项目10 库的构建与发布

知识目标

★ 了解 Python 的计算生态。
★ 了解生态库的功能及应用。
★ 熟悉库的构建与发布。
★ 掌握常用内置库的使用方法。

技能目标

★ 会利用随机库解决各种编码问题。
★ 会进行计算机绘图。
★ 会利用时间库进行时间控制。

素养目标

★ 养成科学严谨的工作态度。
★ 提升编程思维的创新能力。
★ 培养团队协作的沟通能力。

学习任务

★ 随机生成验证码。
★ 绘制黄色六边形。
★ 模拟时钟。

学习路径

★ 通过信息单学习相关的理论知识。
★ 通过任务单进行相关的实践操作。
★ 通过评价单获知学习的不足并及时改进。
★ 通过完成巩固练习，实现课后的再学习再提高。

Python 的生态库非常丰富，内容包含了当今的所有新一代信息技术，为各个领域 Python 的应用提供了极大便利。

本项目将通过三个任务的讲解，介绍 Python 中的计算生态及部分生态库的功能和应用领域、库的构建与发布、Python 常用内置库的应用。

任务 1　随机生成验证码

任务单

任务编号	10-1	任务名称	随机生成验证码
任务简介	Python 内置库中的 random 库可以生成随机数据。本任务通过 random 库中的 randint() 函数生成随机整数功能来生成随机密码，以完成设置密码任务		
设备环境	台式机或笔记本电脑，建议使用 Windows 10 以上操作系统		
所在班级		小组成员	
任务难度	中级	指导教师	
实施地点		实施日期	年　月　日
任务要求	创建 Python 文件，完成以下操作： （1）在程序开头加入注释信息，说明程序； （2）导入 random 模块； （3）创建一个空列表 code； （4）生成 6 个随机字符逐个拼接到 code 后面； （5）运行 Python 程序，显示正确的运行结果		

信息单

任务编号	10-1	任务名称	随机生成验证码

一、Python 的计算生态

Python 计算生态涵盖网络爬虫、数据分析、数据可视化、游戏开发、文本处理、图形艺术、图形用户界面、机器学习、虚拟现实、Web 开发、网络应用开发等领域，为各领域的 Python 应用提供了便利，这里仅介绍部分生态库的功能和应用。

（一）网络爬虫

网络爬虫是一种按照一定的规则，自动从网络上抓取信息的程序或者脚本。通过网络爬虫可以代替手工完成很多工作。

网络爬虫程序涉及 HTTP 请求、Web 信息提取、网页数据解析等操作，Python 计算生态通过 Requests、Python-Goose、Re、Beautiful Soup、Scrapy 和 PySpider 库为网络爬虫提供了强有力的支持。这些生态库的功能见表 10-1。

表 10-1　Python 网络爬虫生态库

库名	功能
Requests	Requests 提供了简单易用的类 HTTP 协议，支持连接池、SSL、Cookies，是 Python 最主要的、功能最丰富的网络爬虫功能库
Python-Goose	Python-Goose 专用于从文章、视频类型的 Web 页面中提取数据
Re	Re 提供了定义和解析正则表达式的一系列通用功能，除网络爬虫外，还适用于各类需要解析数据的场景
Beautiful Soup	Beautiful Soup 用于从 HTML、XML 等 Web 页面中提取数据，它提供一些便捷的、Python 式的函数，使用起来非常简单
Scrapy	Scrapy 支持快速、高层次的屏幕抓取和批量、定时的 Web 抓取以及结构性数据的抓取，是一款优秀的网络爬虫框架
PySpider	PySpider 也是一款爬虫框架，它支持数据库后端、消息队列、优先级、分布式架构等功能。与 Scrapy 相比，它灵活便捷，更适合小规模的爬取工作

（二）数据分析

数据分析是指用适当的统计分析方法对收集来的大量数据进行分析，将它们加以汇总、理解与消化，以求最大化地发挥数据的作用。

Python 计算生态通过 Numpy、Pandas、Scipy 库为数据分析领域提供支持。这些生态库的功能见表 10-2。

任务编号	10-1	任务名称	随机生成验证码

表 10-2　Python 数据分析生态库

库名	功能
Numpy	数据分析离不开科学计算，Numpy 定义了表示 N 维数组对象的类型 ndarray，通过 ndarray 对象可以便捷地存储和处理大型矩阵；包含了成熟的用于实现线性代数、傅里叶变换和随机数生成的函数，能以优异的效率实现科学计算
Pandas	Pandas 是一个基于 Numpy 开发的、用于分析结构化数据的工具集，它为解决数据分析任务而生，同时提供数据挖掘和数据清洗功能
Scipy	Scipy 是 Python 科学计算程序中会使用的核心库，它用于有效地计算 Numpy 矩阵，可以处理插值、积分、优化等问题，也能处理图像和信号、求解常微分方程数值。

（三）数据可视化

数据可视化是一门关于数据视觉表现形式的科学技术研究，它既要有效传达数据信息，也需兼顾信息传达的美学形式，二者缺一不可。

Python 计算生态主要通过 Matplotlib、Seaborn、Mayavi 生态库为数据可视化领域提供支持。这些生态库的功能见表 10-3。

表 10-3　Python 数据可视化生态库

库名	功能
Matplotlib	Matplotlib 是一个基于 Numpy 开发的 2D Python 绘图库，该库提供了上百种图形化的数据展示形式。Matplotlib 库中 pyplot 包内包含一系列类似 MATLAB 中绘图功能的函数，利用 Matplotlib.pyplot，开发者编写几行代码便可生成可视化图表
Seaborn	Seaborn 在 Matplotlib 的基础上进行了更高级的封装，支持 Numpy 和 Pandas，但它比 Matplotlib 调用更简单，效果更丰富，多数情况下可利用 Seaborn 绘制具有吸引力的图表
Mayavi	Mayavi 是一个用于实现可视化功能的 3D Python 绘图库，它包含用于实现图形可视化和处理图形操作的 mlab 模块，支持 Numpy 库

（四）游戏开发

游戏开发是 Python 的一项重要功能，Python 计算生态通过 PyGame、Panda3D 库为游戏开发领域提供支持。这些生态库的功能见表 10-4。

表 10-4　Python 游戏开发生态库

库名	功能
Pygame	Pygame 是为开发二维游戏而设计的 Python 第三方库，开发人员利用 Pygame 中定义的接口，可以方便快捷地实现图形用户界面创建、图形和图像的绘制、用户键盘和鼠标操作的监听、播放音频等游戏中常用的功能
Panda3D	Panda3d 是由迪士尼 VR 工作室和卡耐基梅隆娱乐技术中心开发的一个三维渲染和游戏开发库，该库强调能力、速度、完整性和容错能力，提供场景浏览器、性能监视器和动画优化工具，并通过完善代码来有效降低开发者跟踪和分析错误的难度

