实验 3

选择结构：选择结构的使用

【实验目的】

（1）掌握条件语句中逻辑表达式的正确书写规则。

（2）掌握单分支、双分支及多分支条件语句的使用方法。

【相关知识】

选择结构是一种常用的基本结构，其特点是根据给定的选择条件来决定从不同操作中选择一种操作。常见的选择结构有以下几种。

（1）单分支选择结构。

```
if  表达式:
    语句块
```

（2）双分支选择结构 1。

```
if  表达式:
   语句块 1
else:
      语句块 2
```

（3）双分支选择结构 2。

```
表达式 1  if  条件  else  表达式 2
```

（4）多分支选择结构。

```
if  表达式 1:
  语句块 1
elif 表达式 2:
  语句块 2
  …
else:
  语句块 n
```

(5) if 语句的嵌套。

```
if 表达式 1:
   if 表达式 2:
       语句块 1
   else:
       语句块 2
else:
   if 表达式 2:
       语句块 3
   else:
       语句块 4
```

说明：

① if 语句中，表达式表示一个判断条件，可以是任何能够产生 true(1)或 false(0)的表达式或函数。表达式中一般包含关系运算符、成员运算符或逻辑运算符。

② Python 最具特色的功能就是使用缩进来表示语句块，不需要使用大括号。缩进的字符数是可变的，但是同一个语句块的语句必须包含相同的缩进字符数，如果缩进不一致会导致逻辑错误。

③ 在 Python 中，条件表达式中不允许使用赋值运算符“=”。

【实验范例】

例 3.1　某校三好学生的评定标准为：语文(c1)和数学(c2)两科的平均成绩大于 90 分，且每科成绩不低于 85 分，编写程序进行判断并输出判断结果。

分析：此例根据学生成绩判断该学生是否符合三好学生评定标准，判断结果只有两种情况，“是”或“不是”，所以这里只须采用 if 语句的双分支结构来表达即可。

程序代码如下：

```
c1=int(input("请输入语文成绩："))
c2=int(input("请输入数学成绩："))
if (c1+c2)/2>90 and c1>=85 and c2>=85:
    print("符合三好学生条件")
else:
    print("不符合条件")
```

例 3.2　某网吧根据上网时间来计算上网费用，计算规则如下，编程实现自动计费功能。

(1) 上网时间为 10 小时(含 10 小时)以内，基本网费 20 元；

(2) 上网时间为 10～50 小时(含 50 小时)，除基本网费外，超过 10 小时的部分每小时 1.5 元；

(3) 上网时间超过 50 小时，除基本网费外，超过 10 小时的部分每小时 1 元。

分析：此例需要根据已知条件进行上面 3 种情况的判断分析，因此采用 if 语句的多分支选择结构来表达比较简明、清晰。

程序代码如下：

```
t=int(input("请输入上网的小时数："))
if t<=10:
      cost =20
elif t>10 and t<=50 :
      cost=20+(t -10) * 1.5
elif t>50 :
      cost=20 +(t -10) * 1
print ("上网时间是%d 小时,网费=%.1f 元"%(t,cost))
```

【实验任务】

(1) 编写程序用于判断输入的年份是否为闰年,判断条件是能被 400 整除或者被 4 整除但不被 100 整除的年份是闰年。

程序代码如下：

```
year=eval(input("请输入任意年份："))
if (year%400==0) or (year%4==0 and year%100!=0) :
    print("%d 年是闰年!"%(year))
else:
    print("%d 年不是闰年!"%(year))
```

(2) 某商场购物打折范围如下：消费在 200 元以内不打折,200～500 元范围内打九折,超过 500 元打八折,请编写根据消费金额计算最终交费金额的程序代码。

程序代码如下：

```
x =int(input("请输入消费金额："))
if x<200:
    cost=x
elif x>=200 and x<=500:
    cost=x * 0.9
elif x>500:
    cost=x * 0.8
print ("最终交费=%.1f 元"%(cost))
```

