# Python概述

1. **填空题**
2. 对象
3. 可移植性
4. Python
5. import
6. from…import \*
7. **判断题**
8. √
9. ×
10. ×
11. √
12. √
13. **选择题**
14. C
15. D
16. C
17. **简答题**
18. Python主要的特点有代码简洁、语法优美、简单易学、开源、可移植、类库丰富、中文支持等。
19. Python中可以使用关键字import导入一个或多个模块，也可以使用from 模块名 import \* 导入模块中的全部内容。
20. 模块是最基础的代码组织方式，每个包含有组织的代码片段的.py文件都是一个模块；包以类似目录的机构组织模块文件或子包；库是一个抽象的概念，它是指具有相关功能模块的集合。
21. **编程题**

省略，代码详见教材。

# Python基础

1. **填空题**
2. 4
3. True False
4. type
5. 浮点
6. 1
7. **判断题**
8. ×
9. ×
10. ×
11. ×
12. √
13. **选择题**
14. A
15. A
16. D
17. B
18. C
19. **简答题**
20. 根据数据存储形式的不同，Python使用不同的数据类型存储不同类型的数据。数字类型提供了数值的存储，Python中的数字类型又包含整型、浮点型、复数类型和布尔类型。
21. 常量名使用大写的单个单词或由下画线连接的多个单词（如ORDER\_LIST\_LIMIT）；模块名、函数名使用小写的单个单词或由下画线连接的多个单词（如low\_with\_under）；类名使用大写字母开头的单个或多个单词（如Cat、CapWorld）。
22. Python运算符是一种特殊的符号，主要用于实现数值之间的运算。根据操作数数量的不同，运算符可分为单目运算符、双目运算符；根据功能的不同，运算符可分为算术运算符、赋值运算符、比较运算符、逻辑运算符和成员运算符。
23. **编程题**

答案：

radius = float(input("请输入园的半径："))

# 直径

diameter = 2 \* radius

# 面积

area = 3.14 \* radius \* radius

print('园的直径为：', diameter)

print('园的面积为：', area)

答案：

frequency = (29.5 - 4 \* 3) / 2.5

print("还需运送的次数为：",frequency)

# 流程控制

1. **填空题**
2. if
3. for ，while
4. True
5. for
6. continue
7. **判断题**
8. ×
9. ×
10. ×
11. ×
12. ×
13. **选择题**
14. C
15. C
16. C
17. C
18. B
19. **简答题**
20. break语句用于结束整个循环；continue的作用是用来结束本次循环，紧接着执行下一次的循环。
21. while语句一般用于实现条件循环；for语句一般用于实现遍历循环。
22. **编程题**
23. 编写程序，实现利用while循环输出100以内偶数的功能。

num = 0

while num <= 100:

if num % 2 == 0:

print(num)

num += 1

1. 编写程序，实现判断用户输入的是整数还是负数的功能。

num = int(input("请输入一个数："))

if num > 0:

print("输入的数是正数")

elif num < 0:

print("输入的数是负数")

else:

print("输入的书是零")

1. 编写程序，实现输出100以内质数的功能。

i = 2

for i in range(2, 100):

j = 2

for j in range(2, i):

if i % j == 0:

break

else:

print(i)

# 字符串

1. **填空题**
2. 单引号
3. ltrip
4. join，+
5. **判断题**
6. ×
7. √
8. √
9. ×
10. √
11. √
12. **选择题**
13. B
14. D
15. A
16. A
17. D
18. **简答题**
19. 字符串是由字母、符号或者数字组成的字符序列，Python支持使用单引号、双引号和三引号定义字符串。
20. Python有三种格式化字符串的方式：使用%格式化、使用format()方法格式化和使用f-string格式化字符串，其中使用%格式化格式为format % values；format()方法格式为str.format(values)；f-string只需要在字符串前加上引领的字符f或F即可。
21. Python提供了center()、ljust()、rjust()这3个方法来设置字符串的对齐方式，center()方法表示居中对齐显示；ljust()方法表示左对齐显示；rjust()方法表示右对齐显示。
22. **编程题**

