电子科技大学 2023-2024 学年第 2 学期
《Python 语言程序设计》课程考核情况说明

《Python 语言程序设计》课程考核说明：
（1）课程总成绩构成：平时作业（50%）+课程设计（50%）
（2）平时作业需提交材料：平时作业答题卡（不抄题，请按照附件答题卡格式提交答案）
（3）课程设计需提交材料：①程序功能简介；②程序的运行使用方法和步骤说明；③程序运行效果的截屏图；④程序源代码（添加上适当注释）；⑤运行程序所需数据等资源文件（例如试题题库、商品库、图片等，若不需要就不提供）
（4）课程考核材料（平时作业和课程设计）都需要同时提交纸质版和电子版：电子文档用于评审成绩时运行验证，纸质文档用于学校存档。
（5）电子版考核材料（平时作业答题卡和课程设计文档）请提交到电子邮箱：nchenlua@sina.com，邮件标题为：Python+姓名+学号
（6）平时作业答题卡文件名为：Python+姓名+学号+平时作业
（7）课程设计的相关资料【程序功能简介+使用说明文档+运行效果图+程序源代码+数据等资源文件（若不需要就不提供）】装在一个文件夹中提交，文件夹的名称：Python+姓名+学号+课程设计
（8）课程考核材料提交的截止时间：2024 年 6 月 15 日。（若有困难，请在截止时间前单独联系老师协商提交时间）
（9）纸质文档请提交到电子科技大学清水河校区西二门旁边的研究院大楼 3 楼 316#实验室陈峦老师处（实验室一直有研究生在里面学习，把平时作业答案和课程设计文档放在陈峦老师的办公桌上即可，请一定写上自己的姓名、学号、专业等个人信息）。

第一部分 《Python 语言程序设计》课程设计

说明：
（1）课程设计占课程总成绩的 50%。
（2）在以下五个备选题目中任选一题完成。
（3）课程设计需提交的材料：程序功能简介+使用方法说明文档+运行效果图+程序源代码（含注释）+数据等资源文件（例如试题题库、商品库、图片等，若不需要就不提供）。
（4）使用 Python 解释器安装时自带的基础库（例如 tkinter 模块等）来实现系统功能。（尽量不要使用其他第三方库等）

课程设计备选题目一：自动组卷评卷考试系统
内容：

用 Python 语言编程实现自动组卷评卷考试系统，软件主要功能包括：从题库中随机抽取试题自动组成试卷（满分 100 分）；实现考生考试答题操作界面；实现自动阅卷评分功能；等等。

要求：

（1）题型包括单项选择题、填空题、判断题等等。

（2）能提供组卷评卷考试服务功能。

（3）题库可以采用文本文件、CSV 文件或数据库等来实现。

（4）要求在源程序中标注必要的注释。

（5）要求对程序的使用和运行方法进行必要说明，并有 GUI 运行界面截屏图。

（6）课程设计要提交程序源代码及附属的测试题库文档等（便于阅卷测试）。

评价标准：

（1）源程序功能（占 50%）

（2）源代码质量（占 20%）

（3）GUI 界面美观友好（占 20%）

（4）GUI 运行界面截屏图、源程序注释和使用说明文档（占 10%）

课程设计备选题目二：绘图工具的实现

内容：

用 Python 语言编程实现一个绘图工具。该绘图工具具有绘制直线、曲线、圆、椭圆、圆弧、矩形、多边形、插入文本、显示图片、删除某个图形、设置图形颜色等功能。例如，具有类似于 Windows 附件中的画图工具等绘图软件的功能。

其他功能举例：

例 1：从数据文件中读出 n 维数据，能将其可视化显示成曲线或图形等。

例 2：拖动滑块，产生随机数据及曲线，源代码如下。

```
from tkinter import *
w=Tk()
w.geometry("200x80")
d=Tk()
draw=Canvas(d,width=200,height=500)
x=100
y=0
def show(text):
    global x,y
    draw.create_line(x,y,2*float(v.get()),y+3)
    x=2*float(v.get())
    if y>500:
        y=0
    else:
        y+=1
    print("v=",v.get())
def clear():
    global y
    for obj in draw.find_withtag("all"):
        draw.delete(obj)
```

```
    y=0
v=StringVar()
scl=Scale(w,from_=0,to=100,resolution=0.1,orient=HORIZONTAL,variable=v,command=show)
scl.set(50)
b=Button(w,text="clear",command=clear)
scl.pack()
b.pack()
draw.pack()
w.mainloop()
```

也可以把上面程序产生的随机数据保存在数据文件中。

要求：
（1）使用 Python 解释器安装时自带的基础库（例如 tkinter 或 turtle 模块等）来实现系统功能；（尽量不要使用其他第三方库等）
（2）能提供绘制图形的相关功能；
（3）人机界面美观友好；
（4）给出程序运行结果截屏图；
（5）要求在源程序中标注必要的注释；
（6）要求对程序的使用方法和运行方法进行必要说明。
评价标准：
（1）源程序功能（占 50%）
（2）源代码质量（占 20%）
（3）人机界面美观友好（占 10%）
（4）运行结果截屏图、源程序注释和程序的使用说明文档（占 20%）

课程设计备选题目三：记事本工具的实现
内容：

用 Python 语言编程实现一个记事本工具。该记事本工具具有文本插入、删除、替换更新、查找、复制、粘贴等编辑功能，也具有设置字体、字形、字号、颜色等文字格式功能，还具有新建、打开和保存等文本文件管理功能。例如，具有类似于 Windows 附件中的记事本工具等文本编辑软件的功能。