【拓展训练】

相关知识：多分支 if 语句有多个 elif 为条件分支,当满足多个 elif 中的条件时,仅执行首次匹配成功的 elif 中的语句,虽然也满足后面 elif 中的条件,但都不被执行。

训练要求：编写一个模拟彩票兑奖的程序,当兑奖者输入一个 4 位数时,将此数字与计算机随机产生的 4 位数相比较,根据比较的结果来决定获奖等级。中奖规则为：4 位数字全部相同为一等奖;后 3 位数字相同为二等奖;后 2 位数字相同为三等奖;最后 1 位数字相同则为四等奖。

分析：此题需要根据彩票的比对情况进行 5 种情况的判断分析,因此采用 if 语句的多分支选择结构来表达。这里调用 random. randint(1000,9999)函数产生一个 4 位的随机数,通过 str()函数把随机数由数值转换为字符,与 input()函数输入的内容进行比对,每次比对的数字逐渐减少,对应的获奖等级也逐渐降低。

程序代码如下：

```
import random
#随机生成一个 4 位数作为中奖码
x=random.randint(1000,9999)
print("本期中奖号码是",x)
winnum=str(x)
ynum=input("请输入你的彩票 4 位号码：")
#如果 ynum 等于 winnum,则获一等奖
if ynum==winnum:
    print("恭喜!你中了一等奖")
  #如果后 3 位数字相同,则获二等奖
elif ynum[-3:]==winnum[-3:]:
    print("恭喜!你中了二等奖")
  #如果后 2 位数字相同,则获三等奖
elif ynum[-2:]==winnum[-2:]:
    print("恭喜!你中了三等奖")
  #如果最后 1 位数字相同,则获四等奖
elif ynum[-1:]==winnum[-1:]:
    print("恭喜!你中了四等奖")
else:
    print("谢谢参与!祝你下次好运!")
```

实验4

循环结构：循环结构的使用

【实验目的】

(1) 掌握循环的概念，能够用循环结构来解决算法问题。

(2) 熟练掌握实现遍历循环操作的 for 语句。

(3) 熟练掌握实现无限循环操作的 while 语句。

(4) 掌握用来辅助控制循环执行的 break 语句和 continue 语句。

【相关知识】

1. 遍历循环：for 语句

Python 的 for 循环结构有以下几种形式。

(1) 遍历序列：

```
for 循环变量 in 遍历序列：
  循环体语句块
```

执行过程：依次将遍历序列的每一个值传递给循环变量，每传递一个值时执行一次循环体语句块，直至传递完遍历序列的最后一个值，for 语句退出。

for 遍历循环可以遍历任何序列的项目，例如字符串(str)、列表(list)、元组(tuple)等。

例如：

```
for x in "ABCD":
    print("Hello!",x)
```

其中，for 语句循环的次数等于字符串“ABCD”中值的个数，遍历时，for 语句把字符串的值依次赋给了 x，执行循环体并打印输出。

输出结果如下：

```
Hello! A
Hello! B
Hello! C
Hello! D
```

（2）有限次循环：

```
for 循环变量 in range ( i, j[,k ]):
    循环体语句块
```

其中，*i* 是初始值（默认为'0'），*j* 是终止值（默认为'1'），*k* 是步长。

例如：

```
s=0
for i in range(1,100,2):
    s=s+i
print("s=1+3+5+7+..+99=",s)
```

输出结果如下：

```
s=1+3+5+7+..+99=2500
```

（3）遍历文件：

```
for eachrow in open("D:\\zzu.txt"):
    print(eachrow)
```

（4）遍历字典：

```
for x,y in {"姓名":'李明',"年龄":18}.items():
    print(x,y)
```

2. 无限循环：while 语句

while 语句也称为无限循环语句，常用于控制循环次数未知的循环结构，语法格式如下：

```
while 条件表达式:
    循环体语句块
```

while 循环中，当条件表达式为真时，就会重复执行循环体语句块，直到条件表达式为假才结束循环。

3. break 和 continue 语句，以及循环中的 else 子句

参见本实验的拓展训练部分。

【实验范例】

例 4.1　有一个分数序列 2/1，3/2，5/3，8/5，13/8，21/13，编程计算这个数列的前 20 项之和。

分析：观察分子与分母的变化规律，总结出序列规律如下：前一个分数的分子与分母的和作为下一个分数的分子，前一个分数的分子作为下一个分数的分母。采用 for 循环遍历，range(1,21)控制循环次数。