答案：

s = 'AbcDeFGhIJ'

count = 0

for i in s:

for j in s.lower():

if i==j:

count+=1

print(count)

答案：

string = " Life is short. I use python"

if string.find('python'):

new\_string =string.replace('python','Python')

print(new\_string)

else:

print(string)

# 组合数据类型

1. **填空题**
2. list
3. 索引，切片
4. tuple
5. 键，值
6. keys()
7. 值的集合
8. **判断题**
9. ×
10. ×
11. ×
12. √
13. √
14. ×
15. **选择题**
16. A
17. A
18. C
19. B
20. C
21. **简答题**
22. Python中组合类型有列表、元组、字典和集合，这些数据类型都可以存储任意个元素。从访问元素来看，列表、元组和集合都可以通过索引获取数据，而字典类型数据没有索引；从存储数据来看，列表中的元素可以随意更改；存储在元组中的元素不可更改；存储在字典中键具有唯一性；存储在集合中的元素具有不重复性。
23. 字典中移除元素的方式有3种，分别为pop()、popitem()和clear()，pop()方法可以根据指定的键值删除字典中指定的元素；popitem()方法会随机删除字典中的元素；clear()方法会清空字典中的元素。
24. **编程题**

答案：

li\_num1 = [4, 5, 2, 7]

li\_num2 = [3, 6]

li\_num1.extend(li\_num2)

li\_num1.sort(reverse=True)

print(li\_num1)

答案：

tu\_num1 = ('p', 'y', 't', ['o', 'n'])

tu\_num1[len(tu\_num1)-1].append('h')

print(tu\_num1)

答案：

dict\_data = dict()

string= 'skdaskerkjsalkj'

for elem in string:

if elem not in dict\_data.keys():

dict\_data[elem] = 1

else:

dict\_data[elem] = dict\_data[elem] + 1

print(dict\_data)

答案：

li\_one = [1,2,1,2,3,5,4,3,5,7,4,7,8]

new\_li =[]

for i in li\_one:

if i not in new\_li:

new\_li.append(i)

print(new\_li)

# 函数

1. **填空题**
2. 函数
3. 函数名
4. 递归函数
5. global
6. 外
7. **判断题**
8. ×
9. ×
10. √
11. ×
12. √
13. **选择题**
14. C
15. D
16. C
17. B
18. C
19. **简答题**
20. 位置参数会将实参依次传递给形参；关键字参数是通过“形参=实参”形式将实参传递给形参；默认参数是在定义函数时，为形参赋值。
21. 混合传递参数规则为：优先按位置参数传递；然后按照关键字参数方式传递；之后按照默认参数传递；最后按照打包传递方式传递。
22. 根据作用域的不同，变量可以分为全局变量和局部变量。全局变量指的是可以在整个程序的范围内起作用；局部变量通常指在函数内定义的变量，该变量只能在函数体中使用。
23. **编程题**

答案：

def event\_num\_sum():

result = 0

counter = 1

while counter <= 100:

counter += 1

if counter % 2 == 1:

continue

result += counter

return result

print(event\_num\_sum())

答案：

def func(num):

if num == 2:

return 1

else:

return num \* func(num - 1)

result = func(20)

print(result)

答案：

def is\_palindrome():

num = input('请输入整数：\n')

palindrome\_num = num[::-1]

return num == palindrome\_num

print(is\_palindrome())

答案：

def triangle():

side\_length\_one = int(input("请输入第一个边长：\n"))

side\_length\_two = int(input("请输入第二个边长：\n"))

side\_length\_three = int(input("请输入第三个边长：\n"))

if (side\_length\_one + side\_length\_two > side\_length\_three and

side\_length\_one + side\_length\_three > side\_length\_two and

side\_length\_two + side\_length\_three > side\_length\_one):

return "能构成三角形"

else:

return "不能构成三角形"

print(triangle())

答案：

def lcm(x, y):

