北 京 林 业 大 学

2022 学年—2023 学年第 2 学期 Python 应用 实验报告书

专　　业：大数据　　班　　级：大数据 212

姓　　名：余睿捷　　学　　号：211002328

实验地点：机房 N07　　任课教师：王春玲

实验题目：实验 2 tkinter 小学数学口算题生成器、电子时钟

实验环境：Python、PyCharm 、VSC 等

一、 实验目的

（1）tkinter 小学数学口算题生成器

1．熟悉 Python 标准库 tkinter 创建 GUI 应用程序的方法和步骤。
2．熟练安装 Python 扩展库 python-docx。
3．熟悉 Python 扩展库 pythondocx 操作 Word 文档的方法。
4．了解使用 Python 扩展库 python-docx 在 Word 文件中创建表格并写人数据的方法。
5．了解小学生各年级数学知识的学习程度和口算题目要求。
6．熟练使用 Python 标准库 random 中的函数。
7．熟练使用 Python 标准库 os 中的函数。

（2）tkinter 电子时钟

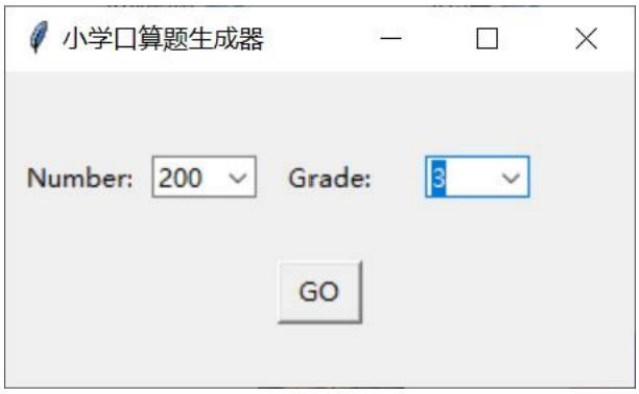
1．熟练使用 tkinter 创建窗体并设置窗体属性。
2．熟练使用 tkinter 创建标签组件并设置属性。
3．了解多线程编程的基础知识。
4．熟悉为窗口组件绑定鼠标事件的方法。