程序代码如下：

```
a,b,s,t=1,1,0,0
for n in range(1,21):
    t=a
    a=a+b
    b=t
    s+=a/b
print("2/1+3/2+5/3+8/5+13/8+…=%.2f" %s)
```

例 4.2 设计一个“过 7 游戏”的程序，这个游戏有 5 人以上参与，从任意一人从 1 开始报数：当遇到 7 的倍数(如 7,14,21,…)或含有数字 7(如 17,27,…)时，必须以敲桌子代替，报出 7 的倍数和含有数字 7 的人为输。

分析：7 的倍数，即除以 7 余数是 0(num % 7 == 0)；个位数含有 7 的数，即除以 10 余数是 7 (num % 10 == 7)；十位数含有 7 的数，即除以 10 后取整是 7 (num // 10 == 7)。

通过 while 循环分别对 2～99 的每一个数进行判断，如果某个数是 7 的倍数或含有数字 7，则通过 continue 语句跳过本轮循环，忽略此数，开始下一轮循环去判断下一个数。

程序代码如下：

```
n =0
while n <=99:
    n=n +1
    if (n %7==0) or (n %10==7) or (n // 10==7):
        print("敲桌子")
        continue
    else:
        print(n)
```

例 4.3 如果将 20 元钱换成零钱，要求只能换成 1 元、5 元、10 元面值的纸币，共有多少种兑换方法，分别是什么？

分析：20 元零钱中，每种纸币可能出现的次数：10 元纸币是 0～2 次，5 元纸币是 0～4 次，1 元纸币是 0～20 次。判断所有的组合中，币值总和正好是 20 元的情况有几种，即是此题所求。

```
n=0
for x in range(3):                    #10元面值的纸币张数为 0～2
    for y in range(5):                #5元面值的纸币张数为 0～4
        for z in range(21):           #1元面值的纸币张数为 0～20
            if x*10+y*5+z==20:
                print('10元=%d张  5元=%d张  1元=%d张  '%(x, y, z))
                n+=1
print('纸币兑换方法有%d种'%n)
```

本程序使用了三重嵌套的 for 循环结构，x、y、z 分别代表 10 元、5 元、1 元面值的纸币张数。for 循环把 range(3) 生成的列表[0,1,2]中的元素不断地赋值给 x，把 range(5) 生

成的列表[0,1,2,3,4]中的元素赋值给 y，把 range(21) 生成的列表[0,1,2,…,20]中的元素赋值给 z。三重嵌套遍历循环共执行 3×5×21＝315 次 if 判断，筛选出 9 种面值总和是 20 元的组合情况。

【实验任务】

(1) 参照例 4.3，把主教材中的百钱百鸡问题，用 for 遍历循环编写程序。

```
for cock in range(21):                  #公鸡个数为 0～20
    for hen in range(34):               #母鸡个数为 0～33
        chick=100-cock-hen              #小鸡个数
        if cock*5+hen*3+chick/3==100:
            print("公鸡=%d,母鸡=%d,雏鸡=%d"%(cock,hen,chick))
```

(2) 编程计算自然对数 e 的近似值，要求其误差小于 0.00001，公式为

$$e=1+1/1!+1/2!+1/3!+\cdots+1/n!+\cdots$$

```
from math import e
print("math 库里数学常数 e 的值是：",e)
x,i,n,t=0,0,1,1
while t>=0.00001:
    x=x+t
    i=i+1
    n=n*I
    t=1/n
print("利用级数公式所求的 e 近似值是：",x)
```