任务编号	10-1	任务名称	随机生成验证码

（五）图形艺术

图形艺术是一种通过标志来表现意义的艺术。标志是一些单纯、显著、易识别的具有指代性或具有表达意义、情感和指令等作用的物象、图形或文字符号，也是图形艺术的表现手段。

Python 计算生态通过 Quads、ascii_art 和 turtle 库为图形艺术领域提供支持。这些生态库的功能见表 10-5。

表 10-5　Python 图形艺术生态库

库名	功能
Quads	Quads 是一个基于四叉树和迭代操作的图形艺术库，它是以图像作为输入，将输入图像分为 4 个象限，根据输入图像中的颜色为每个象限分配平均颜色，误差最大的象限会被分成 4 个子象限以完善图像，以上过程重复 n 次
ascii_art	ascii_art 是一种使用纯字符表示图像的技术，ascii_art 库可对接收到的图片进行转换，以字符形式重构图片并输出
turtle	turtle 提供了绘制线、圆以及其他形状的函数，使用该库可以创建图形窗口，在图形窗口中通过简单重复动作直观地绘制界面与图形

（六）图像处理

图像处理是指数字图像处理。图像处理技术一般包括图像压缩、增强和复原、图像匹配、描述和识别。

Python 通过 Numpy、Scipy、Pillow、OpenCV-Python 等库为图像处理领域提供支持。这些生态库的功能见表 10-6。

表 10-6　Python 图像处理生态库

库名	功能
Numpy	数字图像的本质是数组，Numpy 定义的数组类型非常适用于存储图像；Numpy 提供基于数组的计算功能，利用这些功能可以很方便地修改图像的像素值
Scipy	Scipy 提供了对 N 维 Numpy 数组进行运算的函数，这些函数实现的功能，包括线性和非线性滤波、二值形态、B 样条插值等都适用于图像处理
Pillow	Pillow 库是 PIL 库的一个分支，也是支持 Python3 的图像处理库，该库提供了对不同格式图像文件的打开和保存操作，也提供了包括点运算、色彩空间转换等基本的图像处理功能
Opencv-Python	Opencv-Python 是 OpenCV 的 Python 版 API，OpenCV 是基于 BSD 许可发型的跨平台计算机视觉库，该库内部代码由 C/C++ 编写，实现了图像处理和计算机视觉方面的很多通用算法；Opencv-Python 以 Python 代码对 OpenCV 进行封装，因此该库即方便使用又非常高效

任务编号	10-1	任务名称	随机生成验证码

二、库的构建与发布

库是 Python 中常提及的概念，但事实上 Python 中的库只是一种对特定功能集合的统一说法而不是严格定义，其具体表现形式为模块和包。

（一）模块

1. 模块及其功能

随着程序复杂度的提高及代码量的同步增长，如果还是在一个文件中编写代码，代码的维护就会越来越困难。为了保证代码的可维护性，通常将一些功能性代码放在其他文件中。人们把用于存放功能性代码的文件称为模块，即 Python 中的每个“.py”文件。

作为一种功能强大、使用便捷的编程语言，Python 支持以模块的形式组织代码。Python 中的模块包括三种类型，一是系统内置的标准模块；二是 Python 使用者贡献的第三方模块；三是 Python 开发者自行定义的自定义模块。通过这些丰富且强大的模块可以极大地提高开发者的开发效率。这些模块中，标准模块先导入再使用，第三方模块则需要安装后再导入和使用，而自定义模块直接使用。

模块的功能主要体现在以下三个方面：

（1）代码的可重用性。模块可以在文件中永久保存代码。当编写好一个模块后，只要编程过程中需要用到该模块中的某个功能，无须做重复性的编写工作，直接在程序中导入该模块即可使用该功能。

（2）实现共享的服务和数据。从操作层面看，模块对实现跨系统共享的组件很方便，只需要存在一份单独的副本即可。

（3）系统命名空间的划分。模块可以被认为是变量名的软件包。模块将变量名封装进自包含的软件包，避免了变量名的冲突。要使用这些变量，不精确定位是看不到这些变量的。

2. 模块的分类

Python 中，模块可分为三类，分别是内置模块、第三方模块和自定义模块。

（1）内置模块。内置模块指 Python 安装时自带的模块，可直接导入程序供开发人员使用，如 random 模块、time 模块等。

（2）第三方模块。第三方模块指由第三方制作发布的、供给大家使用的 Python 模块，在使用前需要开发人员自行安装。

（3）自定义模块。自定义模块指开发人员在编写程序的过程中自行编写的、存放功能性代码的“.py”文件。

3. 模块的安装

利用 Python 内置的 pip 工具（安装 Python3.13.3 时会自动安装该工具）可以非常方便地安装 Python 第三方模块，该工具可在命令行中使用，语法格式如下：

任务编号	10-1	任务名称	随机生成验证码

pip install 模块名

例如，安装用于开发游戏的 pygame 模块，具体命令如下：

pip install pygame

专家点睛

pip 是在线工具，pip 命令执行后，它需要联网获取模块资源，若没有网络或网络不佳，pip 将无法顺利安装第三方模块。

4. 模块的使用

用户可以通过在当前“.py”文件中导入其他“.py”文件来使用被导入的内容。

假设有 Python 模块“test.py”，其包含的代码如下：

课堂检验 1

具体操作：

```
def swap(x,y):
    x,y=y,x
    return x,y
```

利用 import 语句或 from…impor 语句在当前程序中导入“test.py”模块，就可以在当前程序中使用模块中包含的代码。

课堂检验 2

具体操作：

```
import test
print(test.swap(2,78))
```

运行结果如图 10-1 所示。

```
(78, 2)
```

图 10-1 课堂检验 2 结果

模块既可以被导入到其他程序中使用，也可以作为脚本直接使用。在实际开发中，为了保证模块实现功能达到预期效果，开发人员通常会在模块文件中添加若干测试代码，对模块中的功能代码进行测试。

例如，如下的 sum() 函数为一个模块“sum.py”，在模块中添加了测试代码来对 sum() 函数的功能代码进行测试。

课堂检验 3

具体操作：

```
# 功能代码
def sum(n):
    sum=0
    for i in range(1,n+1):
```

任务编号	10-1	任务名称	随机生成验证码

```
            sum+=i
        return sum
# 测试代码
result=sum(100)
print(f'function test:{result}')
```

运行结果如图 10-2 所示。

```
function test:5050
```

图 10-2　课堂检验 3 结果

由上可知，将以上文件作为脚本直接执行，就可利用测试代码测试 sum() 函数的功能，但此时会出现一个问题，就是在其他文件中导入“sum.py”模块，模块中的测试代码会在其他文件执行时一起执行。

例如，导入“sum.py”模块，使用模块中的 sum() 函数。

课堂检验 4

具体操作

```
import sum
s=sum.sum(10)
print(s)
```

运行结果如图 10-3 所示。

```
function test:5050
55
```

图 10-3　课堂检验 4 结果

为解决以上出现的问题，Python 为“.py”文件定义了一个名字属性 __name__，在文件中对 __name__ 属性的取值进行判断，当 __name__ 的值为 __main__ 时，说明“.py”文件以脚本形式执行，否则，说明“.py”文件作为模块被导入其他程序中。根据以上原理对模块进行修改。

课堂检验 5

具体操作：

```
# 功能代码
def sum(n):
    sum=0
    for i in range(1,n+1):
        sum+=i
    return sum
# 测试代码
if __name__ == '__main__':
```

任务编号	10-1	任务名称	随机生成验证码

```
result=sum(100)
print(f'function test:{result}')
```

当再次导入“sum.py”模块调用其中的 sum() 函数时，运行结果如图 10-4 所示。

55

图 10-4　课堂检验 5 结果

5. 模块的导入

在使用模块中定义的内容前需要先将模块导入到当前程序中。Python 使用 import 关键字导入模块，其语法格式如下：

```
import 模块 1, 模块 2,…
```