二、 实验内容

（1）tkinter 小学数学口算题生成器

在小学一、二年级，只能口算 20 以内整数的加、减法；三、四年级可以口算超过 20 的整数四则运算；五年级以上可以口算带括号的式子。

编写程序，批量生成小学口算题，要求把生成的口算题写入 Word 文件中的表格。表格共 4 列，用户指定表格行数和题目适用年级。程序运行后界面如下图所示。



程序运行界面

（2）tkinter 电子时钟

编写程序,实现如下图所示的电子时钟。要求：

（1）不显示标题栏，总是顶端显示，不被其他窗口覆盖；

（2）实时显示日期和时间；

（3）可以用鼠标左键按住拖动，在电子时钟上右击可以结束程序；

（4）拖动时透明度变大，鼠标左键抬起时恢复半透明状态。

三、 实验步骤及结果

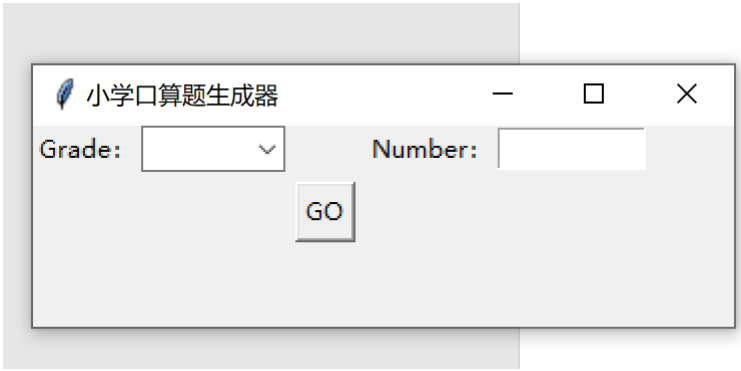
（1）tkinter 小学数学口算题生成器

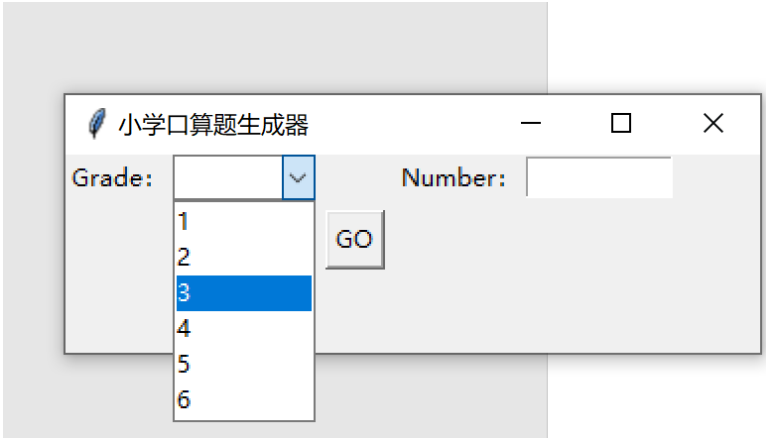
```
# (1)tkinter 小学数学口算题生成器

import tkinter as tk
from tkinter import ttk
import os
import random
from docx import Document

# 创建题目生成器类
class MathQuizGenerator:
    def __init__(self, master):
        # 初始化主窗口
        self.master = master
        self.master.title("小学口算题生成器")  # 设置窗口标题
        self.master.geometry("350x100")  # 设置窗口大小
        self.create_widgets()  # 调用创建控件的方法

    def create_widgets(self):
        # 创建下拉菜单和输入框等控件
        grade_label = tk.Label(self.master, text="Grade: ")  # 创建标签控
件，显示年级
```

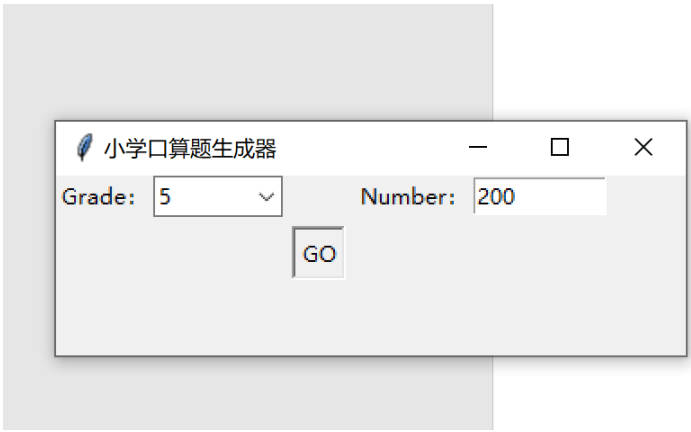
```
        options = ["1", "2", "3", "4", "5", "6"]  # 创建下拉菜单的选项
        self.grade_combo = ttk.Combobox(self.master, values=options,
width=7)  # 创建下拉菜单控件，显示选项

        num_label = tk.Label(self.master, text="Number: ")  # 创建标签控件，
显示题目数量
        self.num_entry = tk.Entry(self.master, width=10)  # 创建文本框控件，
输入题目数量

        grade_label.grid(row=0, column=0)  # 将年级标签控件放置在窗口中的第
一行第一列
        self.grade_combo.grid(row=0, column=1)  # 将下拉菜单控件放置在窗口
中的第一行第二列
        num_label.grid(row=0, column=4)  # 将题目数量标签控件放置在窗口中的
第一行第五列
        self.num_entry.grid(row=0, column=5)  # 将文本框控件放置在窗口中的
第一行第六列

        # 创建生成题目按钮
        self.generate_button = tk.Button(self.master, text="GO",
command=self.generate_quiz)  # 创建按钮控件，点击生成题目
        self.generate_button.grid(row=1, column=2, padx=5, pady=5)  # 将
按钮控件放置在窗口中的第二行第三列，并添加一些内边距

    def generate_quiz(self):
        # 获取用户选择的年级、题目数量和输出格式
        grade = int(self.grade_combo.get())  # 获取下拉菜单中选择的年级
        num = int(self.num_entry.get())  # 获取文本框中输入的题目数量

        # 根据年级设置运算符和数字范围
        if grade < 3:
            operators = '+-'  # 一年级和二年级只有加减法
            Max = 20  # 数字范围在 20 以内
        elif grade <= 4:
            operators = '＋－×÷'  # 三年级和四年级有加减乘除法
            Max = 100  # 数字范围在 100 以内
        elif grade <= 6:
            operators = '＋－×÷('  # 五年级和六年级加减乘除和括号都有
            Max = 100  # 数字范围在 100 以内

        # 创建一个新的 word 文档对象
        document = Document()
        # 创建一个表格，行数为题目数量除以 4，列数为 4
        table = document.add_table(rows=num//4, cols=4)
```