部分功能举例：
（1）文件：新建、打开、保存等；
（2）编辑：剪切、复制、粘贴、删除、替换、查找等；
（3）格式：字体、字号、字形、颜色等；
（4）其他功能。
要求：
（1）使用 Python 解释器安装时自带的基础库（例如 tkinter 等）.（尽量不要使用其他第三方库等）
（2）能提供记事本的相关功能；
（3）人机界面美观友好；
（4）给出程序运行结果截屏图。
（5）要求在源程序中标注必要的注释。

（6）要求对程序的使用方法和运行方法进行必要说明。
评价标准：
（1）源程序功能（占 50%）
（2）源代码质量（占 20%）
（3）人机界面美观友好（占 10%）
（4）算例运行结果截屏图、源程序注释和程序的使用说明文档（占 20%）

课程设计备选题目四：酒店点餐系统的实现
内容：

用 Python 语言编程实现一个酒店点餐系统，软件主要功能包括：餐饮商品包括热菜、凉菜、汤菜、主食、点心、水果、酒水饮料等种类；顾客可以在餐饮商品目录中按大类、小类等顺序浏览选择各种商品；当顾客浏览到某个具体商品时，系统能提供该商品的名称、单价、简介、图片等信息；当用户在选择商品过程中，可以随时查看已选商品的名称、单价、数量和总价等信息，也可以随时从已选商品中修改某个商品的购买数量或删除某个商品；等等。

学有余力的同学可以实现高阶版本：用数据库实现商品信息管理；可以根据商品名称查询商品信息；可以根据菜品特点查询菜品信息（例如输入“肉”字，可以模糊查找出全部肉菜供选择，输入“川菜”、“湘菜”等文字可以查看相关菜系的信息和菜品目录）；可以根据顾客的就餐人数或经费预算推荐相应的商品购买方案；顾客可以给商品评价打分或建议留言；系统管理员可以对商品数据库进行插入、删除、修改更新等等管理操作；可以生成报表；等等功能。
要求：
（1）商品类型包括热菜、凉菜、汤菜、主食、点心、水果、酒水饮料等种类。
（2）能为用户提供点餐服务功能。
（3）餐饮商品库可以采用文本文件、CSV 文件或数据库等来实现。
（4）要求在源程序中标注必要的注释。
（5）要求对程序的使用和运行方法进行必要说明，并有 GUI 运行界面截屏图。
（6）课程设计要提交程序源代码及附属的测试商品库文档等（便于阅卷测试）。
评价标准：
（1）源程序功能（占 50%）
（2）源代码质量（占 20%）
（3）GUI 界面美观友好（占 20%）
（4）GUI 运行界面截屏图、源程序注释和使用说明文档（占 10%）

课程设计备选题目五：虚拟网店购物系统的实现
内容：

用 Python 语言编程实现一个虚拟网店购物系统，软件主要功能包括：商品包括食品、日常生活用品、服装鞋帽、家用电器、家具、图书、药品等等；顾客可以在商品目录中按大类、小类等顺序浏览选择各种商品；当顾客浏览到某个具体商品时，系统能提供该商品的名称、单价、简介、图片等信息；当用户在选择商品过程中，可以随时查看已选商品的名称、单价、数量和总价等信息，也可以随时从已选商品中修改某个商品的购买数量或删除某个商品；等等。

学有余力的同学可以实现高阶版本：用数据库实现商品信息管理；可以根据商品名称查询商品信息；可以根据商品特点查询商品信息（例如输入“鞋”字，可以模糊查找出全部鞋

类商品供选择，输入“儿童用品”等文字可以查看相关儿童类商品的信息和商品目录）；可以根据商品类别和经费预算推荐相应的商品购买方案；顾客可以给商品评价打分或建议留言；系统管理员可以对商品数据库进行插入、删除、修改更新等等管理操作；可以生成报表；等等功能。

