷 | 1.数据库发生死锁 2.数据库的表、缺省值未加约束条件 3.数据库连接错误 4.数据库中的表有过多的空字段 |  |
| 接口缺陷 | 1.数据通信错误 2.程序接口错误 |  |
| 功能缺陷 | 1.功能无法实现 2.功能实现错误 | 严重地影响系统要求或基本功能的实现，但有合理的办法更正（重新安装或重新启动不属更正方法） |
| 安全性缺陷 | 1.用户权限无法实现 2.超时限制错误 3.访问控制错误 4.加密错误 |  |
| 兼容性缺陷 | 与需求规定配置兼容性不符合 |  |
| 性能缺陷 | 1.未达到预期的性能目标 2.性能测试中出错，导致无法继续进行测试 |  |
| 界面缺陷 | 1.操作界面错误 2.打印内容、格式错误 3.删除操作未给出提示 4.长时间操作未给出提示 5.界面不规范 | 操作者不方便或遇到麻烦，但不影响执行工作功能的实现 |
| 建议 | 1.功能建议 2.操作建议 | 建议性的改进要求 |

**报告缺陷注意事项**

1. 尽量保证缺陷可以重现
2. 简洁、准确、完整
3. 一个缺陷一个报告

**书写规范**

1. 标题简洁、提供缺陷的本质信息即可
2. 复现的步骤要详细，用数字编号
3. 实际结果要描述清楚复现后的结果
4. 列出期望结果
5. 提供附件
6. 提供严重性属性和其他公司需要填写的属性



**缺陷密度**

每千行代码的缺陷数（ 个/KLOC ）

缺陷密度=缺陷数\*1000/代码行数

**常用寻找缺陷的方法**

* UI用户界面：色彩搭配、功能结构布局、图片
* 页面大小
* 字体
* 窗体大小
* 界面文字（语病、简繁体）
* 容错处理（功能缺陷，无引导提示）
* 数据转换（增删改查）
* 性能缺陷（内存占用高、页面打开时间长）

**App功能测试分类：**

1. 业务逻辑正确性测试：产品文档、隐性需求—写成测试用例
2. 兼容性测试 系统版本（android、iOS） 分辨率（480\* 720、1920\*1080，主要关注图片展示情况） 网络情况（2G、3G、4G、wifi，图片兼容处理）
3. 异常测试 热启动应用：应用置于后台，应用置于前台的过程 网络切换&中断恢复 电话&信息中断恢复
4. 升级&安装卸载测试
5. 健壮性测试 手机资源消耗、流量消耗、崩溃恢复（闪退、报错）等测试

数据库

一、数据库基础

**RDBMS**：关系型数据库系统

核心：用表存储数据

表头有多个字段名，代表每一行数据的含义，表包含多行数据，每一行都有多个字段值

**SQL**：结构化查询语言（不区分大小写）

mysql：开源、免费、使用范围广、支持多平台

**Linux Centos(服务端使用)**

打开Terminal命令行

查看mysql是否存在：ps ajx | grep mysql

**root用户下执行命令**

启动mysql服务：service mysqld start

停止：service mysqld stop

重启：service mysqld restart

**二、数据类型与约束**

**常用数据类型**

**整数**：int，有符号范围signed（-2147483648～2147483647），无符号范围unsigned（0～4294967295）

**极小值**：tinyint（0～255）

**小数**：decimal，如decimal（5，2），则小数占2位，整数占3位，四舍五入

**字符串**：varchar（0～65533），如varchar（3）表示最多存3个字符，一个中文或一个字母都占一个字符

**日期时间**：datetime，格式为2020-03-27 12:49:23

\*查看任何一个数据类型的使用

1. 打开mysql命令行客户端
2. 输入help tinyint(数据类型名称)

**约束**

主键（primary key)：，一条记录的唯一标识，必须存在值且唯一，自动递增

非空(non null)：此字段不能填写空值

唯一(unique)：此字段不允许重复

默认值(default)：当不填写此值时会使用默认值

外键：维护两个表之间的关联关系

**数据增删改查**

# 1.增加

insert into 表名 values（……）

例：插入一个学生，设置所有字段的信息

Insert into stu values(0,'小王',23,'男')

# 2.修改

Update 表名 set 列1=值1,列2=值2

例：update stu set name='王昭君',age=20,sex='女'