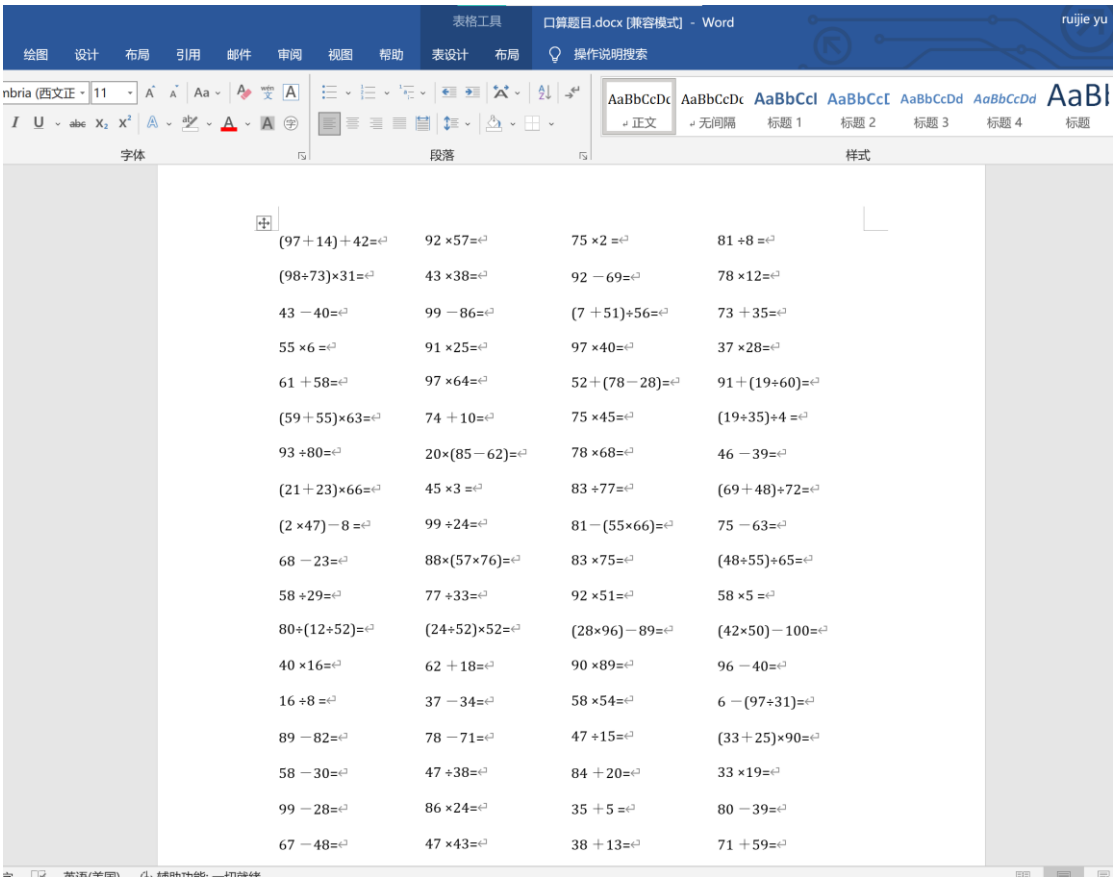
```
        # 循环生成题目并填充到表格中
        for row in range(num//4):
            for col in range(4):
                # 随机生成两个数和一个运算符
                first = random.randint(1,Max)
                second = random.randint(1,Max)
                operator = random.choice(operators)

                if operator != '(':  # 如果不是五年级
                    if operator == '-'or'÷':    # 如果是减法或除法，确保第一个数大于等于第二个数
                        if first < second:
                            first,second = second,first
                    # 格式化题目字符串
                    r = str(first).ljust(2,' ')+' ' + operator + str(second).ljust(2,' ') + '='

                else:  # 如果是五年级
                    third = random.randint(1,100)  # 随机生成第三个操作数
                    while True:  # 随机生成两个操作符
                        o1 = random.choice(operators)
                        o2 = random.choice(operators)
                        if o1 != '(' and o2 !='(':
                            # 如果两个操作符均不为左括号，则跳出循环
                            break
                    # 考虑括号的口算题
                    r2 = random.randint(1,100)
                    if r2 > 50:
                        if o2 == '-':  # 如果第二个操作数小于第三个操作数，则交换它们的值
                            if second < third:
                                second,third = third,second
                        # 格式化题目字符串
                        r = str(first).ljust(2,' ') + o1 +'('+str(second).ljust(2,' ')+o2+str(third).ljust(2,' ')+')='
                    else:
                        if o1 == '-':
                            if first < second:  # 如果第一个操作数小于第二个操作数，则交换它们的值
                                first,second = second,first
                        # 格式化题目字符串
                        r = '(' + str(first).ljust(2,' ') + o1 +str(second).ljust(2,' ')+')'+o2+str(third).ljust(2,' ')+'='
```

```
                # 将题目写入表格单元格
                cell = table.cell(row,col)
                cell.text = r

        # 保存并打开文档
        document.save('口算题目.docx')
        os.startfile("口算题目.docx")



if __name__ == '__main__':
    root = tk.Tk()
    app = MathQuizGenerator(root)
    root.mainloop()
```