# 获取最大的数

if x > y:

greater = x

else:

greater = y

while True:

if greater % x == 0 and greater % y == 0:

lcm = greater

break

greater += 1

return lcm

# 获取用户输入

num1 = int(input("输入第一个数字: "))

num2 = int(input("输入第二个数字: "))

print(num1, "和", num2, "的最小公倍数为", lcm(num1, num2))

# 文件与数据格式化

1. **填空题**
2. close()
3. offset
4. 列表
5. 创建目录
6. tell()
7. **判断题**
8. √
9. ×
10. ×
11. √
12. ×
13. **选择题**
14. C
15. D
16. C
17. B
18. C
19. D
20. C
21. **简答题**
22. 根据数据的逻辑结构不同，计算机中的文件可分为文本文件和二进制文件，其中文本文件用于存储文本字符数据，而二进制文件用于存储图像、音频、视频等数据。
23. read()方法可以从指定文件中读取指定字节的数据；readline()方法可以从指定文件中读取一行数据；readlines()方法可以一次性读取文件中的所有数据，若读取成功返回一个列表，文件中的每一行对应列表中的一个元素。
24. **编程题**

答案：

def file\_read():

data\_li = []

with open('file.txt', 'r',encoding='utf8') as file:

for data in file.readlines():

if not data.startswith('#'):

data\_li.append(data)

return data\_li

print(file\_read())

答案：

passwordBook = {}

def add(password, value):

if password not in passwordBook.keys():

passwordBook[password] = value

save()

else:

print("该密码已存在")

def delete(password):

if password in passwordBook.keys():

del passwordBook[password]

print(passwordBook)

save()

else:

print("该密码不存在")

def modify(password, newValue):

if password in passwordBook.keys():

passwordBook[password] = newValue

print(passwordBook)

save()

else:

print("该密码不存在")

def find(value):

if value in passwordBook.values():

print("该网址存在")

else:

print("该网址不存在")

def save():

f = open("password.txt","w")

f.write(str(passwordBook))

f.close()

def printInfo():

print("密码簿的功能如下：")

print("1.添加")

print("2.删除")

print("3.修改")

print("4.查找")

i = 0

while i<5:

printInfo()

num = int(input("请输入要选择的功能："))

if num==1:

web = input("请输入存入的网址：")

password1 = input("请输入密码：")

add(password1, web)

print(passwordBook)

elif num==2:

password2 = input("请输入密码：")

delete(password2)

elif num==3:

password3 = input("请输入密码：")

value3 = input("请输入新的网址：")

modify(password3,value3)

elif num==4:

value4 = input("请输入要查找的网址：")

find(value4)

i+=1

答案：

def num\_sort():

file\_li = []

num\_li = []

with open('num.txt', 'r', encoding='utf8') as file:

data = file.readlines()

for i in data:

file\_li.append(i.split(' '))

for file\_elem in file\_li:

for elem in file\_elem:

if elem.find('\n'):

new\_elem = elem.replace('\n', '')

num\_li.append(int(new\_elem))

else:

num\_li.append(int(elem))

num\_li.sort()

print(num\_li)

num\_sort()

# 面向对象

1. **填空题**
2. class
3. 类方法，类属性
4. 双下画线
5. 父类或基类，子类或派生类
6. super()
7. **判断题**
8. ×
9. √
10. ×
11. √
12. ×
13. **选择题**
14. D
15. D
16. B
17. A
18. A
19. **简答题**
20. 类方法需要使用@classmethod进行标识，该方法可以访问类属性，无法访问实例属性，可以通过类实例和类进行调用。静态方法使用@staticmethod进行标识，该方法无法访问实例属性和类属性，起到类似于函数的作用，使用类或者类实例进行调用。实例方法直接在类中使用def进行定义，可以访问其实例属性和类属性，使用类实例进行调用。如果要修改实例属性的值，就直接使用实例方法；如果要修改类属性的值，就直接使用类方法；如果是辅助功能，比如打印菜单，这时可以考虑使用静态方法，可以在不创建对象的前提下使用。
21. 构造方法为\_\_init\_\_()作用是初始化对象的属性，析构方法为\_\_del\_\_()作为是释放类所占用的资源。
22. 封装是指隐藏类的实现细节，只提供访问类成员的公开接口；继承是指在一个现有类的基础上构建一个新的类；多态是指在不考虑对象类型的情况下使用对象。
23. **编程题**