# 3.删除

delete from 表名 where 条件

例：delete from stu where id=4

# 4.逻辑删除：对于重要的数据，防止删除后无法恢复，设置为逻辑删除

1）给表添加字段，代表数据是否删除，一般起名isdelete，0代表未删除，1代表删除，默认值为0

2）当要删除某条数据时，只需要设置这条数据的isdelete字段为1

3）以后在查询数据时，只查询出isdelete为0的数据

例：

1）删除id为1的学生

update students set isdelete=1 where id=1

2）查询未删除的数据

select \* from students where isdelete=0

**比较查询和逻辑查询**

=、>、<、!=（不等于）

and、or 、not

去重：select **distinct** cate from goods

**模糊查询**

**like**

**\_** 代表一个字符

% 代表任意字符、任意长度

select \* from students where name like '%孙%'

**范围查询 in**

select \* from students where hometown in ('北京',上海,'广州')

**between… and**：小值必须在前面

**空判断：**

**值为 null**（统计时该行不统计）

select \* from students where name is null

**字段值为空字符串**

select \* from students where name=‘ ‘

**排序查询 order by**

**当查询的字段名为order时，要在order加上反单引号``**

升序（默认）：asc

降序：desc

先按年龄降序排列，有相同的按分数升序排列：select \* from students order by age desc, score

**聚合函数：用于统计**

count(计数) 、avg(平均数) 、max(最大值) 、min(最小值) 、sum(求和)

select count(age) from students where score > 80

**分组 group by**

查询性别为男的总人数：select sex,count(\*) from from students group by sex having sex = '男'

having：先分组后过滤，只用在group by中

# 需求：返回每个部门中销售的最低工资

# 先对返回的结果进行排序，取部门最低工资；# 通常用于分组，选择最新／大／高数据情况

select count(\*),t.ename,min(t.sal) from (select \* from emp order by sal) t where t.job = '销售' group by t.deptno

**分页查询 limit**

多用于**限制商城、资讯页面**展示信息数量

索引从0开始，3为展示几个数据

查询前三个学生信息：select \* from students limit 0,3

**需求**：12条数据，每一页显示3条，总共4页

select \* from students limit 0,3

select \* from students limit 3,3

select \* from students limit 6,3

select \* from students limit 9,3

**ER模型**

E表示entry，实体：一个数据对象，描述具有相同特征的事物

R表示关系：一个或多个实体的关联关系，包括一对多、多对一、多对多

**避免数据冗余**

笛卡尔积=两个表数据的个数相乘

**三、连接查询**

**等值连接 where…… =**

需求：查询成绩表和课程表中的课程号、成绩、课程名、学号等信息

Select courses.name, scores.score,scores.studentNo, scores.courseNo from courses,scores where courses.courseNo = scores.courseNo

**内连接 inner join ……on**

需求：查询成绩表和课程表中的课程号、成绩、课程名、学号等信息

Select courses.name, scores.score,scores.studentNo, scores.courseNo from courses inner join scores on courses.courseNo = scores.courseNo

**左连接**

查询的结果为两个表匹配到的数据加左表特有的数据，对于右表中不存在的数据使用null填充

# 需求：查询所有学生的成绩、包括没有成绩的学生，需要显示课程名

select stu.name,sc.score,cs.name from students as stu left join scores as sc on stu.studentNo=sc.studentNo left join courses as cs on sc.courseNo=cs.courseNo

**右连接**

查询的结果为两个表匹配到的数据加右表特有的数据，对于左表中不存在的数据使用null填充

# 查询所有课程的成绩，包括没有成绩的课程

select \* from scores sc

right join courses cs on cs.courseNo = sc.courseNo

left join students stu on stu.studentNo = sc.studentNo

**自关联**

通常用于**设计省市等含有子类**的情况

# 查询河北省的所有城市

# 把一个表当作两个表使用，自己关联自己，需要给表设置别名

select \* from areas p

inner join areas c on c.pid=p.aid

where p.atitle='河北省'

**子查询**

在一个 select 语句中，嵌入了另外一个 select 语句， 那么被嵌入的 select 语句称之为子查询语句

**子查询分类**

* 标量子查询: 子查询返回的结果是一个数据(一行一列)
* 列子查询: 返回的结果是一列(一列多行)
* 行子查询: 返回的结果是一行(一行多列)
* 表级子查询: 返回的结果是多行多列

# 1.标量子查询

查询大于平均年龄的学生

select \* from students where age > (select avg(age) from students)

# 2.列级子查询

查询18岁的学生的成绩，要求显示成绩

select \* from scores where studentNo in (select studentNo from students where age=18)

# 3.行级子查询

查询男生中年龄最大的学生信息

select \* from students where (sex,age) = (select sex,age from students where sex='男' order by age desc limit 1)

# 4.表级子查询（查询结果要起别名）

查询数据库和系统测试的课程成绩

select \* from scores s

inner join (select \* from courses where name in ('数据库','系统测试')) c on s.courseNo = c.courseNo

