**计算机与信息学院实验报告**

**实验6 异常处理**

1. 实验目的和要求
2. 理解异常的概念
3. 熟悉Python提供的内建异常类
4. 掌握try-except语句捕获和处理异常
5. 掌握raise和assert语句
6. 实验内容及原理
7. **错误信息提醒（Pyprog01.py，本题25分）**

**题目描述：**读入1个数A，然后输出20/A的值，保留两位小数；如果输入不正确，则输出相应的异常信息。如输入0，应捕获“ZeroDivisionError”异常，打印“The divisor cannot be 0: \*”;如输入字符，应捕获“NameError”异常类，并打印“The input is not a number: \*”。其中"\*"为系统异常类型对应的异常内容提示，冒号为英文。

**输入格式：**在一行中输入1个整数A。

**输出格式：**对每一组输入，在一行中输出20/A的值或相应的异常信息。

**输入输出样例1:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 8 | 2.50 |

**输入输出样例2:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| a | The input is not a number: name 'a' is not defined |

1. **求均值处理（Pyprog02.py，本题25分）**

**题目描述：**小明在帮老师处理数据，这些数据的第一行是n，代表有n行整数成绩需要统计。若输入的n不是整数则让用户一直输入，直到用户输入一个整数n。然后程序才让用户接着输入n个成绩，如果输入过程中均未出现异常，则输出“All OK”，然后计算平均值(保留2位小数)并输出。数据有错误，直接停止处理，并打印相关错误提示信息。不管输入过程中有无出现异常，最后都要输出“Process Completed”。注：该程序可以适当处理小错误，比如对于有些数据左右有多余的空格，可以将空格去除掉进行正常统计。

**输入格式：**第一行为n，代表接下来要输入的行数。若输入的n不是整数，则继续输入n。然后输入n行整数成绩。

**输出格式：** 若未出现异常，则第一行输出“All OK”，第二行输出“avg grade=平均值”(保留2位小数)。若输入有误，打印相关错误提示信息。不管输入过程中有无出现异常，最后都要输出“Process Completed”。

**输入输出样例1:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 2  3  5 | All OK  avg grade=4.00  Process Completed |

**输入输出样例2:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| bb  g  3  4.5 | invalid literal for int() with base 10: '4.5'  Process Completed |

1. 身份证号码断言**（Pyprog03.py，本题25分）**

**题目描述**：输入一个表示身份证号码的字符串，使用**assert**断言判断身份证号码是否正确。如果正确，则输出身份证号码。如果不正确，输出原因。判断身份证号码是否正确，可使用**assert**断言判断以下几个规则：

1. 身份证号码长度必须为18位。否则打印“身份证号码长度必须为18位”
2. 身份证号码必须由数字或者字母x（X）组成，否则打印“身份证号码必须由数字或X组成”。若含有x（X），则x（X）只能在最后一位。否则打印“x不在最后一位”
3. 月份和日期必须有效。若月份不是1-12月，打印“月份不正确”；日期不是1-31，打印“日期不正确”。（暂时认为每月都为31天）
4. 校验码不正确，打印“校验码不正确”。校验码的计算规则为，将身份证号码前17位数字分别乘一个对应的权重，然后将所有相乘的结果相加再对11取模，得到的余数通过查表得到相应的校验码，其中权重和表分别定义为：

FACTOR=(7,9,10,5,8,4,2,1,6,3,7,9,10,5,8,4,2)

CODE\_TABLE=('l','0','X','9','8','7','6','5','4','3','2')

**输入格式：**输入一个身份证号码

**输出格式：**判断输入的身份证号码，根据情况输出对应的内容

**输入输出样例1:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 110101202509070099 | 身份证号码110101202509070099是正确的 |

**输入输出样例2:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 11010120250907009xs | 身份证号码长度必须为18位 |

1. **求三角形面积（Pyprog04.py，本题25分）**

**题目描述：**请编写一段代码，根据输入的三角形三条边的整数边长，求出三角形的面积。输入的三条边的边长用空格分隔。程序应当处理一些异常：若输入的不是3条边，则抛出 IndexError异常；若输入的不是3个正整数，则抛出ValueError 异常；若输入的3条边无法构成三角形，则抛出ValueError。程序要能捕捉这些异常。三角形的面积=1/4sqrt[( a + b + c )( a + b - c )( a + c - b )( b + c - a )]

**输入格式：**一行输入一些数字，并用空格分隔

**输出格式：**根据输入的数字，若无异常则输出“三角形的面积是\*”，\*为面积（保留2为小数）；若输入的边数不是3，则输出“输入的边数是?不等于3”，？为你输入的边数；若输入的数据不是正整数，则输出“输入的数据#不是正整数”，其中#为你输入错的数据；若3个正整数不能构成三角形，则输出“a,b,c无法构成三角形”，a，b，c为输入的3条边

**输入输出样例1:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 3 4 5 | 三角形的面积是6.00 |

**输入输出样例2:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 2 4.5 | 输入的边数是2不等于3 |

**输入输出样例3:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 3 4.5 6 | 输入的数据4.5不是正整数 |

**输入输出样例4:**

|  |  |
| --- | --- |
| 输入 | 输出 |
| 1 3 5 | 输入的数据1,3,5无法构成三角形 |

1. 实验设备与环境

⑴硬件：微型计算机、局域网；⑵软件：Python

1. 方法与实验步骤

（1）题目1：

try:

A=eval(input())

a=20/A

except ZeroDivisionError as c:

print("The divisor cannot be 0:{}".format(c))

except NameError as e:

print("The input is not a number:{}".format(e))

else:

print("{:.2f}".format(a))