要求：

（1）商品类型包括食品、日常生活用品、服装鞋帽、家用电器、家具、图书、药品等等。

（2）能为用户提供网店购物服务功能。

（3）商品库可以采用文本文件、CSV 文件或数据库等来实现。

（4）要求在源程序中标注必要的注释。

（5）要求对程序的使用和运行方法进行必要说明，并有 GUI 运行界面截屏图。

（6）课程设计要提交程序源代码及附属的测试商品库文档等（便于阅卷测试）。

评价标准：

（1）源程序功能（占 50%）

（2）源代码质量（占 20%）

（3）GUI 界面美观友好（占 20%）

（4）GUI 运行界面截屏图、源程序注释和使用说明文档（占 10%）

第二部分 《Python 语言程序设计》平时作业

说明：

（1）平时作业占课程总成绩的 50%。

（2）请给出下面 100 道单项选择题的答案。

（3）请按照附件答题卡格式提交答案：只提交答案，不抄题。

单项选择题：

1. 下面关于 Python 语言的说法中，错误的是（　　）

A. 是脚本语言

B. 是动态类型语言

C. 是编译型语言

D. 是面向对象程序设计语言

2. 下面关于 Python 语言的说法中，错误的是（　　）

A. Python 2.x 和 Python 3.x 两个版本是不兼容的

B. Python 程序有命令交互和程序批处理两种运行模式

C. 实现相同功能，Python 程序的源代码行数只有其他语言的 1/10~1/5

D. 与 C、C++语言相比，Python 程序的跨平台性较差

3. 下面关于 Python 语言的说法中，错误的是（　　）

A. Python 语言采用的是基于值的内存管理方式，不同的值分配不同的内存空间

B. 对于一些简单对象，如一些数值较小的整型（int）对象，Python 采取重用对象内存的办法来提高内存利用效率

C. 源代码的强制缩进规则中，Tab 符号（键）和空格符号（键）等效，可以混用
D. 在 Python 中，长语句的续行符是“\”
4. 下面哪个不是 Python 语言的基本数据类型？（　　）
A. 复数
B. 字典
C. 字符
D. 字符串
5. 下面引用 Python 语言模块库的表达方式中错误的是（　　）
A. import math
B. import math.*
C. from math import *
D. import math as mymath
6. 下面关于 Python 语言的说法中，错误的是（　　）
A. Python 源程序中，注释是以“%”符号开头的
B. Python 语言的标识符是要区分英文字母大小写的
C. Python 语言的标识符支持汉字，但不能以数字开头
D. Python 语言中二进制整数是以“0b”或“0B”开头的
7. 下面哪个是 Python 语言的复数？（　　）
A. 3+4i
B. 3+4*i
C. 3+4j
D. 3+4*j
8. 下列 Python 表达式中，值与其余 3 个不同的是（　　）
A. 3 in [1,2,3]
B. 3 in (1,2,3)
C. 3 in {1,2,3}
D. 3 in {'a':1,'b':2,'c':3}
9. 下列 Python 表达式值最大的是（　　）
A. (-9)^(-7)
B. (-9)&(-7)
C. (-9)|(-7)
D. (-9)//(-7)
10. 在 Python 命令提示符状态下查看 math 模块中 sin()函数的帮助信息，正确的语句是（　　）
A. help(math.sin())
B. help(math.sin)
C. import math;help(math.sin())
D. import math;help(math.sin)
11. 在 Python 语言中，要让字符串不按转义字符解释，需要在字符串前加上哪个前缀字符？（　　）
A. r 或 R
B. @
C. !
D. ~
12. 在 Python 命令提示符状态下，执行语句“2+3,4+5;print(_)”的输出是（　　）

A. 5
B. 9
C. (5,9)
D. 出错
13. 在 Python 中，执行语句“x=[1,2,3];y=x;x[1]=4;print(y)”的输出是（　　）
A. [1,2,3]
B. [4,2,3]
C. [1,4,3]
D. 出错
14. 在 Python 中，执行语句“x=[1,2,3];y=x;x=x+[4];print(y)”的输出是（　　）
A. [1,2,3]
B. [1,2,3,4]
C. [4]
D. 出错
15. 在 Python 中，执行语句“x=2+3j;y=x;x=x+x.real+x.imag;print(y)”的输出是（　　）
A. 7+3j
B. (7+3j)
C. (2+3j)
D. (4+6j)
16. 在 Python 中，执行语句“x={'a':1,'b':2,'c':3};y=x;x['b']=4;print(y)”的输出是（　　）
A. {'a':1,'b':2,'c':3}
B. {'a':1,'b':4,'c':3}
C. {'a':1,'b':2,'c':3,'b':4}
D. 出错
17. 在 Python 中，执行语句“x={'a':2,'b':3};y=x;x={'c':4,'d':5};print(y)”的输出是（　　）
A. {'a': 2, 'b': 3}
B. {'c':4,'d':5}
C. {'a': 2, 'b': 3,'c':4,'d':5}
D. 出错
18. 在 Python 中，执行语句“x={'a':2,'b':3};y=x;x['c']=4;x=x.update({'d':5});print(y)”的输出是（　　）
A. {'a': 2, 'b': 3, 'c': 4}
B. {'a': 2, 'b': 3, 'd': 5}
C. {'a': 2, 'b': 3, 'c': 4, 'd': 5}
D. {'d': 5}
19. 读入 1,2,3 依次赋值给变量 x,y,z，读入语句是：x,y,z=eval(input('a,b,c=')），则正确的键盘输入数据方式是（　　）。注：□代表空格键，↙代表<Enter>键
A. 1□2□3↙
B. 123↙
C. 1,2,3↙
D. 1↙2↙3↙
20. 语句“x=12;y=34.5678;print('x=%-5d,%08.2f'%(x,y))”执行后，输出结果是（　　）。注：□代表空格
A. x=12□□□,00034.57

B. x=□□□12, □□□34.57
C. x=12□□□,00034.56
D. x=□□□12, 34.56□□□
21. 语句“x=12;y=34.5678;print('x=',format(x,'^6d'),'y=',format(y,'>8.2f'),sep='')”执行后，输出结果是（　　）。注：□代表空格
A. x=□□12□□,□□□34.57
B. x=□□□□12, □□□34.56
C. x=□□12□□,34.57□□□
D. x=12□□□□, 34.56□□□
22. 语句“x=12;y=34.5678;print('x={0:*^6},y={1:>08.4}'.format(x,y))”执行后，输出结果是（　　）。注：□代表空格
A. x=**12**,y=□□□34.57
B. x=**12**,y=34.57□□□
C. x=**12**,y=34.57000
D. x=**12**,y=00034.57
23. 执行下面语句后的输出是（　　）。注：□代表空格
d=[1,(2,3,4),[5,6,7]];e=('ABC','DEFG','HIJK');print('{m[1][2]:*^3},{n[2][3]:#<3}'.format(m=d,n=e))
A. □3□,F□□
B. *3*,F##
C. □4□,K□□
D. *4*,K##
24. 执行语句“d={'aa':2,'bb':3,'cc':4};e=[5,6,7];print('{0[bb]:^03},{1[2]:<03}'.format(d,e))”后，输出结果是（　　）。注：□代表空格
A. 030,600
B. □3□,6□□
C. 030,700
D. □3□,7□□
25. 在 Python 命令提示符状态下，浏览历史命令中上一条命令的快捷键是（　　）
A. <Ctrl>+<P>
B. <Ctrl>+<PageUp>
C. <Alt>+<P>
D. <Alt>+<PageUp>
26. 在 Python 命令提示符状态下，浏览历史命令中下一条命令的快捷键是（　　）
A. <Ctrl>+<N>
B. <Ctrl>+<PageDown>
C. <Alt>+<N>
D. <Alt>+<PageDown>
27. 执行语句：“print(3==3!=4<6>5<=8>=7)”后的输出结果是（　　）
A. True
B. False
C. 1
D. 出错
28. 执行语句：print(1 if 2>3 else (4 if 5>6 else 7))后的输出结果是（　　）
A. 0