例如，在程序中导入 pygame 模块，具体代码如下：

```
import pygame
```

模块导入后，可以通过点字符“.”调用模块中的内容，其语法格式如下：

```
模块 . 函数
模块 . 变量
```

例如，使用 import 语句导入 pygame 模块后要调用其中的 init() 函数，代码如下：

```
Pygame.init()
```

通过点字符调用模块中的内容可避免多个模块中存在同名函数时代码产生歧义，但若不存在同名函数，可使用 from…import…语句直接将模块的指定内容导入到程序中，并在程序中直接使用模块中的内容。

例如，将 pygame 模块中 init() 函数导入到程序中，并直接使用该函数，其代码如下：

```
from pygame import init
init()
```

使用 from…import…语句也可以将指定模块的全部内容导入到当前程序中，此时用“*”指代模块中的全部内容。

例如，将 pygame 模块的全部内容导入到程序中，代码如下：

```
from pygame import *
```

专家点睛

虽然 from…import * 可以方便地导入一个模块中的所有内容，但考虑到代码的可维护性，此种方式不宜被过多地使用。

（二）包

Python 中的包是一个包含“__init__.py”文件的目录，该目录下还包含一些模块以及子包，如图 10-5 所示。

任务编号	10-1	任务名称	随机生成验证码

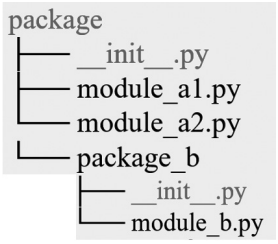

图 10-5　package 包

专家点睛

包中的“__init__.py”文件可以为空，但必须存在，否则包将退化为一个普通目录。

“__init__.py”文件有两个作用，一个是标识当前目录是一个 Python 的包；另一个是模糊导入。如果“__init__.py”文件中没有声明 __all__ 属性，那么使用 from ... import * 导入的内容为空 。

包的导入有两种方法，一是使用 import 导入，二是使用 from…import…导入。

1. 使用 import 导入

使用 import 导入包中的模块时，需要在模块名的前面加上包名，格式为“包名 . 模块名”。若要使用已导入模块中的函数时，需要通过“包名 . 模块 . 函数”实现。

```
import package_demo.module
package_demo.module.test(1,3)
```

2. 使用 from…import…导入

通过 from…import…导入包中模块包含的内容，若需要使用导入模块中的函数，需要通过“模块 . 函数”实现。

```
from package_demo import operation_demo
operation_demo.test(2,3)
```

（三）库的发布

Python 中的第三方库是由使用者自行编写与发布的模块或包，同样的，人们也可以将自己编写的模块与包作为库发布。

具体操作步骤如下：

（1）在与待发布的包同级的目录中创建“setup.py”文件。

（2）编辑“setup.py”文件，在该文件中设置包中包含的模块。

（3）在“setup.py”文件所在目录下打开命令行，使用 python setup.py build 命令构建 Python 库。

（4）在“setup.py”文件所在目录下打开命令行，使用 python setup.py sdist 命令创建库的安装包。

任务编号	10-1	任务名称	随机生成验证码

三、常用的内置库

Python 中常见的内置库有 random、turtle、time 库等。

（一）random 库

random 库是 Python 内置的标准库，在程序中导入该库，可利用库中的函数生成随机数据。见表 10-7 为 random 库中常用的函数。

表 10-7　random 库中常用函数

函数	说明
random()	返回 (0,1] 之间的随机小数
randint(x,y)	返回 [x,y] 之间的整数
choice(seq)	从序列 seq 中随机返回一个元素
uniform(x,y)	返回 [x,y] 之间的浮点数

程序举例：随机生成旅游信息。

想要出去旅游，但时间还没有确定，需要在 10 ～ 19 号之间挑选一个时间作为出游时间，同时在北京、云南、浙江、海南、四川中挑选一个出游的地方。

本实例拟采用 random 库的来完成，其代码如下：

```
import random
print(" 旅游的时间为 :",end='')
print(random.randint(10,19)," 号 ",sep='')
place=[' 北京 ',' 云南 ',' 浙江 ',' 海南 ',' 四川 ']
print(" 出游的地方是 :",end='')
print(random.choice(place))
```

以上代码中，由于旅游的时间和地点没有确定，但范围给出，对于时间而言，采用随机整数函数 randint() 在给定范围内确定，对于地点，由于是列表，采用随机选择函数 choice() 在给定范围内进行选择，最终确定旅游的时间和地点。

运行结果如图 10-6 所示。

旅游的时间为：13号
出游的地方是：云南

图 10-6　随机生成旅游信息

程序举例：生成随机密码。

编写程序，在 26 个大小写字母和 9 个数字组成的列表中随机生成 10 个 8 位密码。注意：在 Python 中，所有字符是按照 unicode 码进行编码而非 ASCII 码。

本实例拟采用随机整数 randint() 函数来完成，其代码如下：

任务编号	10-1	任务名称	随机生成验证码

```
import random
def rancre():
    password=''
    for i in range(8):
        u=random.randint(0,62)
        if u>=10:
            if 90<(u+55)<97:
                password+=chr(u+62)
            else:
                password+=chr(u+55)
        else:
            password+='%d'%u
    return password
def main():
    for i in range(1,11):
        print(' 生成的第 {} 个密码是 :{}'.format(i,rancre()))
main()
```

运行结果如图 10-7 所示。

```
生成的第1个密码是:f4r5CeYg
生成的第2个密码是:ggpYg5Xc
生成的第3个密码是:d9EBRmQC
生成的第4个密码是:MZABHeQr
生成的第5个密码是:qTmNl2hq
生成的第6个密码是:TjSe7oVh
生成的第7个密码是:nsRLrRZH
生成的第8个密码是:enadfdK6
生成的第9个密码是:nC8fElBd
生成的第10个密码是:bEIpqPfl
```

图 10-7　生成随机密码

任务实践：随机生成验证码

很多网站的注册登录业务都加入了验证码技术，以区分用户是人还是计算机，有效地防止刷票、论坛灌水、恶意注册等行为。目前验证码的种类层出不穷，其生成方式也越来越复杂，常见的验证码是由大写字母、小写字母、数字组成的 6 位验证码。

本任务要求编写程序，实现随机生成 6 位验证码的功能。

本任务的 6 位验证码是由 6 个字符组成，每个字符都是随机字符，要实现随机字符的功能需要用到随机数 random 库。

打开 PyCharm，在右侧的工具栏中单击“AI 聊天”图标，即打开插件 AI Assistant，如图 10-8 所示。

任务编号	10-1	任务名称	随机生成验证码

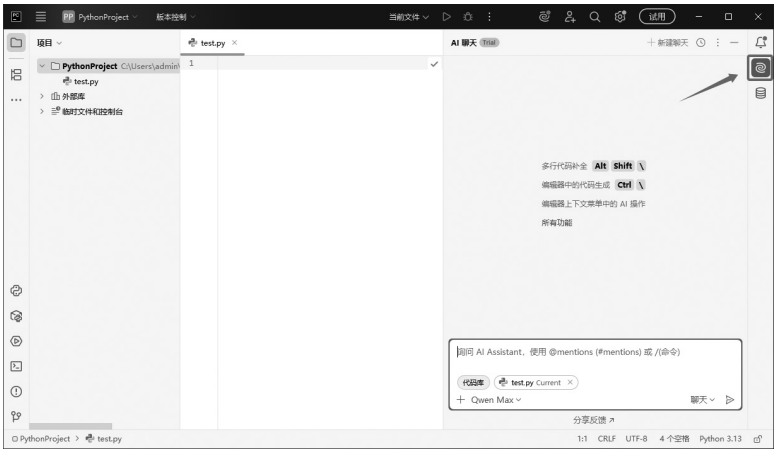

图 10-8　AI Assistant 聊天界面

输入提示词：

假设你是一个有着 20 年编程经验的程序员主管，同时精通 Python 技术，现在帮我写一个 Python 程序，生成由大写字母、小写字母、数字组成的 6 位验证码。