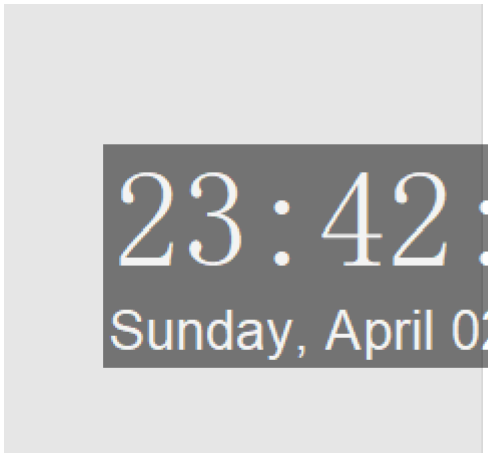
(97+14)+42=	92 ×57=	75 ×2 =	81 ÷8 =
(98÷73)×31=	43 ×38=	92 −69=	78 ×12=
43 −40=	99 −86=	(7 +51)÷56=	73 +35=
55 ×6 =	91 ×25=	97 ×40=	37 ×28=
61 +58=	97 ×64=	52+(78−28)=	91+(19÷60)=
(59+55)×63=	74 +10=	75 ×45=	(19÷35)÷4 =
93 ÷80=	20×(85−62)=	78 ×68=	46 −39=
(21+23)×66=	45 ×3 =	83 ÷77=	(69+48)÷72=
(2 ×47)−8 =	99 ÷24=	81−(55×66)=	75 −63=
68 −23=	88×(57×76)=	83 ×75=	(48÷55)÷65=
58 ÷29=	77 ÷33=	92 ×51=	58 ×5 =
80÷(12÷52)=	(24÷52)×52=	(28×96)−89=	(42×50)−100=
40 ×16=	62 +18=	90 ×89=	96 −40=
16 ÷8 =	37 −34=	58 ×54=	6 −(97÷31)=
89 −82=	78 −71=	47 ÷15=	(33+25)×90=
58 −30=	47 ÷38=	84 +20=	33 ×19=
99 −28=	86 ×24=	35 +5 =	80 −39=
67 −48=	47 ×43=	38 +13=	71 +59=

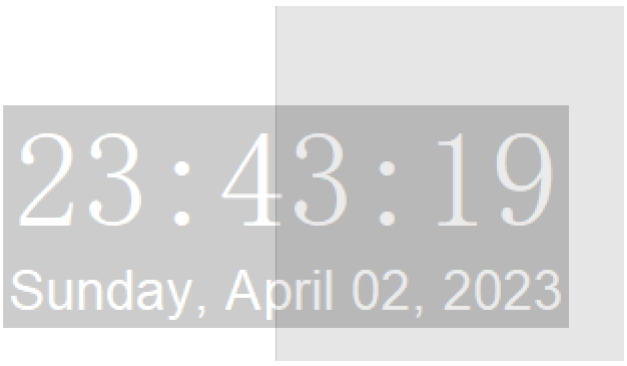
（2）tkinter 电子时钟

```
#（2）tkinter 电子时钟
import tkinter as tk
import time

class Clock:
    def __init__(self, master):
        # 初始化窗口
        self.master = master
        #self.master.wm_attributes('-type', '_NET_WM_WINDOW_TYPE_DOCK')
```

```
        self.master.overrideredirect(True)  # 不显示标题栏
        self.master.geometry("+0+0")  # 位置在屏幕左上角
        self.master.attributes('-topmost', True)  # 窗口始终在顶层
        self.master.config(bg='black')  # 窗口背景色为黑色
        self.master.attributes('-alpha', 0.5)  # 窗口透明度初始值为 0.5

        # 绑定鼠标事件
        self.master.bind("<Button-3>", self.quit)  # 鼠标右键点击退出程序
        self.master.bind("<ButtonPress-1>", self.click)   # 鼠标左键按下，
开始移动窗口
        self.master.bind("<ButtonRelease-1>", self.release)  # 鼠标左键松
开，停止移动窗口
        self.master.bind("<B1-Motion>", self.drag)          # 鼠标左键拖
动，移动窗口

        # 创建标签组件用于显示时间和日期
        self.time_label = tk.Label(master, font=('Digital-7', 50),
fg='white', bg='black')
        self.time_label.pack(fill='both', expand=True)

        self.date_label = tk.Label(master, font=('Helvetica', 20),
fg='white', bg='black')
        self.date_label.pack(side='bottom', fill='x')

        self.tick()  # 开始时钟

    def tick(self):
        # 更新时间和日期标签内容
        current_time = time.strftime('%H:%M:%S')
        self.time_label.config(text=current_time)
        current_date = time.strftime('%A, %B %d, %Y')
        self.date_label.config(text=current_date)
        # 每秒调用一次 tick 函数更新时间和日期
        self.master.after(1000, self.tick)

    def click(self, event):
        # 鼠标左键点击时，记录鼠标按下时的坐标,将窗口透明度设置为 0.2
        self.x = event.x
        self.y = event.y
        self.master.attributes('-alpha', 0.2)

    def drag(self, event):
        # 计算窗口移动后的新位置，并移动窗口
```

```
        self.master.geometry(f'+{event.x_root - self.x}+{event.y_root -
self.y}')

    def release(self, event):
        # 鼠标左键释放时，停止移动窗口,将窗口透明度设置为0.5
        self.master.attributes('-alpha', 0.5)

    def quit(self, event):
        # 鼠标右键点击时退出程序
        self.master.quit()
        self.master.destroy()


root = tk.Tk()
clock = Clock(root)
root.mainloop()
```