B. 1
C. 4
D. 7
29. 运行下面程序段后的输出结果是（　　）
```
s,i=0,1
for i in range(3):
    s+=i
    i+=2
else:
    print(s)
```
A. 1
B. 3
C. 4
D. 6
30. 运行下面程序段后的输出结果是（　　）
```
t=[str(i)*j for i in range(5) if i%2==0 for j in range(5) if j%2!=0]
print(t)
```
A. ['01', '21', '41', '03', '23', '43']
B. ['0', '2', '4', '000', '222', '444']
C. ['0', '000', '2', '222', '4', '444']
D. 出错
31. 运行下面程序段后的输出结果是（　　）
```
d={'x':2,'y':3,'z':4}
t=[i+'='+str(j) for i,j in d.items()]
print(t)
```
A. [x=2, y=3, z=4]
B. ['x':2,'y':3,'z':4]
C. ['x=2', 'y=3', 'z=4']
D. ['x=2, y=3, z=4']
32. 运行下面程序段后的输出结果是（　　）
```
k=0
t=[22,'55',33,0.66,44,'AA',True]
for i in t:
    if type(i)==type(0):
        k+=1
else:
    print(k)
```
A. 0
B. 3
C. 5
D. 7
33. 运行下面程序段时输入“5”后的输出结果是（　　）
```
t=''
k=int(input('key='))
```

```
s=[chr(ord('A')+i) for i in range(5)]
for i in s:t+=i
t=t[::-1]
print(t[:-k:-2])
```

A. EC

B. AC

C. ECA

D. ACE

34. 运行下面程序段后的输出结果是（　　）

```
t=''
s='a1b2c3#d@f*'
for i in s:
    if not('a'<=i<='z' or 'A'<=i<='Z'):
        continue
    t+=i
print(t)
```

A. abcdf

B. 123

C. 123#@*

D. #@*

35. 在字符串 s 中，若整数 k 满足 0≤k<len(s)，则下列各项中与 s[k]字符相同的是（　　）

A. s[len(s)-k+1]

B. s[k-len(s)+1]

C. s[k-len(s)-1]

D. s[k-len(s)]

36. 在字符串 s 中，若整数 k 满足-len(s)≤k≤-1，则下列各项中与 s[k]字符相同的是（　　）

A. s[k+len(s)+1]

B. s[k+len(s)-1]

C. s[-k-len(s)]

D. s[k+len(s)]

37. 在字符串 s 中，若整数 k 满足-len(s)≤k≤-1，则下列各项中与 s[k:]相同的是（　　）

A. s[:-k:-1]

B. s[:k+len(s)]

C. s[-k-len(s):]

D. s[k+len(s):]

38. 在字符串 s 中，若整数 k 满足 0≤k<len(s)，则下列各项中与 s[:k]相同的是（　　）

A. s[-k::-1]

B. s[:len(s)-k]

C. s[:k-len(s)]

D. s[:len(s)-k+1]

39. 下列选项中与其余三项的运算结果不同的是（　　）

A. 'abc'.join('**')

B. 'abc'.center(5,'*')

C. '**'.join('abc')

D. ('abc'.zfill(4)+'0').replace('0','*')

40. 已知：s='10203040'；若想得到：'1234'；则不正确的表达式是（　　）

A. ''.join([i for i in s if i!='0'])

B. s.strip('0')

C. ''.join(s.split('0'))

D. s.replace('0','')

41. 若已知：s='uestc'；则下列表达式中与其余三项的值不同的是（　　）

A. [(i,s[i]) for i in range(5)]

B. [(eval(i[0]),i[1]) for i in zip('01234',s)]

C. list(enumerate(s))

D. [(i,j) for i,j in zip(b'01234',s)]

42. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.findall('[0-9]+','a2bc34d5')

B. import re;re.findall('[^a-z]+','a2bc34d5')

C. import re;re.findall('[^A-Z]+','a2bc34d5',re.I)

D. import re;re.findall('[^abc]+','a2bc34d5')

43. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.findall('[abcd]','a2bc34d5')

B. import re;re.findall('[^0-9]','a2bc34d5')

C. import re;re.findall('[a-z]','a2bc34d5')

D. import re;re.findall('[a-z]+','a2bc34d5')

44. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.findall('[a-z]{2,3}','a2bc34def5ghij6')

B. import re;re.findall('[\D]+','a2bc34def5ghij6')

C. import re;re.findall('bc|def|ghi','a2bc34def5ghij6')

D. import re;re.findall('[^aj\d]+','a2bc34def5ghij6')

45. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.sub('ab','-','ab2ba34bab567aabb890aaabbb')