随后在 AI Assistant 中就会生成初始代码，如图 10-9 所示。

图 10-9　AI Assistant 生成的随机生成验证码代码

任务编号	10-1	任务名称	随机生成验证码

进行二次修改后的代码如下：

```
# 随机生成验证码
import random
import string

def generate_verification_code():
    # 定义字符集：大写字母、小写字母和数字
    upper_case = string.ascii_uppercase
    lower_case = string.ascii_lowercase
    digits = string.digits
    # 确保验证码中至少有一个大写字母、一个小写字母和一个数字
    code = [
        random.choice(upper_case),
        random.choice(lower_case),
        random.choice(digits)
    ]
    # 剩余的 3 个字符从所有字符集中随机选择
    all_characters = upper_case + lower_case + digits
    code += random.choices(all_characters, k=3)
    # 打乱验证码顺序
    random.shuffle(code)
    # 将列表转换为字符串并返回
    return ''.join(code)

# 测试生成验证码
if __name__ == "__main__":
    verify_code = generate_verification_code()
    print(f" 生成的 6 位验证码是：{verify_code}")
```

运行程序，结果如图 10-10 所示

```
生成的6位验证码是：l9ZgVe
```

图 10-10　随机生成验证码

评价单

任务编号	10-1		任务名称	随机生成验证码
评价项目			自评	教师评价
课堂表现	学习态度（20 分）			
	课堂参与（10 分）			
	团队合作（10 分）			
技能操作	创建 Python 文件（10 分）			
	编写 Python 代码（40 分）			
	运行并调试 Python 程序（10 分）			
评价时间	年　月　日		教师签字	

评价等级划分

项目		A	B	C	D	E
课堂表现	学习态度	在积极主动、虚心求教、自主学习、细致严谨上表现优秀	在积极主动、虚心求教、自主学习、细致严谨上表现良好	在积极主动、虚心求教、自主学习、细致严谨上表现较好	在积极主动、虚心求教、自主学习、细致严谨上表现尚可	在积极主动、虚心求教、自主学习、细致严谨上表现不佳
	课堂参与	积极参与课堂活动，参与内容完成的很好	积极参与课堂活动，参与内容完成的好	积极参与课堂活动，参与内容完成的较好	能参与课堂活动，参与内容完成的一般	能参与课堂活动，参与内容完成的欠佳
	团队作	具有很强的团队合作能力，能与老师和同学进行沟通交流	具有良好的团队合作能力，能与老师和同学进行沟通交流	具有较好的团队合作能力，尚能与老师和同学进行沟通交流	能与团队进行合作，与老师和同学进行沟通交流能力一般	不能与团队进行合作，不能与老师和同学进行沟通交流
技能操作	创建	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	编写	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	运行并调试	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成

任务 2　绘制黄色六边形

任务单

任务编号	10-2	任务名称	绘制黄色六边形
任务简介	turtle 库也是 Python 内置的标准库，它提供了绘制线条、圆或弧、其他形状的函数。本任务就是利用 turtle 库的函数绘制黄色六边形		
设备环境	台式机或笔记本电脑，建议使用 Windows 10 以上操作系统		
所在班级		小组成员	
任务难度	初级	指导教师	
实施地点		实施日期	年　　月　　日
任务要求	创建 Python 文件，完成以下操作： （1）在程序开头加入注释信息，说明程序功能； （2）指定绘制的边形数 N 和填充的颜色 c； （3）根据边数计算出旋转角度 angle； （4）利用 for 循环绘制长度为 150 的 n 边形； （5）填充指定的颜色 c； （6）运行 Python 程序，显示正确的运行结果		

信息单

任务编号	10-2	任务名称	绘制黄色六边形

（二）turtle 库

turtle 英文是海龟的意思。turtle 库是 Python 语言中一个很流行的绘制图像的外部函数库。在程序中导入该库，可利用库中的函数创建图形窗口，在图形窗口中通过简单重复的动作直观地绘制界面和图形。turtle 库的逻辑非常简单，利用其内置的函数，用户可以像使用笔在纸上绘图一样。在 turtle 画布上绘制图形。turtle 的使用主要分为创建窗口、设置画笔和绘制图形（移动画笔）。表 10-8 为 turtle 库中常用的函数。

表 10-8　turtle 库中常用函数

函数	说明
shape(name)	设置海龟展示的形状。默认形状为箭头。这里 name 必须为 TurtleScreen 形状库里的形状，可以填写以下形状：arrow（箭头）、turtle（海龟）、circle（圆）、square（方块）、triangle（三角形）、classic（带尾箭头）
setup(width,height,startx,starty)	设置主窗体的大小和位置
screensize(width,height,color)	设置画布的大小为 weight×hight，颜色为 color
pensize(width) 或 turtle.width()	设置画笔宽度，若为 None 或者为空则为当前画笔宽度
pencolor(colorstring) 或 pencolor((r,g,b))	设置画笔颜色，若为空则为当前画笔颜色。其包括 grey（灰）、darkgreen（深绿）、gold（金）、violet（紫罗兰）、purple（紫）
penup() 或 pu() 或 up()	抬起画笔，之后移动画笔不绘制形状
goto(x,y)	将画笔移到绝对坐标（x,y）处
pendown() 或 pd() 或 down()	落下画笔，之后移动画笔绘制形状
forward(distance) 或 fd(distance)	沿着前进的方向行进 distance 像素的距离。当为负数时，表示向相反方向前进
back(distance) 或 bk(distance)	沿着后退的方向行进 distance 像素的距离。当为负数时，表示向相反方向前进
setheading(to_angle) 或 seth(to_angle)	设置当前行进的方向为 to_angle（按逆时针方向）绝对角度
left(to_angle) 或 right(to_angle)	设置当前行进的方向为向左（右）to_angle 度
circle(radius,extent/None)	根据半径 radius 绘制 extent 角度的弧形。当 radius 为正数时，半径在左侧（沿逆时针方向画弧），当 radius 为负数时，半径在右侧（沿顺时针方向画弧）。当 extent 为空或为 None，则绘制圆形
turtle.fillcolor(color)	设置要填充的颜色。
turtle.begin_fill() 或 end_fill()	开始填充或结束填充

