5.1字符串格式化:就是把一个或多个值替换另一个字符串的某个标记

#字符串格式化,用%运算符，标记和提供参数值个数必须相同

#step1:定义一个模板

formateStr="Hello %s, Today %s, Are u there any activitiestoday?"

#step2:准备要替换标记的值

values=("Bill","Friday")

#step3:替换标记

print(formateStr%values)

#%f,%d

from math import pi

formatStr="PI is %.3f"

values1=(pi)

print(formatStr%values1)

formatStr="u will get %.2f%%"

values=(68.5688)

print(formatStr%values)

Hello Bill, Today Friday, Are u there any activitiestoday?

PI is 3.142

u will get 68.57%

## 5.2#用Template类格式化字符串

#标记$name $age

#格式化方法:substitute

from string import Template

template1 = Template("$lang is one of program language,$lang is power.")

print(template1.substitute(lang="Python"))

Python is one of program language,Python is power.

template2=Template("${s}stitute")

print(template2.substitute(s="sub"))

substitute

template3=Template("how much $pounds does $dollor$$ equal?")

print(template3.substitute(dollor=10,pounds="pounds"))

how much pounds does 10$ equal?

template4=Template("how much $pounds does $dollor$$ equal?")

data={}

data["dollor"]=10

data["pounds"]="pounds"

print(template4.substitute(data))

how much pounds does 10$ equal?

## 5.5使用format方法格式化字符串

#标记(格式化参数)：{...}

#如何进行格式化:template.format(...)

（1）#安顺序来制定格式化参数值

s1="Today is {}, the temperature is {} degrees."

print(s1.format("Monday",30))

Today is Monday, the temperature is 30 degrees.

（2）#使用命名格式化参数

s2="Today is {week},the temperature is {degree} degrees"

print(s2.format(week="Sunday",degree=20))

Today is Sunday,the temperature is 20 degrees

(3) #混合使用顺序格式化，命名格式化,顺序命名参数放在前面，后面所有为命名参数

s3="Today is {week},{}, the {} templature is {degree} degrees"

print(s3.format("hello",123,degree=28,week="monday"))

Today is monday,hello, the 123 templature is 28 degrees

(4) #使用序号格式化参数

s4="Today is {1}, the temperature is {0} degrees."

print(s4.format("Monday",30))

Today is 30, the temperature is Monday degrees.

(5) #获取列表中的指定值

fullname=["Bill","Gates"]

print(fullname)

s5="Mr.{name[1]}"

print(s5.format(name=fullname))

['Bill', 'Gates']

Mr.Gates

import math

s6="The {mod.\_\_name\_\_} module defines the values {mod.pi} for PI"

print(s6.format(mod = math))

The math module defines the values 3.141592653589793 for PI

5.6format的其他形式

（1）s1="原样输出：{first!s} 调用repr:{first!r} 输出Unicode编码:{first!a}"

print(s1.format(first="中国"))

原样输出：中国 调用repr:'中国' 输出Unicode编码:'\u4e2d\u56fd'

（2）#将一个整数按浮点数格式输出

s2="整数：{num} 浮点数:{num:f}"

print(s2.format(num=6))

整数：6 浮点数:6.000000

（3）#进制转换

s3="十进制：{num}, 二进制：{num:b}, 八进制:{num:o} 十六进制:{num:x}"

print(s3.format(num=12))

（4）#科学计数法

s4="科学计数法：{num:e}"

print(s4.format(num=12345))

#百分比

s5="百分比：{num:%}"

print(s5.format(num=0.23))

'''

a 将字符串按Unicode编码输出

b 将一个整数格式化为一个二进制数

c 将一个整数解释为ASCII码

d 将整数格式化为十进制的数

e/E 科学计数法

f 将一个整数格式化为浮点数，(nanheinf)转换为小写

g/G 会根据整数值的位数，在浮点数和科学计数法之间切换，在整数位超过6位时与e相同，否则同f

o 格式化为八进制

s 按原样格式化字符串

x/X 将一个整数格式化为十六进制数

% 将数格式化为百分比形式

'''