B. import re;re.sub('a+b','-','ab2ba34bab567aabb890aaabbb',count=4)

C. import re;re.sub('[a]{1}[b]{1}','-','ab2ba34bab567aabb890aaabbb',count=4)

D. import re;re.sub('(ab){1}','-','ab2ba34bab567aabb890aaabbb')

46. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.sub('b+\d{0,}','-','ab2ba34bab567aabb890aaabbb')

B. import re;re.sub('b{1,}\d*','-','ab2ba34bab567aabb890aaabbb')

C. import re;re.sub('b+|b+\d|b+\d\d|b+\d\d\d','-','ab2ba34bab567aabb890aaabbb')

D. import re;re.sub('bb*\d*','-','ab2ba34bab567aabb890aaabbb')

47. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;re.sub('\d*','','a2b33c444d5555e666f77g8')

B. import re;''.join(re.findall('[a-z]','a2b33c444d5555e666f77g8'))

C. import re;z=re.compile('\d{0,}');re.sub(z,'','a2b33c444d5555e666f77g8')

D. import re;z=re.compile('\d+');re.subn(z,'','a2b33c444d5555e666f77g8')

48. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）

A. import re;z=re.search('\W','a2b@3cd#4*5%6e');print(z)

B. import re;z=re.search('[^a-zA-Z0-9]','a2b@3cd#4*5%6e');print(z)

C. import re,string;z=re.search('['+string.punctuation+']','a2b@3cd#4*5%6e');print(z)
D. import re,string;z=re.search('[^'+string.digits+']','a2b@3cd#4*5%6e');print(z)
49. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. import re,string;z=re.finditer('['+string.digits+']','a2b3cd');[i.group() for i in z]
B. import re,string;z=re.finditer('[^'+string.ascii_letters+']','a2b3cd');[i.group() for i in z]
C. import re;z=re.findall('\D','a2b3cd');z
D. import re;z=re.finditer('[^a-z]','a2b3cd',re.I);[i.group() for i in z]
50. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. import re;re.split('\W+','ab?@c#$def&*%gh/\ijk')
B. import re;re.split('\W\W*','ab?@c#$def&*%gh/\ijk')
C. import re;re.findall('\W+','ab?@c#$def&*%gh/\ijk')
D. import re;re.findall('\w+','ab?@c#$def&*%gh/\ijk')
51. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. len([0 for i in range(3) for j in range(4)])
B. len([i for i in range(3) for j in range(4)])
C. len([j for i in range(4) for j in range(3)])
D. len([(i,j) for i in range(5) for j in range(4)if i>j])
52. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. s=['abc','x','uestc'];s.sort();s
B. s=['abc','x','uestc'];sorted(s)
C. s=['abc','x','uestc'];sorted(s,key=len)
D. s=['abc','x','uestc'];sorted(s,key=max)
53. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. t=((i,ord(i)) for i in 'uestc');dict(t)
B. t=enumerate('uestc');dict(t)
C. t=[(i,ord(i)) for i in 'uestc'];dict(t)
D. t={(i,ord(i)) for i in 'uestc'};dict(t)
54. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. s='a2bc34d5';str([i for i in s if i.isdigit()])
B. s='a2bc34d5';''.join([i for i in s if i.isnumeric()])
C. s='a2bc34d5';t=[i for i in s if i.isalpha()];''.join([i for i in s if i not in t])
D. s='a2bc34d5';t=[];x=[t.append(i) for i in s if i.isdecimal()];''.join(t)
55. 在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. s='a2bc34d5';t=list(s);x=[t.remove(i) for i in s if i.isdigit()];t
B. s='a2bc34d5';[i for i in s if i.isalpha()]
C. s='a2bc34d5'; t=[];x=[t.append(i) for i in s if i.isalpha()];t
D. import re;s='a2bc34d5';re.findall('[a-z]+',s,re.I)
56. 如果 s={2,3,4};t={3,4,5};则在 Python 交互式命令窗口中运行下面程序段，输出结果与其余三项不同的是（　　）
A. x=s.difference(t);y=t.difference(s);x.union(y)
B. x=s.union(t);x.difference(s.intersection(t))
C. s.update(t);s.difference(x.intersection(t))
D. s.symmetric_difference(t)
57. 向集合 s 中添加元素 2（假设 s 中原来没有该元素），错误的是（　　）

A. s.add(2)
B. s.update({2})
C. s.union({2})
D. s.symmetric_difference_update({2})
58. 从集合 s 中删去元素 2（假设 s 中原来有该元素），错误的是（　　）
A. s.difference_update({2})
B. s.intersection_update(s.difference({2}))
C. s.discard({2})
D. s.symmetric_difference_update({2})
59. 如果 s={2,3,4};则下列表达式含义与其余三项不同的是（　　）
A. s.clear()
B. s.difference(s)
C. s.difference_update(s.copy())
D. s.symmetric_difference_update(s)
60. 如果 s={1,3,5};t={2,3,4};则下列表达式的值与其余三项不同的是（　　）
A. s
B. s-t|s&t
C. s^t|s-(t-s)
D. (s|t)-s^t|s-t
61. 如果 s={2,3,4};欲删去 s 中的元素 3，则下列表达式不能实现该操作的是（　　）
A. s-={3}
B. s^={3}
C. s&=s-{3}
D. s|=s^{3}
62. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
f=(lambda:2,lambda x:x*3,lambda y:y**2)
def main():
    print(f[0]()+f[1](2)+f[2](3))
    return
main()
```

A. 0　　B. 17　　C. None　　D. 出错
63. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(x):
    return x(2)
def main():
    y=f(lambda x:x**2)
    print(y)
    return
main()
```