答案：

class Circle:

def \_\_init\_\_(self,tup, radius, color):

self.center = tup

self.radius = radius

self.color = color

def perimeter(self):

return 3.14 \* 2 \* self.radius

def area(self):

return 3.14 \* self.radius \* self.radius

circle = Circle((0,0),5,"蓝色")

print(circle.perimeter())

print(circle.area())

答案：

class Course:

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

self.number = 1001

self.name = "语文"

self.teacher = "张老师"

self.\_\_location = "2号教学楼3层305室"

def show\_info(self):

return ("""

课程编号：%d

课程名称：%s

任课教师：%s

上课地点：%s

""" % (self.number, self.name, self.teacher, self.\_\_location))

course = Course()

print(course.show\_info())

# 异常

1. **填空题**
2. BaseException
3. NameError
4. Exception
5. AssertionError
6. **判断题**
7. ×
8. √
9. √
10. ×
11. ×
12. **选择题**
13. A
14. B
15. A
16. A
17. B
18. **简答题**
19. 在Python中，程序在执行的过程中产生的错误称为异常，比如列表索引越界、打开不存在的文件等。
20. Exception类常见的异常子类有NameError、FileNotFoundError、IndexError、AttributeError，其中NameError异常产生的原因是使用了未定义的变量；FileNotFoundError异常是打开不存在的文件；IndexError异常是访问了规定索引之外的数据；AttributeError异常产生的原因是使用对象访问不存在的属性引发的。
21. 抛出异常的方式有raise 异常类；raise 异常对象和raise，其中第一种是使用异常类名引发指定的异常；第二种是使用异常类的对象引发指定的异常；第三种是使用刚出现过的异常重新引发异常。
22. **编程题**

答案：

class NumericalError(Exception):

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

self.message = '请输入正确的数据'

class CircleArea:

def circle(self):

try:

radius = int(input("请输入圆的半径：\n"))

if radius <0:

raise NumericalError()

except NumericalError as e:

print(e.message)

else:

# 计算圆的面积

print(3.14\*radius\*\*2)

circle\_area = CircleArea()

circle\_area.circle()

答案：

class MessageError(Exception):

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

self.no\_message = '不能构成三角形'

class Triangle:

def composed\_triangle(self):

side\_length\_one = int(input("请输入第一个边长：\n"))

side\_length\_two = int(input("请输入第二个边长：\n"))

side\_length\_three = int(input("请输入第三个边长：\n"))

try:

if not (side\_length\_one + side\_length\_two > side\_length\_three and

side\_length\_one + side\_length\_three > side\_length\_two and

side\_length\_two + side\_length\_three > side\_length\_one):

raise MessageError()

except MessageError as e:

print(e.no\_message)

else:

print('可以构成')

triangle = Triangle()

triangle.composed\_triangle()

# Python计算生态与常用库

1. **填空题**
2. 网络爬虫
3. 数据分析
4. Numpy，Pandas，SciPy
5. 标准库，第三方库
6. init()
7. **判断题**
8. √
9. √
10. ×
11. ×
12. √
13. √
14. √
15. **选择题**
16. B
17. D
18. B
19. A
20. B
21. **简答题**
22. Python计算生态覆盖的领域包括网络爬虫、数据分析、文本处理、数据可视化、机器学习、图形用户界面等。
23. Python中的库分为标准库和第三方库，标准库可以在Python中直接使用，第三方库需要进行安装；模块本质上是一个包含Python代码片段的.py文件；将模块放入到一个文件夹中，并在该文件夹中创建\_\_init\_\_.py文件，就构建了一个Python包。
24. 若两个表示时间的变量进行计算，可以先将其转换为时间戳形式。因为时间戳是以浮点型显示，便于时间计算与转换。
25. **编程题**

答案：

import turtle

'''全局变量'''

amount = 10 #词频排列显示个数

words = [] #单词集合-x轴数据

wCounts = [] #单词频数集合-y轴数据

xPoint = -360

yPoint = -200

'''turtle start'''