程序举例：绘制星星。

利用所学函数绘制在蓝色天空中有一颗闪耀的小星星。

任务编号	10-2	任务名称	绘制黄色六边形

本实例首先绘制蓝色窗口，设置图形填充的颜色为黄色，设置海龟形状后，利用前进 fd() 函数和左转 left() 函数完成星星的绘制。其代码如下：

```
import turtle
turtle.screensize(None,None,"blue")
turtle.fillcolor("yellow")
turtle.shape("turtle")
turtle.begin_fill()
turtle.fd(100)
turtle.left(144)
turtle.fd(100)
turtle.left(144)
turtle.fd(100)
turtle.left(144)
turtle.fd(100)
turtle.left(144)
turtle.fd(100)
turtle.end_fill()
```

运行结果如图 10-11 所示。

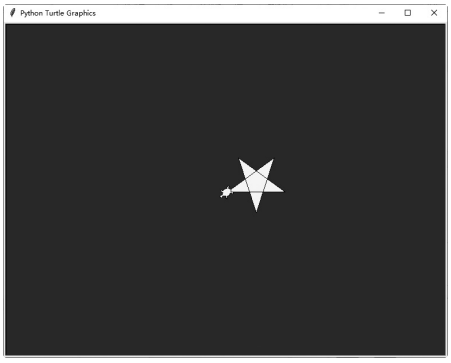

图 10-11　绘制星星

程序举例：绘制 Python 蟒蛇。

利用所学函数绘制紫色的 Python 蟒蛇。

本实例首先绘制蓝色窗口，设置图形填充的颜色为黄色，设置海龟形状后，利用前进 fd() 函数和左转 left() 函数完成星星的绘制。

其代码如下：

```
import turtle
turtle.setup(650, 350, 200, 200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
```

任务编号	10-2	任务名称	绘制黄色六边形

```
turtle.pencolor("purple")
turtle.seth(-40)
for i in range(4):
    turtle.circle(40, 80)
    turtle.circle(-40, 80)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40* 2/3)
```

上面代码中，运行结果如图 10-12 所示。

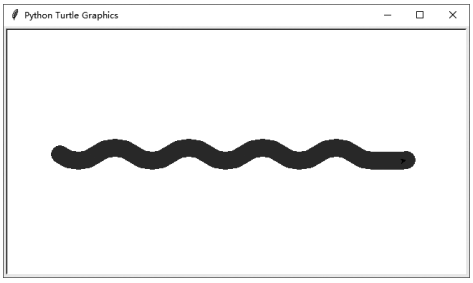

图 10-12　绘制蟒蛇

任务实践：绘制黄色六边形

如果你喜欢绘画，那么一定要尝试 Python 的内置模块 turtle 模块，turtle 是一个专门的绘图模块，你可以利用该模块通过程序绘制一些简单图形。

本任务要求编写程序，使用 turtle 模块绘制一个如图 10-13 所示的黄色六边形。

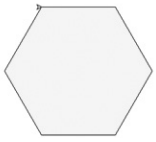

图 10-13　绘制黄色六边形

绘制 N 边形的功能可以视为将画笔沿顺时针方向旋转固定角度画指定颜色的直线的操作，直到绘制指定边数为止结束绘图，之后在画好的图形上填充颜色即可。此 N 边形绘制的过程中涉及的 turtle 模块中的函数及说明如下：

（1）begin_fill()：开始填充。

（2）end_fill()：停止填充。

（3）fillcolor()：设置填充的颜色。

（4）forward()：将画笔向前方移动指定的距离。

（5）right()：将画笔顺时针旋转指定的角度。

任务编号	10-2	任务名称	绘制黄色六边形

（6）done()：启动事件循环，必须位于末尾位置。

打开 PyCharm，在右侧的工具栏中单击“AI 聊天”图标，即打开插件 AI Assistant，如图 10-14 所示。

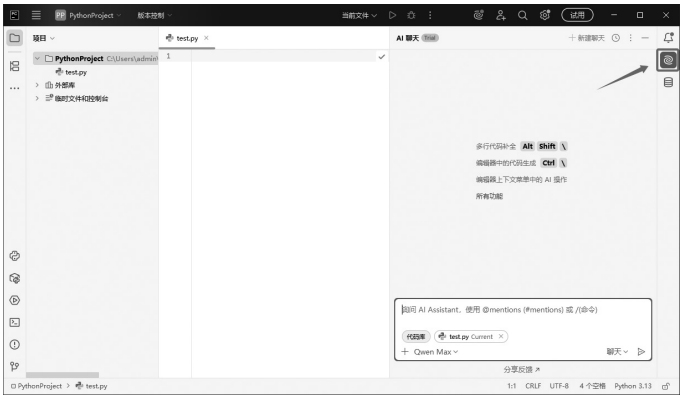

图 10-14　AI Assistant 聊天界面

输入提示词：

假设你是一个有着 20 年编程经验的程序员主管，同时精通 Python 技术，现在帮我写一个 Python 程序，绘制一个黄色六边形。

随后在 AI Assistant 中就会生成初始代码，如图 10-15 所示。

图 10-15　AI Assistant 生成的绘制黄色六边形代码

任务编号	10-2	任务名称	绘制黄色六边形

进行二次修改后的代码如下：

```
# 绘制黄色六边形
import turtle
# 设置窗口大小
turtle.setup(650, 350, 200, 200)
# 抬起笔，移动到起点位置
turtle.penup()
turtle.goto(-100, -100)                # 移动到合适的位置以确保六边形居中
turtle.pendown()
# 设置画笔颜色和填充颜色
turtle.pencolor("black")               # 边框颜色
turtle.fillcolor("yellow")             # 填充颜色为黄色
# 开始填充
turtle.begin_fill()
# 绘制六边形
for _ in range(6):
    turtle.forward(100)                # 每条边的长度
    turtle.left(60)                    # 转角为 60 度
# 结束填充
turtle.end_fill()
# 隐藏画笔
turtle.hideturtle()
# 结束绘制
turtle.done()
```

以上代码首先使用 import 语句导入了 turtle 模块，通过 input() 函数确定绘制的边数和颜色；其次调用 fillcolor() 函数设置填充的颜色，根据边数计算顺时针旋转的角度 angle，调用 beglin_fill()/end_fill() 开始填色，然后使用 for 语句绘制指定边数的多边形；最后调用 done() 函数停止绘制，保持主窗口不关闭。

运行结果如图 10-16 所示。

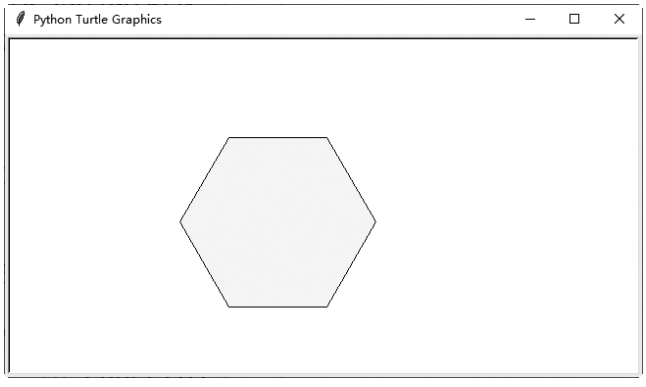

图 10-16　绘制黄色六边形

评价单

任务编号	10-2	任务名称	绘制黄色六边形
评价项目		自评	教师评价
课堂表现	学习态度（20 分）		
	课堂参与（10 分）		
	团队合作（10 分）		
技能操作	创建 Python 文件（10 分）		
	编写 Python 代码（40 分）		
	运行并调试 Python 程序（10 分）		
评价时间	年　月　日	教师签字	