A. 0　　B. 4　　C. None　　D. 出错
64. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def g():
    return lambda x,y=3:x+y
def main():
```

```
    f=g()
    print(f(2))
    return
main()
```

A. 2　　B. 3　　C. 5　　D. 出错

65. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(x):
    x[1]=5
def main():
    x=[2,3,4]
    f(x)
    print(x)
    return
main()
```

A. [2,5,4]　　B. [5,3,4]　　C. [2,3,4]　　D. [2,5,3,4]

66. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(x,y=4,z=5):
    print(x,y,z,sep=',')
def main():
    f(y=2,x=3)
    return
main()
```

A. 3,2,5　　B. 3,4,5　　C. 2,3,5　　D. 出错

67. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(x=6,y=7,*z,**t):
    print(x,y,z,t,sep=',')
def main():
    f(2,a=4,b=5)
    return
main()
```

A. 2,7,(),{'a': 4, 'b': 5}　　B. 6,7,(2,),{'a': 4, 'b': 5}

C. 2,7,(4,),{'b': 5}　　D. 2,7,(4,5),{}

68. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(x=6,y=7,*z):
    print(x,y,z,sep=',')
def main():
    x,*y,z=2,3,4,5
    f(x,*y,z)
    return
main()
```

A. 2,[3, 4],(5,)　　B. 2,3,(4, 5)　　C. 2,[3],(4, 5)　　D. 2,[3,4], (5)

69. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f():
    global x
```

```
    x,y=4,5
    def g():
        nonlocal y
        global x
        x,y=6,7
        print('{}{}'.format(x,y),end='')
    g()
    print('{}{}'.format(x,y),end='')
x,y=2,3
f()
print('{}{}'.format(x,y),end='')
```

A. 674523　　B. 674543　　C. 674743　　D. 676763

70. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(y=3):
    x=y
    print(x,end='')
    def g():
        nonlocal x
        print(x,end='')
        x=4
    g()
    print(x,end='')
def main():
    global x
    x=2
    f(x)
    print(x,end='')
main()
```

A. 2242　　B. 2224　　C. 2442　　D. 3242

71. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(y=3):
    x=y
    print(x,end='')
    def g():
        global x
        print(x,end='')
        x=4
        def h():
            global x
            print(x,end='')
            x=5
        h()
    g()
    print(x,end='')
```

```
def main():
    global x
    x=2
    f()
    print(x,end='')
main()
```

A. 32435　　B. 22425　　C. 32455　　D. 32432

72. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
def f(i):
    if(i>=len(s)):
        return
    else:
        f(i+1)
    print(s[i],end='')
def main():
    global s
    s='uestc'
    f(0)
main()
```

A. uestc　　B. ctseu　　C. None　　D. 出错

73. 按照源程序文件方式运行下面程序 c.py 后的输出结果是（　　）

在 D:\user 文件夹中建立如下三个 Python 源程序文件：

```
# D:\user\a.py
def f():
    print('AA',end='')
# D:\user\b.py
def g():
    print('BB',end='')
# D:\user\c.py
from a import f
from b import g
def main():
    f()
    g()
    print('CC')
main()
```

A. AABBCC　　B. CC　　C. 无输出　　D. 出错

74. 按照源程序文件方式运行下面程序 c.py 后的输出结果是（　　）

在 D:\user\aa 文件夹中建立如下 a.py 源程序文件：

```
# D:\user\aa\a.py
def f():
    print('AA',end='')
```

在 D:\user\bb 文件夹中建立如下 b.py 源程序文件：

```
# D:\user\bb\b.py
```

```
def g():
    print('BB',end='')
```
在 D:\user\cc 文件夹中建立如下 c.py 源程序文件：
```
# D:\user\cc\c.py
import sys
sys.path.append('D:\\user\\aa')
sys.path.append('D:\\user\\bb')
from a import f
from b import g
def main():
    f()
    g()
    print('CC')
main()
```
A. AABBCC　　B. CC　　C. 无输出　　D. 出错

75. 按照源程序文件方式运行下面程序 b.py 后的输出结果是（　　）

在 D:\user 文件夹中建立如下两个 Python 源程序文件：
```
# D:\user\a.py
def f():
    print(__name__,end='')
if __name__=='__main__':
    f()
    import b
    b.g()
# D:\user\b.py
def g():
    print(__name__,end='')
if __name__=='__main__':
    g()
    import a
    a.f()
```
A. ab　　B. __main__a　　C. __main__b　　D. __main____main__

76. 如果当前源程序模块文件是主模块文件，则其系统变量“__name__”的值为（　　）

A. __main__

B. 其源程序文件的文件根名（不加后面的.py，即模块名）

C. 其源程序文件中的函数名

D. 其源程序文件所在的文件夹名字

77. 如果当前源程序模块文件是非主模块文件，则其系统变量“__name__”的值为（　　）

A. __main__

B. 其源程序文件的文件根名（不加后面的.py，即模块名）

C. 其源程序文件中的函数名

D. 其源程序文件所在的文件夹名字

78. 按照源程序文件方式运行下面程序段后的输出结果是（　　）
```
import string
```

```
f=open(r'D:\user\t.txt','w+')
f.write(string.digits+'\n'+string.ascii_uppercase+'\n'+string.ascii_lowercase)
print(f.tell(),end=',')
f.seek(0)
s=f.read(1)
print(f.tell(),end=',')
t=f.readline()
print(f.tell(),end=',')
print(len(s+t))
f.close()
```

A. 65,1,12,12　　B. 66,1,12,11　　C. 66,1,12,12　　D. 65,1,12,11

79. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
import string
f=open(r'D:\user\t.dat','wb+')
s='电子科技大学'
sb=s.encode('gbk')
f.write(sb)
f.seek(-12,1)
print(f.tell(),end=',')
t=f.read(4)
x=t.decode('gbk')
print(x,end=',')
print(f.tell())
f.close()
```