**关键字**

* in 范围
  1. 格式: 主查询 where 条件 in (列子查询)
* any | some 任意一个（用于结果比较的情况中）
  1. 格式: 主查询 where 列 = any (列子查询)
  2. 在条件查询的结果中匹配任意一个即可,等价于 in
* all

格式: 主查询 where 列 = all(列子查询) : 等于里面所有

格式: 主查询 where 列 <>all(列子查询) : 不等于其中所有

**查询演练**

1.查询所有电脑产品的平均价格，并且保留两位小数

SELlect round(avg(price)) from goods

2.查询所有价格大于平均价格的商品，并且按照价格降序排列

select \* from goods where price >(select round(avg(price),2)  from goods) order by price desc

3.查询类型为“超级本”的商品价格

select price from goode where cates = '超级本'

4.查询价格大于或等于‘超级本’价格的商品，并且按照价格升序排列

# 此时子查询中返回一列值，而比价只需要一个最大值，因此要加上any

select \* from goods where price >= any(select price from goods where cates= '超级本') order by price asc

5.把商品分类数据插入到‘商品分类’表中

# 从一个表中查询到数据（cate）之后插入一个新表（goods\_cate）中，cate\_name为新表指定的字段名

insert into goods\_cate(cate\_name) select distinct cate from goods

6.创建“商品品牌”表

# 创建表与select语句结合，查询得到的结果（brand\_name)要与表中创建的字段名称一致才能插入

create table goods\_brand(

id int unsigned primary key auto\_increment,

brand\_name varchar(20)

) select distinct brand\_name from goods

7.备份表

# create table与select结合使用

create table goods\_bank select \* from goods

8.按照goods\_cate表更新goods表中字段名称

# 使用连接，update与inner join结合使用

update goods g inner join goods\_cate c on g.cate = c.cate\_name set g.cate = c.cate\_id

9.查询当前表中的商品名称、品牌、类型

select g.name,b.brand\_name,c.ctae\_name from goods g

inner join goods\_brand b on g.brand = b.brand\_id

inner join goods\_cate c on g.cate g.cate\_id = c.cate\_id

**Mysql命令行客户端**

加上‘ ；’代表执行改行内容

连接：mysql -u root -p ;

查看所有库：show datebases ;

打开数据库(student)：use student ;

查看表：show tables ;

对name进行中文转换：set name 'gbk' ;

查询表中所有商品：select \* from goods ;

显示当前数据库的名称：select database<> ;

创建数据库student：create database student charset = 'utf8' ;

删除数据库：drop database student ;

**cmd命令行备份**

必须以管理员身份运行cmd命令行

进入：cd mysql安装地址 ;

备份文件ceshi ：mysqldump -u root -p ceshi > c:\ceshi\_back.sql ;(备份文件存放的地址)

恢复文件ceshi\_back内容到cesh中：mysql -u root -p ceshi < c:\ceshi\_back.sql ;(备份文件存放的地址)

查看表的结构：desc goods；

查询创建表的完整代码：show create table goods；

**字符串函数**

1.拼接字符串

# 往事是江西的男生，必须起别名

select comcat(name,'是',hometown,'的',sex,'生') as des from students

2.字符串长度

# 查看'abc'的长度

select length('abc')

3.

# 查询所有学生的姓氏（默认单姓）,1代表取name里面的第一个字，right同

select left(name,1) from students

substring(name,pos,len)返回字符串name的位置pos起len个字符

4.

# 去除字符串两端空格

select rtrim(ltrim(' bar '))

去左边：ltrim

去右边：rtrim

5.

# 大小写转换

小写：select lower('aNJHsd')

大写：select upper('aNJHsd')

**数学函数**

1.求四舍五入值，选择小数点后2位

Select round(avg(age),2) from students

2.幂函数,求2的2次幂

select pow(2,3)

3.随机数,在表中随机查找一个学生的所有信息

select \* from students order by rand() limit 1

**日期时间函数**

1.当前日期

select current\_time()

2.当前时间

select now()

3.日期格式化 date\_formate

%Y：完整年份（2018）

%y:简写年份（18）

%m:月

%d:日

%H：24进制的小时数

%h:12进制的小时数

%i:分钟

%s:秒

# 将-拼接的日期转换为使用空格连接

select date\_formart('2016-12-21','%Y %m %d')