(3) 编程找出 1000 以内的所有完数。一个数如果恰好等于它的因子之和，这个数就称为“完数”。例如 6＝1＋2＋3，6 是完数。

```
for x in range(1,1000):
    sum=0
    for i in range(1,x):
      if x%i==0:
          sum=sum+i
      if sum==x:
          print("%d 是完数"%x)
```

(4)编写程序，使用 while 循环输出如下图形：

```
*
* *
* * *
* * * *
* * * * *
```

```
i=1
while i<=5:
    j=1
    while j<=i:
```

```
        print("*",end='')       #不换行连续打印 i 个 *
        j=j+1
    i=i+1
    print() #换行
```

输出倒三角形的程序代码为：

```
i=5
while i>=1:
    j=1
    while j<=i:
        print("*",end='')
        j=j+1
    i=i-1
    print()
```

【拓展训练】

相关知识：与其他编程语言不同的是，Python 的循环结构中会有 else 关键字，else 下面的语句在 while 循环或 for 循环正常结束时会被执行。但如果循环被 break 语句结束、执行了 return 语句或有其他异常处理出现时，则不会执行 else 语句。

格式 1：

```
while 表达式 :
      循环体
else:
      语句体
```

格式 2：

```
For 控制变量 in 可遍历的表达式:
    循环体
else:
    语句体
```

(1) 编写程序，输入任意一个整数，判断该整数是否是素数，并输出判断结果。

提示：首先来看一个循环结构中没有应用 else 子句的程序代码：

```
num=int(input("输入一个整数："))
i=2
while(i <num):    #此循环判断某一个 num 是否有因子
    if (num %i==0):
        break
    i=i+1
if (i==num) :
```

```
    print(num,"是素数")
else:
    print(num,"有因子",i,",不是素数")
```

这是一个查找素数的程序，其中应用了 break 来中止循环，跳出循环后再根据变量 i 值的大小来判断是否是素数。

如果在 while 循环中引入 else 关键字，程序后半部分可以由 if（i＝＝num）：print（num,"是素数"）变为 else：print(num,"是素数")，舍弃对 i 值大小的判断，直接由 while 循环所含的 else 子句下结论。程序代码修改如下：

```
num=int(input("输入一个整数："))
i=2
while(i <num):    #此循环判断某一个 num 是否有因子
    if (num %i==0):
        print(num,"有因子",i,",不是素数")
        break
    i=i+1
else:
    print(num,"是素数")
```

因为只有当前面的循环被 break 打断而不是正常结束循环时，else 子句才不被执行，此时已经隐含了循环是由 i 递增到终止值而导致的正常结束，不需要再判断 i 是否递增到 num 了。

使用 else 子句时，应注意结合 break、return 等关键词，否则会得出一些错误的结果。例如：

```
for str in 'abc':
    if str=='b':print("找到字母 b 了")
else:
   print ("没有字母 b")
```

程序运行结果如下：

```
找到字母 b 了
没有字母 b
```

这里因为 for 循环里没有 break 关键字，循环可以正常结束，else 子句也就正常被执行，因此出现两个结果。将该程序代码改造如下：

```
for str in 'abc':
    if str=='b':
        print("找到字母 b 了")
        break
else:
    print ("没有字母 b …")
```

(2) 编写程序,将一个正整数分解质因数并输出。例如,当输入 90 后,输出为 90=2*3*3*5。

分析:对 n 分解质因数,应先找到一个最小的质数 k,然后执行下述步骤。

(1) 如果 $k=n$,则分解质因数的过程已经结束,打印输出 k。

(2) 如果 $n \neq k$,但 n 能被 k 整除,则打印输出 k,并且把 n/k 的值作为新的 n 值,重复执行第(1)步。

(3) 如果 n 不能被 k 整除,则用 $k+1$ 作为 k 的值,重复执行第(1)步。

```
n =int(input("输入任意一个正整数:"))
print("%d ="%n,end="")
for k in range(2,n+1):
    while n!=k:
        if n %k==0:
            n=n/k
            print("%d * " %k,end="")
        else:
            break
print("%d"%n)
```