A. 0,电子,4　　B. 0,电子科技,4　　C. 6,科技大学,18　　D. 4,科技,8

80. 按照源程序文件方式运行下面程序段后的输出结果是（　　）

```
class MyException(Exception):
    def __init__(self,data):
        self.data=data
try:
    s=(2,3,4)
    if s[0]==True: raise MyException("MM")
    if s[2]==3:raise SyntaxError("TT")
    s[1]=2
    print("AA",end='')
except MyException as e:print(e,end='')
except TypeError as e:print("BB",end='')
except SyntaxError as e:print(e,end='')
except NameError as e:print('CC',end='')
except:print('DD',end='')
else:print('EE',end='')
finally:print('FF',end='')
print('GG',end='')
```

A. AAFFGG

B. BBFFGG
C. MMFFGG
D. TTFFGG
81. 按照源程序文件方式运行下面程序段后的输出结果是（　　）
```
try:
    x,*x=1+2,2+3,3+4
    print('AA',end='')
except:
    print('BB',end='')
else:
    print('CC',end='')
finally:
    print('DD',end='')
print('EE',end='')
```
A. BBDDEE
B. BBCCDDEE
C. AACCDDEE
D. 出错
82. 下面程序段中的断言语句：assert x==3,"x!=3"，它不等价于（　　）
```
try:
    x=2
    assert x==3,"x!=3"
except AssertionError as e:
    print("Assertion error: ",e)
```
A. if　x!=3:raise AssertionError("x!=3")
B. if　x-3:raise AssertionError("x!=3")
C. if　True if x==3 else False :raise AssertionError("x!=3")
D. if not (x==3):raise AssertionError("x!=3")
83. 类中私有属性和私有方法的名称是（　　）
A. 以双下划线（__）开头
B. 以双下划线（__）开头，且必须以双下划线（__）结尾
C. 用 private 声明
D. 以单下划线（_）开头，且必须以单下划线（__）结尾
84. 在 Python 类中方法外定义的属性是（　　）
A. 实例属性
B. 类属性
C. 私有属性
D. 公有属性
85. Python 语言中，在哪种方法内部只能访问类属性，不能直接访问实例属性？（　　）
A. 实例方法
B. 带有装饰器@classmethod 的方法
C. 带有装饰器@staticmethod 的方法
D. 构造方法
86. Python 类中，哪种方法可以没有形式参数？（　　）

A. 构造方法和析构方法
B. 实例方法
C. 带有装饰器@staticmethod 的方法
D. 带有装饰器@classmethod 的方法
87. 在 Python 中，同一个类可以有多少个类对象和实例对象？（　　）
A. 只有 1 个类对象，也只有 1 个实例对象
B. 只有 1 个类对象，可以有任意个实例对象
C. 只有 1 个实例对象，可以有任意个类对象
D. 可以有任意个类对象，也可以有任意个实例对象
88. 在 Python 中，哪种属性它被本类的所有对象共享，其内存只有一份？（　　）
A. 类属性
B. 实例属性
C. 公有属性
D. 私有属性
89. Python 类中，在构造方法内定义的属性是（　　）
A. 实例属性
B. 类属性
C. 私有属性
D. 公有属性
90. Python 类中，构造方法和析构方法属于（　　）
A. 类方法
B. 实例方法
C. 静态方法
D. 全局函数
91. Python 中，当创建一个实例对象时，一定会调用哪个方法？（　　）
A. 构造方法
B. 析构方法
C. 类方法
D. 静态方法
92. Python 类中，哪种方法的形式参数表中，最左边的形式参数变量必须是当前类对象的引用？（　　）
A. 构造方法和析构方法
B. 实例方法
C. 带有装饰器@staticmethod 的方法
D. 带有装饰器@classmethod 的方法
93. Python 类中，哪种方法的形式参数表中，最左边的形式参数变量必须是当前实例对象的引用？（　　）
A. 私有方法
B. 实例方法
C. 带有装饰器@staticmethod 的方法
D. 带有装饰器@classmethod 的方法
94. 下面程序中划横线处应填（　　）

```
from tkinter import *
def f(value):
```

```
    c.delete(ALL)
    c.create_arc(0,0,200,200,extent=str(v.get()),fill="blue")
w=Tk()
v=StringVar()
s=Scale(w,_____=0,to=1000,resolution=1,orient=HORIZONTAL,variable=v,command=f)
s.pack()
c=Canvas(w,bg="yellow")
c.pack()
w.mainloop()
```

A. start_

B. start

C. from_

D. from

95. 下面程序中划横线处应填（　　）

```
from tkinter import *
def f(value):
    t.config(font=("隶书",_____(v.get())))
w=Tk()
w.geometry("300x200")
v=StringVar()
s=Scale(w,from_=1,to=100,resolution=1,orient=HORIZONTAL,variable=v,command=f)
s.pack()
t=Label(w,text="电子科技大学",fg="blue")
t.pack()
w.mainloop()
```