**流程控制**

# 选取学生表中的姓名首字母，性别为男则返回先生，性别为女则返回女士

# 以sex作为变量

select left(name,1),sex，

case sex

when '男' then concat(left(name,1),'先生')

when '女' then concat(left(name,1),'女士')

else '保密'

end as result

from students

# 计算员工年薪，需要加上奖金，并且以年薪排序

# ifnull(comm,0)：判断comm字段的值是否为null，若是则返回0；多用于函数加减乘除计算

Select sal\*12 ysal,comm,sal\*12+ifnull(comm,0) s from emp order by s

**存储过程**

# 在cmd命令行中分行输入

# delimiter用于设置分隔符号，默认为分号

# 在‘sql语句‘部分编写的语句通常以分号结尾，在命令行分号+回车默认执行，因此定义／／为分隔符

delimiter //

create procedure my\_stu(存储过程名称)

begin

sql语句

end

//

delimiter ;

# 回到客户端，点击“函数—my\_stu”，会展示“sql语句”，选择查询，输入‘call my\_stu’即可调用查询my\_stu内容并展示sql语句查询结果

**视图**

可用于隐藏真实数据，多见于银行等保密行业

# 1.先创建视图v\_student，查询学生名、课程名、学生分数

# 其中返回的数据名称不能重复，要先自定义名称

create view v\_student as

select stu.name,sc.score,cs.name from students as stu left join scores as sc on stu.studentNo=sc.studentNo left join courses as cs on sc.courseNo=cs.courseNo

# 2.调用视图函数

select \* from v\_student

# 3.查看视图，会将所有视图都列出来

show tables

# 4.删除视图函数

drop view v\_student

**事务**

在命令行中依次输入

begin;(触发事务，可开始修改数据)

sql语句；

commit;（提交事务，即将缓存的数据变更维护到物理表中）

rollback;(回滚事务，放弃缓存中变更的数据)

**索引 (B—Tree)**

# 1.创建表时创建索引

# 创建表时，对于主键和Unique字段，自动添加索引

# 其中，最后一行key(name)表示对name添加索引

create table student\_index(

id int unsigned primary key auto\_increment,

name varchar(10) unique,

age int,

key(name)

)

# 2.对已创建的表添加索引

# 若是字符串，建议长度与定义的字段一致

create index 索引名称 on 表名（字段名称（长度））

例：create index name\_index on create\_index(name(10));

查看索引:show index from 表名

删除索引：drop index 索引名称 on 表名

**外键**

# 1.创建外键时，主表和从表的两个字段，类型要保持一致

# 从表中的字段内容要受到主表的约束，只能在主表的范围内添加值

alter table 从表名 add foreign key(从表字段) references 主表名(主表字段)

例：alter table goods add foreign key(cate\_id) references goods\_cate(cate\_id)

# 2.删除外键

1）先查找表中的外键名称

show create table goodS

2）删除表中外键名称

alter table goods drop foreign key 外键名称

**修改密码**

# 1.在客户端修改密码

# 1)设定新密码为123456

update user set password = password('123456')

# 2)刷新权限使新密码生效

flush privileges

# 2.忘记root密码

1）Centos中：配置文件位置为/data/server/mysql/my.cnf

Windows中：配置文件位置为C:\Program Files (x86)\MySQL\MySQL Server 5.1\my.ini

2）修改，找到mysqld，在它的下一行，添加skip-grant-tables

3）‘服务’中重启mysql命令行，免密码登录，在客户端修改密码（同1）

4）还原配置文件，把刚才添加的skip-grant-tables删除，‘服务’中重启命令行

随机存储就是内存，断电数据丢失，优点是读写数据的速度很快

外存储器即硬盘，优点是断电数据仍然保存，缺点是督读写数据速度慢

**操作系统**

是一组具有特殊功能的程序，扮演了用户和硬件之间桥梁的作用，能简化对硬件的操作

Python开发环境搭建包含两部分：

1. 编辑Python代码的编辑器
2. 运行Python代码的解释器

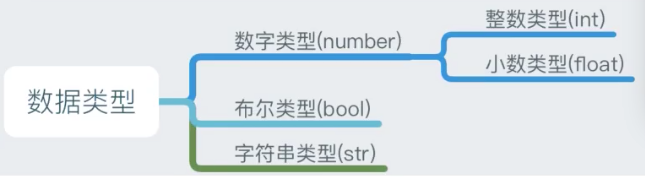
Python集成开发环境（IDE）

变量可以在程序运行过程中，临时存储程序所需要处理的数据

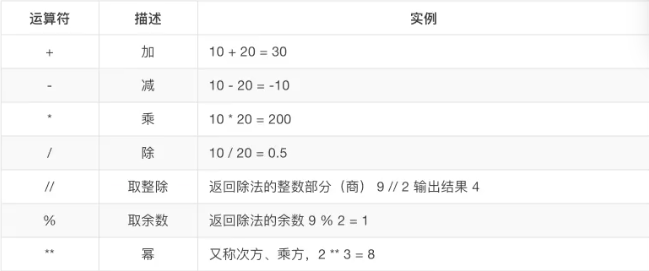
**变量命名规则：**

* 标识符由字母、下划线、数字组成，数字不能开头
* 标识符是区分大小写的
* 变量名一般用小写加下划线组成
* 不能和关键字以及已有名字相冲突
* 变量名一定要有意义，提升代码的可读性