#绘制从点(x1,y1)到(x2,y2)的线段

def drawLine(t,x1,y1,x2,y2):

t.penup()

t.goto(x1,y1)

t.pendown()

t.goto(x2,y2)

#在坐标(x,y)处写文字

def drawText(t,x,y,text,fontSize=10):

t.penup()

t.goto(x,y)

t.pendown()

t.write(text,font=('微软雅黑',fontSize,),align='center')

#绘制矩形

def drawRectangle(t,x,y,rWidth):

drawLine(t,x-rWidth,yPoint,x-rWidth,y)

drawLine(t,x-rWidth,y,x+rWidth,y)

drawLine(t,x+rWidth,y,x+rWidth,yPoint)

drawLine(t,x+rWidth,yPoint,x-rWidth,yPoint)

#绘制柱状图

def drawBarchart(t):

drawText(t,0,-yPoint-40,"词频统计结果",15) #绘制标题

drawRectangle(t,0,-yPoint,-xPoint) #绘制边框

rWidth = -xPoint/(2\*amount) #控制条形柱宽度（2rWidth）

xScale = -xPoint\*2/(amount+1) #x轴显示放大倍数-可根据amount进行调节

yScale = -yPoint/wCounts[0] #y轴显示放大倍数-可根据频数进行调节

for i in range(amount):

i=i+1 #右移以避免与原点重合

x=i\*xScale+xPoint

y=wCounts[i-1]\*yScale+yPoint

drawText(t,x,yPoint-20,words[i-1]) #打印单词

drawText(t,x,y+10,wCounts[i-1]) #打印频率

t.begin\_fill()

drawRectangle(t,x,y,rWidth)

t.end\_fill()

#初始化窗口

def init():

turtle.title('词频结果柱状图')

turtle.screensize(900,750,"#272727")

t=turtle.Turtle()

t.hideturtle() #隐藏箭头

t.width(1) #线条粗细

t.color("#EBEBD0","#006030")

#t.color("#006030","#FFF68C")

drawBarchart(t) #绘制

turtle.exitonclick()

'''data Processing'''

#对文本的每一行计算词频的函数

def processLine(line,wordamounts):

line = replacePunctuations(line) #用空格替换标点符号

words = line.split() #从每一行获取每个词

for word in words:

if word in wordamounts:

wordamounts[word] += 1

else:

wordamounts[word] = 1

#空格替换标点

def replacePunctuations(line):

for ch in line:

if ch in "~!@#$%^&\*()-\_+=<>?/,.:;{}[]|\'\"":

line = line.replace(ch,' ')

return line

#数据处理，得到文件中出现频率最高的单词及其频数

def dataProcess(filename):

infile=open(filename,'r',encoding='UTF-8')

wordamounts={} #建立用于记录词频的空字典

for line in infile:

processLine(line.lower(),wordamounts)

pairs = list(wordamounts.items()) #从字典中获取数据对

items = [[x,y]for (y,x) in pairs] #列表中的数据对交换位置，使频数成为元素的键值

items.sort() #列表排序。默认根据键值排序

#输出amount个词频结果

for i in range(len(items)-1,len(items)-amount-1,-1):

print(items[i][1]+"\t"+str(items[i][0]))

wCounts.append(items[i][0])

words.append(items[i][1])

infile.close()

def main():

#用户输入一个文件名

filename= input("enter a filename:").strip()

dataProcess(filename)

init()

#调用main()函数

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

main()

答案：

import wordcloud

from matplotlib.image import imread

font = 'AdobeHeitiStd-Regular.otf'

# 用于生成词云的字符串

with open('葫芦兄弟.txt','r',encoding='utf-8')as file:

string = str(file.read())

# 词云形状

mk = imread('葫芦娃.jpg')

# 创建词云对象

w = wordcloud.WordCloud(font\_path=font, mask=mk,

max\_words=500,background\_color='white')

# 加载文本

w.generate(string)

# 生成词云

w.to\_file('葫芦娃.png')