A. tuple

B. list

C. set

D. int

96. 下面程序中划横线处应填（　　）

```
from tkinter import *
def f():
    c=("#ff0000","#00ff00","#0000ff","#ffff00","#ff00ff","#00ffff","#ffffff","#000000")
    for i in range(s._____):
        if s.selection_includes(i):
            t["fg"]=c[i]
w=Tk()
w.geometry("300x300")
t=Label(w,text="电子科技大学",font=("隶书",30,"bold"))
t.place(x=10,y=10)
g=LabelFrame(w,text="颜色")
g.place(x=10,y=70)
v=StringVar()
s=Listbox(g,listvariable=v)
```

```
s.pack()
s.insert(END,"红色")
s.insert(END,"绿色")
s.insert(END,"蓝色")
s.insert(END,"黄色")
s.insert(END,"紫色")
s.insert(END,"青色")
s.insert(END,"白色")
s.insert(END,"黑色")
b=Button(w,text="设置",command=f)
b.place(x=200,y=70)
w.mainloop()
```

A. len

B. len()

C. size

D. size()

97. 下面程序中划横线处应填（　　）

```
from tkinter import *
def f1():
    s=["隶书",30,"bold","italic"]
    if v1.get()==0:
        s.remove("bold")
    if v2.get()==0:
        s.remove("italic")
    t.config(font=_____(s))
def f2():
    c1.select()
    c2.select()
    t.config(font=("隶书",30,"bold","italic"))
w=Tk()
w.geometry("300x200")
t=Label(w,text="电子科技大学",font=("隶书",30),fg="blue")
t.place(x=10,y=10)
g=LabelFrame(w,text="格式")
g.place(x=10,y=70)
v1=IntVar()
v2=IntVar()
c1=Checkbutton(g,text="粗体",variable=v1,command=f1)
c1.pack()
c2=Checkbutton(g,text="斜体",variable=v2,command=f1)
c2.pack()
b=Button(w,text="一键粗斜体",command=f2)
b.place(x=100,y=70)
w.mainloop()
```

A. str
B. list
C. tuple
D. set
98. 对于下面的程序段，下列描述中错误的是（　　）
from tkinter import *;w=Tk();w.config(bg="yellow");w.title("uestc");
A. 语句：w.geometry("400x300+200-100"); 设置窗口距离桌面左边的距离为 200 像素
B. 语句：w.geometry("400x300+200-100"); 设置窗口距离桌面下边的距离为 100 像素
C. 语句：w.geometry("400x300-200+100"); 设置窗口距离桌面右边的距离为 200 像素
D. 语句：w.geometry("400x300-0-0"); 和 w.geometry("400x300+0+0"); 等效
99. 下面程序中划横线处不能填（　　）

```
from tkinter import *
def f1():
    for i in range(s2.size()):
        if s2.selection_includes(i):
            s1.insert(END,s2.get(i))
def f2():v1._________
def f3():
    m2=["铅笔","圆珠笔","钢笔","橡皮擦","作业本","笔袋"]
    v2.set(m2)
def f4():
    m3=["雪碧","可乐","咖啡","矿泉水","冰红茶"]
    v2.set(m3)
def f5():
    global m1
    v2.set(m1)
w=Tk()
w.geometry("500x300")
t=Label(w,text="电子科技大学 校园商城",font=("隶书",20,"bold"),fg="blue")
t.place(x=100,y=10)
g1=LabelFrame(w,text="购物车")
g2=LabelFrame(w,text="货架")
g3=LabelFrame(w,text="商品类别")
g1.place(x=10,y=50)
g2.place(x=200,y=50)
g3.place(x=380,y=150)
v1=StringVar()
v2=StringVar()
s1=Listbox(g1,listvariable=v1)
s1.pack()
s2=Listbox(g2,listvariable=v2,selectmode=MULTIPLE)
s2.pack()
m1=["苹果","梨子","葡萄","芒果","樱桃","草莓"]
for i in m1:
```

```
        s2.insert(ACTIVE,i)
b1=Button(w,text="加入购物车",command=f1)
b2=Button(w,text="清空购物车",command=f2)
b3=Button(g3,text="文具",command=f3)
b4=Button(g3,text="饮料",command=f4)
b5=Button(g3,text="水果",command=f5)
b1.place(x=380,y=50);b2.place(x=380,y=100);
b3.pack();b4.pack();b5.pack();
w.mainloop()
```

A. set()

B. set(())

C. set("")

D. set([])

100. 下面程序中划横线处应填（　　）

```
from tkinter import *
def f1():t.config(fg=_________)
def f2():t.config(font=("隶书",s.get()))
def f3():
    if r["state"]!="disabled":
        r.config(state="disabled")
        b["text"]="启用"
    else:
        r.config(state="normal")
        b.config(text="禁用")
w=Tk()
w.geometry("300x210")
t=Label(w,text="电子科技大学",font=("隶书"))
t.place(x=10,y=10)
c=StringVar()
c.set("cyan")
g1=LabelFrame(w,text="颜色")
g1.place(x=10,y=70)
Radiobutton(g1,text="红色",variable=c,value="red",command=f1).pack()
Radiobutton(g1,text="绿色",variable=c,value="green",command=f1).pack()
Radiobutton(g1,text="蓝色",variable=c,value="blue",command=f1).pack()
r=Radiobutton(g1,text="紫色",variable=c,value="#ff00ff",command=f1);r.pack()
b=Button(w,text="禁用",command=f3);b.place(x=200,y=70)
s=IntVar()
g2=LabelFrame(w,text="字号")
g2.place(x=100,y=70)
Radiobutton(g2,text="10",variable=s,value=10,command=f2).pack()
Radiobutton(g2,text="20",variable=s,value=20,command=f2).pack()
Radiobutton(g2,text="30",variable=s,value=30,command=f2).pack()
w.mainloop()
```

A. c.get

B. c.get()

C. r.get

D. r.get()

<完>