5.6#字段宽度、精度和千位分隔符

print("a:{num:12}".format(num=32))

#create table

print("{header1:10}{header2:6}".format(header1="姓名",header2="age"))

print("{cell11:10}{cell12:6}".format(cell11="Bill",cell12=23))

print("{cell21:10}{cell22:6}".format(cell21="Bill2",cell22=43))

#精度控制

from math import pi

print("floot number:{pi:.11f}".format(pi=pi))

print("floot number:{pi:15.4f}".format(pi=pi))

#截取字符串

print("{msg:.5}".format(msg="hello world"))

#千位分隔符

print("One googol is {:,}".format(10\*\*10))

5.7#符号、对齐和用0填充

from math import pi

print("{pi:012.3f}".format(pi=pi))

print("{pi:#12.3f}".format(pi=pi))

#<左对齐 ^居中 >右对齐

print("{pi:@^12.3f}".format(pi=pi))

print("{pi:#=12.3f}".format(pi=-pi))#在符号和数字之间填充

00000003.142 3.142

@@@3.142@@@@

-######3.142

5.8#字符串方法：center

#< hello >

print("<"+"hello".center(20)+">")

print("<{:^30}>".format("hello"))

#

print("<"+"hello".center(20,"\*")+">")

print("<{:\*^30}>".format("hello"))

5.9find方法

s="hello world"

print(s.find("world"))

print(s.find("abc"))

print(s.find("o"))

print(s.find("o",5)) #从5位开始查找

print(s.find("l",4,-1))#从4~-1位查找

5.10 #join,用于连接序列中的元素

list=["a","b","c","d","e"]

s="\*"

s1=s.join(list)

print(s.join(list))

print(s1)

#可以用于操作系统路径

##C:\abc\xyz

#/abc/xyz

dirs='','usr','local','nginx',''

print(dirs)

linuxPath='/'.join(dirs)

windowsPath='c:'+'\\'.join(dirs)

print(linuxPath)

print(windowsPath)

5.11 split方法

#split方法

s1="a b c d e f"

print(s1.split())

s2="a\*b\*c\*d\*e\*f"

print(s2.split("\*"))

#win和Linux系统路径转化

path="usr/local/nginx"

pathList=path.split("/")

print(pathList)

windowPath="c:"+"\\".join(pathList)

print(windowPath)

5.12 lower, upper和capwords

print("Hello world".lower())

print("Hello world".upper())

list=["Python","Ruby","C++","KOTLIN"]

if "Kotlin" in list:

print("find kotlin")

else:

print("kotlin not find")

for item in list:

if "Kotlin".lower() == item.lower():

print("Kotlin find")

break;

from string import capwords

s="i not only like Python, but also like Kotlin"

print(capwords(s))#每个独立单词首字母大写

5.13replace和strip方法

#replace,strip

s="abacdeafg"

print(s.replace("a","123"))

print(s.replace("abc","123"))

print(" geekori .com ".strip())

langList=["python","ruy","perl","c++"]

lang= " python "

if lang.strip() in langList:

print("find python")

else:

print("python not found")

s="\*\*\*\*\* $$\* hello \* world$ \*\* \*\*\*\*$$$\*\*\*"

print(s.strip("\*$ "))#截取掉\*/$/空格

5.14translate和maketrans方法

#translate 替换单个字符

s="I not only like python, but also like kotlin"

s1="helo"

table="s1".maketrans('ak','\*$')#两个参数长度要一一对应

print(table)

print(s.translate(table))

table1="saa".maketrans("ak","\*$"," ")#第三个参数删除

print(table1)

print(s.translate(table1))

'''

2.利用format生成一个塔

'''

num=input("请输入整数：")

num=int(num)

i=0

while i<num:

print("{:<{a}}{:\*<{b}}".format(" ","",a=num-i,b=2\*i+1))#format参数嵌套

i+=1