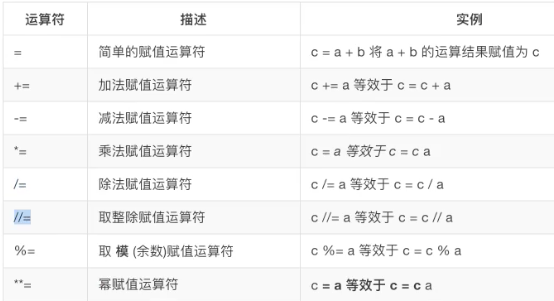
**变量中的数据类型**



**算术运算符**

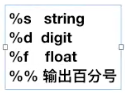


**复合赋值运算符**



小数float 整数int 字符串str %.2f表示小数点后只显示两位

**占位符**



**变量之间的运算规则：**

* 字符串和字符串之间只能进行加法运算
* 数字与数字之间都可运算
* 数字和字符串之间只能进行乘法运算

**if分支语句**：如果条件成立(True)则执行；不成立(Flase)则不执行



**登录案例**

input\_username = input('您的用户名是:')

input\_password = input('您的密码是:')

​

if input\_username == 'admin':

   if input\_password == 'ab123' :

       print('欢迎 %s 登录系统!' % input\_username)

   else:

       print('您的用户名或密码错误!')

else :

   print('登录失败')

**逻辑判断条件**

and和、or且、not非

**猜拳游戏**

import random

​

user = int(input('请输入：拳头（0），剪刀（1），布（2）:'))

computer = random.randint(0,2)

​

if (user == 0 and computer == 1) or \

      (user == 1 and computer == 2) or \

      (user == 2 and computer == 0) :

   print('您赢了')

elif user == computer :

   print('平局')

else :

   print('您输了')

**while语句**

如果条件永远满足，则意味着循环永远被执行，叫做死循环，这是无意义的

# 写出1——100的偶数

i = 1

while i <= 100 :

   if i % 2 == 0 :

       print(i)

   i += 1

​

print('end')

**函数的优点：**

* 减少代码冗余，减少维护量，实现功能的封装，降低学习成本，提升开发速度
* 调用函数时既传递参数问题，又传递关键字参数，应注意：**位置参数一定要在关键字参数前面**

**函数的返回值：return关键字**

* 当函数执行到return时，函数会马上停止执行
* 函数中可以出现多个return，但是有且只有一个return会执行
* return后面可以不跟值，return可以单独使用，等价于return None

若（a，20），在调用函数时传递的真实数据（20）叫实参，代表指定数据（a）的叫形参

按照从左到右的顺序传递叫做位置参数，按照形参名字传递叫做关键字参数

注意点：

1. continue即跳过本次循环，如果在while嵌套，内层循环执行了continue，对外层循环没有任何影响，break（退出循环）同理
2. print、return表示函数，执行一个功能；while（循环语句）、if（分支语句、def（定义函数）是一个语句
3. print只负责将内容输出到屏幕上，而return会将函数计算结果返回给函数的调用者
4. 看到冒号，下面的函数体需要缩进
5. 给函数形参某个位置设置了默认参数，那么该位置之后的所有参数都必须设置默认参数（a=10,b=20),b不能单独写

**全局变量（g\_val)：**在函数外部定义的变量，可以被python文件内的所有函数直接使用

**局部变量：**在函数内部定义的变量，只能在函数内部使用

变量要先定义再使用，遵循就近原则

**作用域：**表明变量能够被访问到的范围

函数文档相当于函数注释，解释函数的作用，快捷键是ctrl+q

函数应当满足单一职责原则：一个函数负责一个功能

**容器**：一个容器可以存储多个变量元素，能减少变量的定义

1. **序列式容器**：在存放时是**连续存放**的，包括字符串、列表、元组；支持索引
2. **非序列式容器：不连续存放**的，包括字典、集合
3. 序列式容器中，会给每一个元素赋予一个编号，该编号从0开始计算，支持根据下标存取元素
4. 遍历：不重复地访问容器中的每一个元素
5. 索引支持正数索引，也支持负数索引

正数：0——10+

负数：字符串倒数-1～-10+

my\_str = 'hello'

​

my\_str = "hel"