1. 题目2：

while(1):

try:

n=eval(input())

except:

print("",end="")

else:

break

num=0

q=n

try:

while(q):

q-=1

x=int(input())

num+=x

except TypeError as t:

print(t)

except ValueError as t:

print(t)

else:

print("ALL OK")

print("avg grade={:.2f}".format(num/n))

print("Process Completed")

1. 题目3：

FACTOR =(7,9,10,5,8,4,2,1,6,3,7,9,10,5,8,4,2)

CODE\_TABLE=('1','0','x','9','8','7','6','5','4','3','2')

num=input()

try:

assert len(num)==18,"身份证号码长度必须为18位"

for i in num[:-1]:

assert i.isdigit() or i.lower()=='x',"身份证号码必须有数字或X组成"

if num[-1].lower() !='x':

assert num[-1].isdigit(),"x不在最后一位"

m=int(num[10:12])

d=int(num[12:14])

assert 1<=m<=12,"月份不正确"

assert 1<=d<=31,"日期不正确"

t=sum(int(num[i])\*FACTOR[i] for i in range(17))%11

assert num[-1].lower()==CODE\_TABLE[t],"校验码不正确"

print("身份证号码{}是正确的".format(num))

except AssertionError as t:

print(t)

（4）题目4：

import math

def calculate\_triangle\_area(a, b, c):

s = (a + b + c) / 2

area = math.sqrt(s \* (s - a) \* (s - b) \* (s - c))

return area

def validate\_positive\_integer(side):

try:

if int(side) <= 0:

raise ValueError

except ValueError:

raise ValueError("输入的数据{}不是正整数".format(side))

def m():

try:

sides = input().split()

if len(sides) != 3:

raise IndexError("输入的边数是{}不等于3".format(len(sides)))

for side in sides:

validate\_positive\_integer(side)

a, b, c = map(int, sides)

if (a + b <= c) or (a + c <= b) or (b + c <= a):

raise ValueError("输入的数据{},{},{} 无法构成三角形".format(a, b, c))

area = calculate\_triangle\_area(a, b, c)

print("三角形的面积是{:.2f}".format(area))

except IndexError as e:

print("{}".format(e))

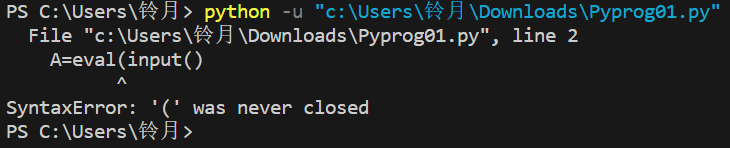
except ValueError as e:

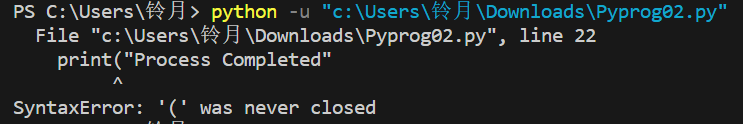
print("{}".format(e))

m()

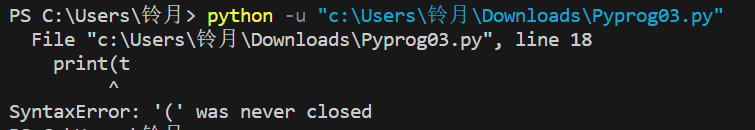
1. 实验调试过程

（1）题目1调试过程

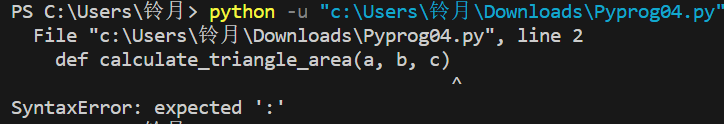
（2）题目2调试过程



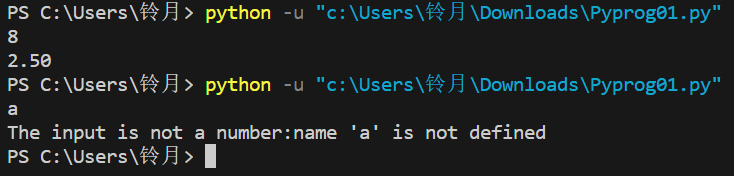
（3）题目3调试过程



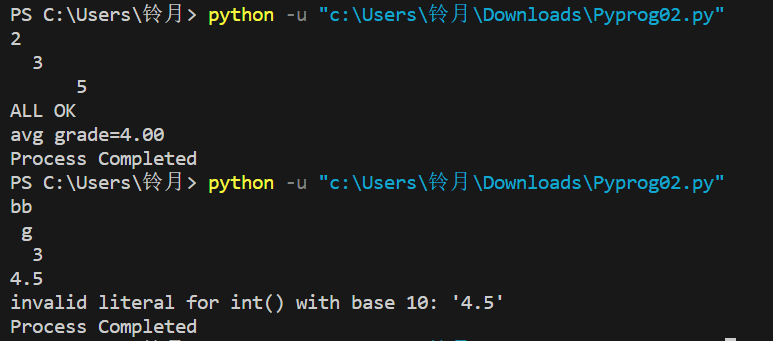
1. 题目4调试过程



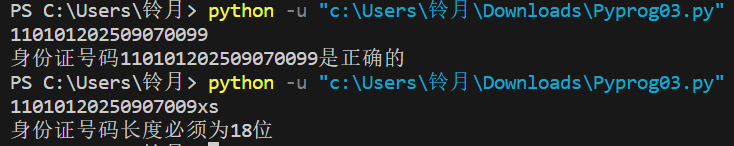
1. 实验结果分析
2. 题目1实验结果



1. 题目2实验结果



1. 题目3实验结果



1. 题目4实验结果



1. 建议、质疑和问题

写一些理解不了的问题，或是对实验有什么提议