评价等级划分

项目		A	B	C	D	E
课堂表现	学习态度	在积极主动、虚心求教、自主学习、细致严谨上表现优秀	在积极主动、虚心求教、自主学习、细致严谨上表现良好	在积极主动、虚心求教、自主学习、细致严谨上表现较好	在积极主动、虚心求教、自主学习、细致严谨上表现尚可	在积极主动、虚心求教、自主学习、细致严谨上表现不佳
	课堂参与	积极参与课堂活动，参与内容完成的很好	积极参与课堂活动，参与内容完成的好	积极参与课堂活动，参与内容完成的较好	能参与课堂活动，参与内容完成的一般	能参与课堂活动，参与内容完成的欠佳
	团队合作	具有很强的团队合作能力，能与老师和同学进行沟通交流	具有良好的团队合作能力，能与老师和同学进行沟通交流	具有较好的团队合作能力，尚能与老师和同学进行沟通交流	能与团队进行合作，与老师和同学进行沟通交流能力一般	不能与团队进行合作，不能与老师和同学进行沟通交流
技能操作	创建	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	编写	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	运行并调试	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成

任务 3　模 拟 时 钟

任务单

任务编号	10-3	任务名称	模拟时钟
任务简介	time、datetime、calendar 库是 Python 内置的用于时间管理的标准库，它是与时间处理相关的库。本任务就是利用 time 库的函数绘制并控制时钟运动		
设备环境	台式机或笔记本电脑，建议使用 Windows 10 以上操作系统		
所在班级		小组成员	
任务难度	中级	指导教师	
实施地点		实施日期	年　　月　　日
任务要求	创建 Python 文件，完成以下操作： （1）在程序开头加入注释信息，说明程序功能； （2）使用 import 语句导入 turtle 模块和 datetime 模块； （3）定义设定指针跳动距离的函数 skip()； （4）定义绘制表盘外框的函数 setup_clock()； （5）定义绘制表盘函数 make_hand()； （6）定义指针初始化函数 init()； （7）分别定义获取星期函数 week() 和获取日期函数 day()； （8）定义动态显示指针函数 tick()； （9）通过主函数 main() 调用以上自定义函数； （10）运行 Python 程序，显示正确的运行结果		

信息单

任务编号	10-3	任务名称	模拟时钟

（三）time、datetime、calendar 库

在 Python 程序开发过程中根据时间来选择不同的处理场景的情况很多，例如，动态时钟秒针的运动时间、游戏的防沉迷控制、外卖平台店铺的营业时间管理、数据的记录及日志的处理等。Python 语言提供了三个与时间管理有关的库，它们是 time 库、datetime 库和 calendar 库。

1. time 库

time 库是 Python 中最常用的与时间处理相关的库。time 库中常用函数见表 10-9。

表 10-9　time 库中常用函数

函数	说明
time()	获取当前时间，结果为实数，单位为秒
sleep(secs)	进入休眠状态，时长由 secs 确定，单位为秒
strptime(string[,format])	将一个年月日时间格式的字符串解析为时间元组
localtime([secs])	以 struct_time 类型输出本地时间
asctime([tuple])	获取时间字符串，或将时间元组转换为字符串
mktime(tuple)	将时间元组转换为秒数
strftime(format[,tuple])	返回字符串表示的当地时间，格式由 format 决定

程序举例：计算时间。

时间是可以进行加减运算的，一般情况下，时间是以时间戳的形式进行加减运算。

本实例首先导入 time 库，获取系统的第一个当前时间，这个时间是以时间戳的形式表示。然后让系统等待几秒后，再获取系统的第二个当前时间，计算这两个时间的和与差，即完成时间的计算。

其代码如下：

```
import time
time_1=time.time()
time.sleep(3)
time_2=time.time()
print(time_1+time_2)
print(time_1+time_2)
```

运行结果如图 10-17 所示。

```
3495979312.0617094
3495979312.0617094
```

图 10-17　计算时间

任务编号	10-3	任务名称	模拟时钟

2. datetime 库

以不同格式显示日期和时间是程序中最常用到的功能。Python 提供了一个处理时间的标准函数库 datetime，它提供了一系列由简单到复杂的时间处理方法。datetime 库可以从系统中获得时间，并以用户选择的格式进行输出。

datetime 库中常用函数见表 10-10。

表 10-10 datetime 库中常用函数

函数	说明
MINYEAR	该量为常量，是获取能表示的最小年份
MAXYEAR	该量为常量，是获取能表示的最大年份
date()	获取当前的日期
time()	获取当前的时间
datetime()	获取当前的日期和时间
timedelta()	获取两个时间的时间差
tzinfo()	获取时区信息

程序举例：确定某天是该年的第几天。

时间和日期是可以进行操作的。

本实例首先导入 datetime 库，获取指定的日期，然后从这一年的 1 月 1 日起计算天数。

其代码如下：

```
import datetime
def day_year(year,month,day):
    date1=datetime.date(year=int(year),month=int(month),day=int(day))
    date2=datetime.date(year=int(year),month=1,day=1)
    return (date1-date2).days+1
y=input(' 请输入年份：')
m=input(' 请输入月份：')
d=input(' 请输入日期：')
n=day_year(y,m,d)
print(f'{y} 年 {m} 月 {d} 日是这一年的第 {n} 天 ')
```

上面代码中，定义了一个根据指定年月日计算第几天的函数 day_year()，函数中通过 date() 函数获取指定年月日的日期和当年 1 月 1 日的起止日期，两个日期的差加 1 即为当年的天数。

运行结果如图 10-18 所示。

```
请输入年份：2025
请输入月份：5
请输入日期：20
2025年5月20日是这一年的第140天
```

图 10-18 确定时间

任务编号	10-3	任务名称	模拟时钟

3. calendar 库

calendar 库是 python 中常用的标准库。该库包含了很多方法和类用来处理年历和月历，可以生成文本形式的日历、月历等。

calendar 库中常用函数见表 10-11。

表 10-11　calendar 库中常用函数

函数	说明
calendar(year,w=2,l=1,c=6)	返回一个多行字符串的 year 年年历，3 个月一行，间隔举例为 c，每日宽度间隔为 w 字符，每行长度为 21×w+18+2×c，1 是每星期行数
firstweekday()	返回当前每周起始日期的设置。默认情况下，首次载入 calendar 模块时返回 0，即星期一
isleap(year)	如果 year 是闰年返回 True，否则返回 False
leapdays(y1,y2)	返回在 y1,y2 两年间的闰年总数
month(year,month,w=2,l=1)	返回一个多行字符串格式的年月日历，两行标题，一周一行，每日宽度间隔为 w 字符，每行的长度为 7×w+6。1 是每星期的行数
monthcalendar(year,month)	返回一个整数的单层嵌套列表。每个子列表装载代表一个星期的整数。year 年 month 月外的日期都设为 0，范围内的日子都由该月第几日表示，从 1 开始
monthrange(year,month)	返回两个整数，第一个是该月的星期几的日期，第二个是该月的日期码，日从 0(星期一) 到 6(星期日)，月从 1 到 12
prcal(year,w=2,l=1,c=6)	相当于 print(calendar.calendar(year,w,l,c))
prmonth(year,month,w=2,l=1)	相当于 print(calendar.calendar(year,w,l,c))
setfirstweekday(weekday)	设置每周的起止日期码，0(星期一) 到 6(星期日)
timegm(tupletime)	接受一个时间组，返回该时刻的时间戳 (1970 纪元后经历的浮点秒数)
weekday(year,month,day)	返回给定日期的日期码，0(星期一) 到 6(星期日)，月从 1(一月) 到 12(12 月)