​

print(my\_str[0])# 0是索引，下标

​

for v in my\_str:

   print(v)

replace并不会替换原本的字符串，替换完毕之后会返回一个新的字符串

sentence = '我有一个梦想，是成为称霸世界的海贼王，我相信我会成功的！'

my\_sen = sentence.replace('我', '她')

print(my\_sen)

print('') # 用于换行

my\_sen = sentence.replace('我', '她',2)

print(my\_sen)

# 她有一个梦想，是成为称霸世界的海贼王，她相信她会成功的！

#

# 她有一个梦想，是成为称霸世界的海贼王，她相信我会成功的！

​

# 后面的数字是几，替换到第几个字

字符串一旦定义不允许修改，字符串容器中的元素都是字符类型的

user\_mail = '1995xrbmfk@gmail.com'

print(user\_mail[2])

# 返回的结果是'9'

​

# 如何获取@的位置 find方法默认从字符串（0）位置开始查找，也可指定位置，返回结果即查找结束位置

my\_mail = user\_mail.find('5')

print(my\_mail)

# 返回的结果是'3',即计数从0开始

​

# 切片语法，左闭右开，右边出现的最后一个结果不显示

​

# 冒号第一位默认为0，冒号两边的数字可写可不写

print(user\_mail[0:10])

# 返回的结果是'1995xrbmfk'

​

#获得容器的个数，即获取字符串长度

str\_length = len(user\_mail)

print(user\_mail[9:str\_length])

# 返回的结果是'k@gmail.com'

​

#步长：即根据数字选取其倍数位置的值，第二个冒号后的数字定义步长

print(user\_mail[0:17:4])

# 返回的结果是'1xfm.'

​

#步长可以为负数，相当于字符串逆序

print(user\_mail[::-1])

# 返回的结果是'moc.liamg@kfmbrx5991'

print(user\_mail[-2:-7:-1])

# 返回的结果是'oc.li'

**邮箱案例**

user\_mail = 'xbrhui@gmail.com'

position = user\_mail.find('@')

# 如果查找得到，则会返回子串第一次出现的位置

# 如果查找不到@，返回结果是-1

​

if position == -1:

   print('邮箱不合法')

else :

   user\_name = user\_mail[:position]

   houzhui = user\_mail[position+1:]

   print('用户名是：', user\_name)

   print('邮箱后缀是：', houzhui)

split函数：将原字符串分割为多个部分，以列表的形式返回结果

count函数：计算指定数值出现的次数

# -\*-coding:utf-8-\*-

from \_\_future\_\_ import print\_function

# 使print中的值不换行且中文输出

​

user\_name = '1999shxbe@qq.com'

count\_name = user\_name.count('@')

if count\_name > 1 :

   print('您的邮箱不合法!')

else :

   user\_name = user\_name.split('@')

   print('您的邮箱用户名是：',user\_name[0])

   print('您的邮箱后缀名是：',user\_name[1])

strip函数默认去除字符串两侧的空格，中间的不去

isalpha函数判断字符串所有元素是否都是字母

isdigit函数判断字符串所有元素是否都是数字

user\_name = str(input('请输入您的用户名：'))

​

user =user\_name.strip()

​

if user.isalpha():

   print('注册成功！')

else :

   print('注册失败！')

字符串中的元素不能修改，而且元素类型单一；列表中元素可以修改，且能自定义，支持索引、切片

# 空列表

my\_list = []

# 列表可存放多个多种类型的元素，为了运行方便，尽量使用同种类型元素

my\_list = [10,20,30]

my\_list = ['aaa', 'bbb', 'ccc']

my\_list = [1,2], [3,6] ,[7,8]

my\_list = ['Trump', 60,3.14, [1,2]]

**列表与遍历**

my\_list = [10,20,30,40]

print(my\_list[0:2])

print(my\_list[0],my\_list[2])

​

# 1.

my\_list = [[10,20,30],[40,50,60],['aa','bb','cc']]

i = 0

while i < len(my\_list):

   print(my\_list[i])

   i += 1

print('#'\*30)

# 2. for 循环一般用于容器中元素的遍历

for v in my\_list:

   print(v)

"""

返回结果是：

[10, 20, 30]

[40, 50, 60]

['aa', 'bb', 'cc']

"""

print("#"\*30)

# 1.寻找列表中的列表数据

my\_list = [[10,20,30],[40,50,60],['aa','bb','cc']]

i = 0

while i < len(my\_list):

   j = 0

   while j <len(my\_list[i]):

       print(my\_list[i][j])

       j += 1

   i += 1

​

print("#"\*30)

​

# 2.

for i in my\_list:

   for v in i :

       print(v)

"""

返回结果是：

10

20

30

40

50

60

aa

bb

cc

"""