（此截图截不到鼠标，此时

鼠标正在拖动电子时钟，使透明度下降）

四、 实验分析

问题 1：一直无法出现 GUI 界面，错误的类型显示是 AttributeError ，但经过检查确定 tkinter 已经成功调入。

解决方法：在 def __init__(self, master)中补上 master.mainloop()用来启动 tkinter 的主循环。

问题 2：in __init__ self.num_entry.geometry("50x10") AttributeError: 'Entry' object has no attribute 'geometry'。

解决方法：这部分代码中的错误是 Entry 对象没有 geometry 属性。geometry 是窗口对象的一个属性，用于设置窗口的大小和位置，而不是控件对象的属性。可以使用 width 和 height 属性来设置， self.num_entry = tk.Entry(self.master, width=10)

问题 3：grade = self.grade_combo.get()；AttributeError: 'MathQuizGenerator' object has no attribute 'grade_combo'

解决方法：要正确导入 ttk 库，并且注意哪里要加 self，哪里不用。grade = int(self.grade_combo.get())

问题 4：一开始写成 num/4，造成文档输出不对。

解决方法：改为 num//4

问题 5：在初始化时使用 self.opacity = 0.5 来设置窗口透明度初始值为 0.5，但代码由于 overrideredirect 属性的存在，导致设置透明度无效。

解决方法：一开始想用 self.master.wm_attributes('-type', '_NET_WM_WINDOW_TYPE_DOCK')解决问题，但出现报错，因为-type 属性通常只在 Linux 系统下可用，而在 Windows 和 macOS 上不可用。最后使用了 self.master.attributes('-alpha', 0.5)来使窗口初始透明度为 0.5。

问题 6： 一开始在设置拖动窗口时，直接简单的当前窗口的 x 坐标和 y 坐标加上鼠标位置，造成窗口拖动时自动变到鼠标右下角，有一种漂移的既视感。

```
        x = self.master.winfo_x() + event.x
        y = self.master.winfo_y() + event.y
        self.master.geometry("+%s+%s" % (x, y))
```

解决方法：先确定鼠标位置，再通过 self.x 和 self.y 保存了鼠标左键按下时的坐标值，用 event.x_root 和 event.y_root 记录当前鼠标指针所在的屏幕上的绝对坐标值。通过计算这两者的差值，得到了鼠标移动的距离，从而实现了窗口的移动。

```
self.master.geometry(f'+{event.x_root - self.x}+{event.y_root - self.y}')
```

收获：这两个实验的实验收获包括以下方面：

对 GUI 应用程序的开发有了更深入的了解，掌握了使用 tkinter 库创建窗

口、标签、按钮等组件的方法。

掌握了 Python 扩展库 python-docx 操作 Word 文档的方法，了解了如何使用该库在 Word 文件中创建表格、写入数据等操作。

对多线程编程有了初步的了解，能够使用线程来实现一些特定的功能。

通过编写实验代码，熟悉了 Python 标准库 random 和 os 中的一些函数的使用方法。

了解了小学生各年级数学知识的学习程度和口算题目要求，为以后的教学或者科普工作提供了一定的帮助。

在实践中不断调试代码，解决错误和问题的能力得到提升，对编程有了更深入的理解和认识。