程序举例：打印月历及日期码。

利用 calendar 库中的函数返回给定日期的日期码。0(星期一) 到 6(星期日)，月份为 1(一月) 到 12(12 月)。

其代码如下：

```
import calendar
# 返回指定年的某月
def get_month(year,month):
    return calendar.month(year,month)
# 返回指定年的日历
def get_calendar(year):
    return calendar.calendar(year)
```

任务编号	10-3	任务名称	模拟时钟

```
# 判定某年是否为闰年
def is_leap(year):
    return calendar.isleap(year)
# 返回某月 weekday 的第一天和这个月的所有天数
def get_month_range(year,month):
    return calendar.monthrange(year,month)
# 返回某月以每周为元素的序列
def get_month_calendar(year,month):
    return calendar.monthcalendar(year,month)

def main():
    year=2025
    month=4
    test_month=get_month(year,month)
    print(test_month)
    print('#'*50)
                                        #print(get_calendar(year))
    print('{0} 这一年是否为闰年 ?:{1}'.format(year,is_leap(year)))
    print(get_month_range(year,month))
    print(get_month_calendar(year,month))
if __name__=='__main__':
    main()
```

运行结果如图 10-19 所示。

```
     April 2025
Mo Tu We Th Fr Sa Su
    1  2  3  4  5  6
 7  8  9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30

##################################################
2025这一年是否为闰年?:False
(calendar.TUESDAY, 30)
[[0, 1, 2, 3, 4, 5, 6], [7, 8, 9, 10, 11, 12, 13], [14, 15, 16, 17, 18, 19, 20], [21,
 22, 23, 24, 25, 26, 27], [28, 29, 30, 0, 0, 0, 0]]
```

图 10-19　打印月历及日期

任务实践：模拟时钟

钟表是一种计时装置，其样式千变万化，但用来显示时间的表盘却相差无几。对于指针式钟表的表盘一般是由刻度、时针、分针和秒针、星期显示、日期显示组成，如图 10-20 所示。

任务编号	10-3	任务名称	模拟时钟

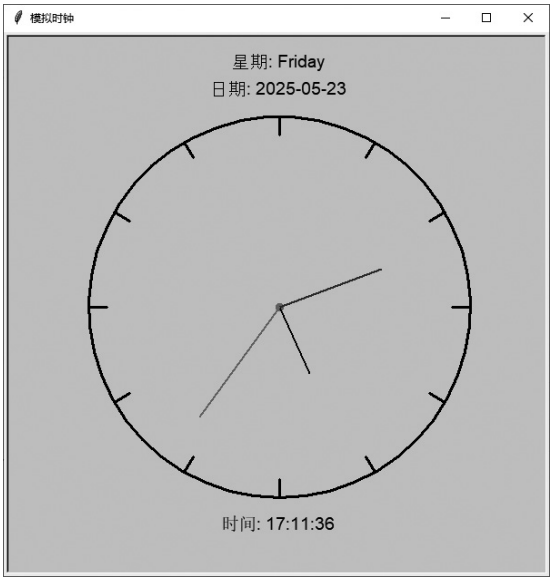

图 10-20　指针式钟表表盘

如图 10-20 所示的表盘有三根指针：时针、分针、秒针。它们的一端被固定在表盘中心，另一端可以沿顺时针方向进行旋转。表盘中最顶端的刻度为 12，它是所有指针的起始点，指针按顺时针刻度依次是 1,2,3,…,59。这里，秒针旋转一周，分针移动一个刻度，同样，分针移动一周，时针移动一格刻度。

本任务要求编写程序，使用 turtle 模块绘制一个如图 10-20 所示的表盘。使用 datetime 模块控制时钟的动态显示，即日期、星期、时间跟随本地时间实时变化。

打开 PyCharm，在右侧的工具栏中单击“AI 聊天”图标，即打开插件 AI Assistant，如图 10-21 所示。

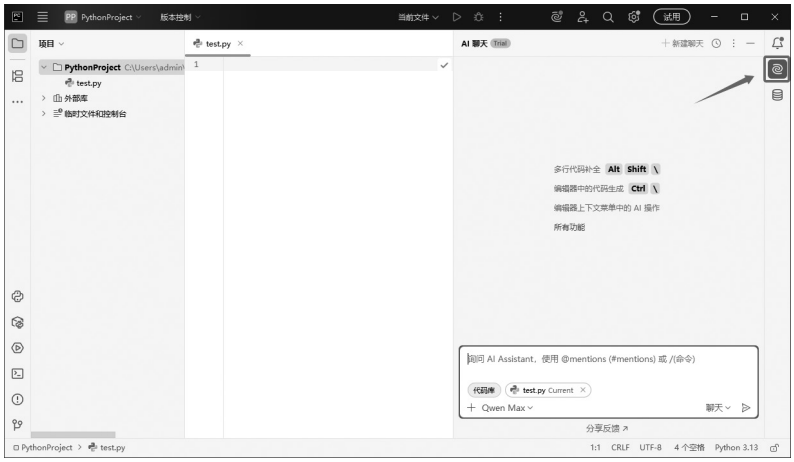

图 10-21　AI Assistant 聊天界面

任务编号	10-3	任务名称	模拟时钟

输入提示词：

假设你是一个有着 20 年编程经验的程序员主管，同时精通 Python 技术，现在帮我写一个 Python 程序，使用 turtle 模块绘制表盘，使用 datetime 模块控制时钟的动态显示，即日期、星期、时间跟随本地时间实时变化的模拟时钟。

随后在 AI Assistant 中就会生成初始代码，如图 10-22 所示。

图 10-22　AI Assistant 生成的模拟时钟代码

进行二次修改后的代码如下：

```
# 模拟时钟程序

import turtle
import datetime
import time

# 设置画布
screen = turtle.Screen()
screen.title(" 模拟时钟 ")
screen.bgcolor("lightblue")
screen.setup(width=600, height=600)
screen.tracer(0)
```

任务编号	10-3	任务名称	模拟时钟

```
# 创建时钟表盘
clock = turtle.Turtle()
clock.hideturtle()
clock.speed(0)
clock.pensize(3)

# 绘制时钟表盘
def draw_clock():
    # 绘制外圆
    clock.penup()
    clock.goto(0, -210)
    clock.pendown()
    clock.circle(210)

    # 绘制时钟刻度
    for i in range(12):
        clock.penup()
        clock.goto(0, 0)
        clock.setheading(90 + i * 30)
        clock.forward(190)
        clock.pendown()
        clock.forward(20)

    # 绘制时钟中心点
    clock.penup()
    clock.goto(0, 0)
    clock.dot(10, "red")

# 绘制时钟指针
def draw_hand(length, angle, color):
    hand = turtle.Turtle()
    hand.hideturtle()
    hand.speed(0)
    hand.pensize(2)
    hand.pencolor(color)
    hand.penup()
    hand.goto(0, 0)
    hand.setheading(90 - angle)
    hand.pendown()
    hand.forward(length)
    return hand

# 更新时钟显示
def update_clock():
```