**列表：**支持位置删除（尾部删除、指定位置删除）

**缺点：**根据关键字查找效率低；在指定位置插入和删除元素，会造成数据元素的移动

**优点：**列表根据位置、索引进行查找时效率较高；对于列表而言，**尾部**插入／删除数值效率更高，不需要移动原有元素位置

**列表元素排序**

# 创建一个包含了10个随机数的列表

import random

my\_list = []

i = 0

while i < 10:

   # 产生随机数，并将随机数插入到列表中

   random\_num = random.randint(1,1000)

   my\_list.append(random\_num)

   i += 1

print(my\_list)

​

# 对列表中的元素进行排序，sort，默认排序从小到大

my\_list.sort()

print(my\_list)

​

# 将sort函数的reverse默认值改为True，即可实现从大到小降序排列

my\_list.sort(reverse = True)

print(my\_list)

​

# 逆序，reverse函数

my\_list.reverse()

print(my\_list)

列表查找

my\_list = [10,20,30,300]

old\_num = 30

new\_num =200

# index用于根据值查询值的所在位置，如果查询值不存在则失败，会报错

if old\_num in my\_list:

   #查找到old\_num的位置

   pos = my\_list.index(old\_num)

   # 根据位置修改定义新的值

   my\_list[pos] = new\_num

print(my\_list)

​

# extend 将一个列表中的所有元素追加到当前列表的尾部

my\_list2 = ['aa','bb']

my\_list.extend(my\_list2)

print(my\_list)

# 返回结果是：[10, 20, 200, 300, 'aa', 'bb']

列表案例练习

# 需求：一个学校有3个空余办公室，将8个新老师随机分配到自己的工位上

​

import random

# 1.先定义学校和办公室

school = [ [],[] ,[] ]

​

# 3.定义一个列表型的函数

def c\_teacher():

# 2.定义列表保存老师

   teacher\_list = []

   index = 1

   while index <= 8 :

       # 创建老师的名字

       teacher\_name = '老师'+ str(index)

       #把老师装进盒子里

       teacher\_list.append(teacher\_name)

       index += 1

   return teacher\_list# 此为局部变量

​

# 此teacher\_list为外部变量，新的赋值，含义不同，修改上面一个teacher\_list不会影响下面一个teacher\_list

# 函数调用多次，每次返回一个新的值

teacher\_list = c\_teacher()

print(teacher\_list)

teacher\_list = c\_teacher()

print(teacher\_list)

​

# 4.分配老师

for teacher in teacher\_list:

   office\_num = random.randint(0,2)

   school[office\_num].append(teacher)

print(school,end='')

print()

​

# 5.查看每个办公室有哪些老师

for office in school:

   for teacher in office:

       print(teacher,end = ' ')

   print()

**元组**

python中的元组和列表类似，不同之处在于元组的元素不能修改，**元组使用小括号**，创建元组只需要在括号中添加元素，并使用逗号隔开即可

元组是序列式容器，支持遍历、查找、切片等操作

作用：从语法的层面来**限制数据的意外修改**

元组相对来说比列表更节省空间

my\_tuple = (10,20,30)

print(my\_tuple[0])

# 返回结果是：10

​

# 注意：元组中如果只有1个元素的话，需要在元素后面添加一个逗号

my\_tuple = (10,)

print(my\_tuple)

​

# 元组可以嵌套元组

my\_tuple = ((1,2),(3,4))

print(my\_tuple)

​

my\_tuple = (1,2,3)

for v in my\_tuple:

   print(v)

​

# 查询1在元组中的位置

pos = my\_tuple.index(1)

print(pos)

**字典**

**查找效率比较高**，但是占用内存高，以时间换空间

字典的每个键值key=>value对用冒号分割，多个键值对用逗号分割

键一般是唯一的，若重复，则最后的一个键值对会替换前面的，值不需要唯一

字典是非序列容器，不支持索引、切片操作、**字典的查询性能优于列表**

my\_dict = {'name':'虾皮','age':18,'sex':'女'}

​

print(my\_dict['name'])

# 返回结果是：虾皮

​

my\_dict['sex'] = '男'

print(my\_dict)

# 返回结果是：{'name': '虾皮', 'age': 18, 'sex': '男'}

​

# 使用 get 方法，如果查找的key不存在，返回结果是 None

print(my\_dict.get('age1'))

​

# 也可以自定义指定返回默认值

print(my\_dict.get('age1','输入key错误'))

# 返回结果是：输入key错误

​

# 如果key不存在，则添加键值对

my\_dict['score']='95'

print(my\_dict)

# 返回结果是：{'name': '虾皮', 'age': 18, 'sex': '男', 'score': '95'}

​

# 如果key存在，则修改键值对

my\_dict['name']='夏天'

print(my\_dict)