任务编号	10-3	任务名称	模拟时钟

```
        # 获取当前时间
        now = datetime.datetime.now()

        # 绘制时钟指针
        hour_angle = (now.hour % 12 + now.minute / 60) * 30
        minute_angle = (now.minute + now.second / 60) * 6
        second_angle = now.second * 6

        # 清除之前的指针
        for hand in hands_and_text:
            hand.clear()

        # 绘制新的指针
        hour_hand = draw_hand(80, hour_angle, "black")
        minute_hand = draw_hand(120, minute_angle, "blue")
        second_hand = draw_hand(150, second_angle, "red")

        # 显示日期和星期
        date_display = turtle.Turtle()
        date_display.hideturtle()
        date_display.penup()
        date_display.goto(0, 230)
        date_display.write(f" 日期 : {now.strftime('%Y-%m-%d')}", align="center", font=("Arial", 14, "normal"))

        weekday_display = turtle.Turtle()
        weekday_display.hideturtle()
        weekday_display.penup()
        weekday_display.goto(0, 260)
        weekday_display.write(f" 星期 : {now.strftime('%A')}", align="center", font=("Arial", 14, "normal"))

        time_display = turtle.Turtle()
        time_display.hideturtle()
        time_display.penup()
        time_display.goto(0, -250)
        time_display.write(f" 时间 : {now.strftime('%H:%M:%S')}", align="center", font=("Arial", 14, "normal"))

        # 更新画布
        screen.update()

        # 返回新的指针和显示文本对象
        return [hour_hand, minute_hand, second_hand, date_display, weekday_display, time_display]

# 主循环
hands_and_text = []
# 绘制时钟表盘
```

任务编号	10-3	任务名称	模拟时钟

```
draw_clock()
while True:
    hands_and_text = update_clock()
    # 等待 1 秒
    time.sleep(1)
```

以上代码创建了一个模拟时钟，具有以下特点：使用 turtle 模块绘制了时钟表盘和指针，使用 datetime 模块获取当前时间，动态显示当前日期、星期和时间，时钟指针根据当前时间实时更新，时针（黑色）、分针（蓝色）和秒针（红色）有不同的长度和颜色。运行这个程序后，会看到一个不断更新的模拟时钟，它会根据当前的本地时间进行动态显示。

运行结果如图 10-23 所示。

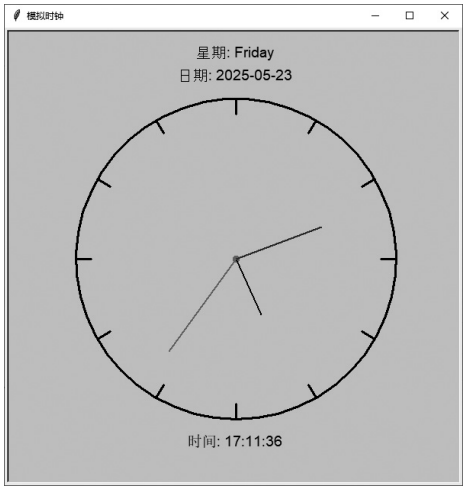

图 10-23　指针式钟表表盘

评价单

任务编号	10-3		任务名称	模拟时钟
评价项目			自评	教师评价
课堂表现	学习态度（20 分）			
	课堂参与（10 分）			
	团队合作（10 分）			
技能操作	创建 Python 文件（10 分）			
	编写 Python 代码（40 分）			
	运行并调试 Python 程序（10 分）			
评价时间	年　月　日		教师签字	

评价等级划分

项目		A	B	C	D	E
课堂表现	学习态度	在积极主动、虚心求教、自主学习、细致严谨上表现优秀	在积极主动、虚心求教、自主学习、细致严谨上表现良好	在积极主动、虚心求教、自主学习、细致严谨上表现较好	在积极主动、虚心求教、自主学习、细致严谨上表现尚可	在积极主动、虚心求教、自主学习、细致严谨上表现不佳
	课堂参与	积极参与课堂活动，参与内容完成的很好	积极参与课堂活动，参与内容完成的好	积极参与课堂活动，参与内容完成的较好	能参与课堂活动，参与内容完成的一般	能参与课堂活动，参与内容完成的欠佳
	团队合作	具有很强的团队合作能力，能与老师和同学进行沟通交流	具有良好的团队合作能力，能与老师和同学进行沟通交流	具有较好的团队合作能力，尚能与老师和同学进行沟通交流	能与团队进行合作，与老师和同学进行沟通交流能力一般	不能与团队进行合作，不能与老师和同学进行沟通交流
技能操作	创建	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	编写	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成
	运行并调试	能独立并熟练地完成	能独立并较熟练地完成	能在他人提示下顺利完成	能在他人帮助下完成	未能完成

项 目 总 结

本项目简单介绍了 Python 的计算生态、生态库的功能及应用、库的构建与发布、模块和库的概念、分类、导入及应用，常用内置库的使用方法。

通过本项目的学习，希望读者能够了解 Python 的计算生态及各生态库的功能和应用，掌握基于库的底层编程逻辑，并能够利用 AI 辅助编程的方法，并在此进行程序的二次开发及优化，具有能够解决复杂实际问题的能力。

巩 固 练 习

一、判断题

1．Python 开发人员可以使用内置库，也可以使用第三方库。（　　）

2．turtle 库中的 goto() 函数可以进行绘图前的起点定位。（　　）

3．Python 程序中使用内置库与第三方库的方式相同，但使用第三方库之前需要先将库导入程序。（　　）

4．自定义库只能由自己在本地使用。（　　）

5．Time 模块是 Python 的内置模块，可以在程序中直接使用。（　　）

二、选择题

1．阅读下面的程序：

```
gmtime = time.gmtime()
time.asctime(gmtime)
```

下列选项中，可以为以上程序输出结果的是（　　）。

A．'Mon Apr 16 08:15:35 2023'

B．time.struct_time(tm_year=2023,tm_mon=5,tm_mday=5,tm_hour=11,tm_min=6,tm_sec=36,tm_wday=8,tm_yday=113,tm_isdst=-1)

C．'11:07:23'

D．2283570376.3155279

2．下列选项中，用于判断“.py”文件是作为脚本执行还是被导入其他程序的属性是（　　）。

A．__init__　　B．__name__

C．__exce__　　D．__main__

3．下列方法中，返回结果是时间戳的是（　　）。

A．time.sleep()　　B．time.localtime()

C．time.strftime()　　D．time.ctime()

4．下列选项中，会在发布自定义库时用到的命令是（　　）。

A．python setup.py install　　B．python setup.py sdist

C．python setup.py build　　D．以上全部

5．阅读下面程序：

```
random.randrange(1,10,2)
```

下列选项中，不可能为以上程序输出结果的是（　　）。

A．1　　B．4　　C．7　　D．9

三、填空题

1．通过 random 库的 _____ 函数生成随机整数。

2．random 是 Python 的 _____ 库，Pandas 是 _____ 库。

3．Python 的计算生态通过 _____、_____、_____ 库为数据分析领域提供支持。

4．_____ 是一种按照一定规则自动从网上抓取信息的程序或者脚本。

5．turtle 库的主要作用是 _____。

四、程序设计题

1．编写程序，绘制奥运五环。

2．编写程序，随机生成登录的验证码。

3．编写程序，生成 2026 年的电子日历。