# 返回结果是：{'name': '夏天', 'age': 18, 'sex': '男'}

**字典的增删改查**

my\_dict = {'name':'Obama','age':20,'socre':50,'sex':'男'}

​

# 删除字典里的元素，也可删除列表中的元素

del my\_dict['age']

print(my\_dict)

​

# 清空字典

my\_dict.clear()

print(my\_dict)

​

#遍历字典

my\_dict = {'name':'Obama','age':20,'socre':50,'sex':'男'}

​

# keys 方法获取所有的键列表

key\_list = my\_dict.keys()

print(list(key\_list))

# 返回结果是：['sex', 'age', 'socre', 'name']

​

# keys 方法获取所有的值列表

value\_list = my\_dict.values()

print(list(value\_list))

# 返回结果是：['男', 20, 50, 'Obama']

​

# items 里面返回的值是元组形式，列表中的每组数据都是一条键值对

key\_value\_list = my\_dict.items()

print(list(key\_value\_list))

# 返回结果是：[('socre', 50), ('age', 20), ('name', 'Obama'), ('sex', '男')]

​

# 用for 循环遍历键值对

for key\_value in key\_value\_list :

   print(key\_value)

# 返回的结果是：

# ('name', 'Obama')

# ('age', 20)

# ('socre', 50)

# ('sex', '男')

​

# 使用 while循环遍历字典

my\_list = list(my\_dict.items())

i = 0

while i < len(my\_list):

   print('key:',my\_list[i][0],'value:',my\_list[i][1])

   i += 1

# 返回结果是：

# key: sex value: 男

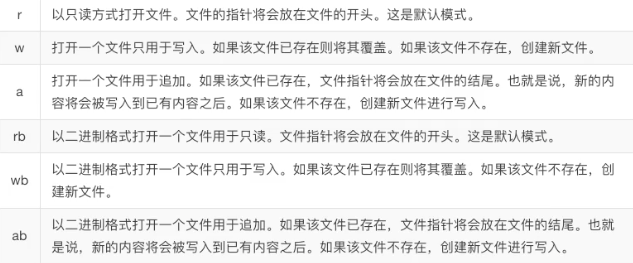
# key: socre value: 50

# key: name value: Obama

# key: age value: 20

**文件**：作用就是计算机存储数据

文件打开分为读、写、追加模式，其中又分为文本模式和二进制模式



1. 打开文件用的文本模式，会进行换行符转换
2. 打开文件用的是二进制模式的话，不会进行换行符转换
3. 文件本质上都是以二进制方式存储在磁盘上的

# 在文件中写入数据

f = open('a.txt','w')

my\_content = '你在桥上看风景，窗边的人在看桥上的你'

f.write(my\_content)

f.close()

​

# 读取文件中的数据

f = open('a.txt','r')

my\_content = f.read()

print(my\_content)

f.close()

​

# write会将已有内容覆盖掉

# 如果原文件不存在，则添加文件

def test01():

   f = open('hh.txt','w')

   f.write('只缘身在此山中')

   f.close()

test01()

​

​

def test02():

   fa = open('hh.txt','r')

   my\_content = fa.read()

   print(my\_content)

   fa.close()

test02()

​

# 在文件中追加数据，"\n"代表换行

def test03():

   fb = open('hh.txt','a')

   my\_content = fb.write('\n'+'一个孤独的人')

   fb.close()

test03()

**文件拷贝**

# 需求：将a.txt的文件拷贝到新文件中

​

old\_file\_name = input('请输入您想要拷贝的文件名：')

new\_file\_name = old\_file\_name + '.bk'

open(old\_file\_name,'rb')

open(new\_file\_name,'wb')

​

new\_file\_name.write()

​

new\_file\_name.close()

old\_file\_name.close()

**文件和目录的操作**

# 对文件的操作需要在 import os 中进行

import os

​

# 文件重命名，将文件'a.txt'的名称改为'xia.txt'

os.rename('a.txt','xia.txt')

​

# 文件删除 括号里应当写 文件完整路径+文件名

# 要注意路径问题，如果只写文件名，默认在当前目录下找文件，如果找不到会报错

os.remove('/Users/chenzhenyuan/Desktop/1.jpg')

​

# 创建目录

os.mkdir('/Users/chenzhenyuan/Desktop/abc.txt')

# 删除目录

os.rmdir('/Users/chenzhenyuan/Desktop/abc.txt')

​

# 获得指定目录下的文件列表

print(os.listdir())

​

# 获得和设置工作目录 cwd:current work directory

print(os.getcwd())

​

# 将默认的工作目录转换为我的桌面 括号里的为新路径

os.chdir('/Users/chenzhenyuan/Desktop')