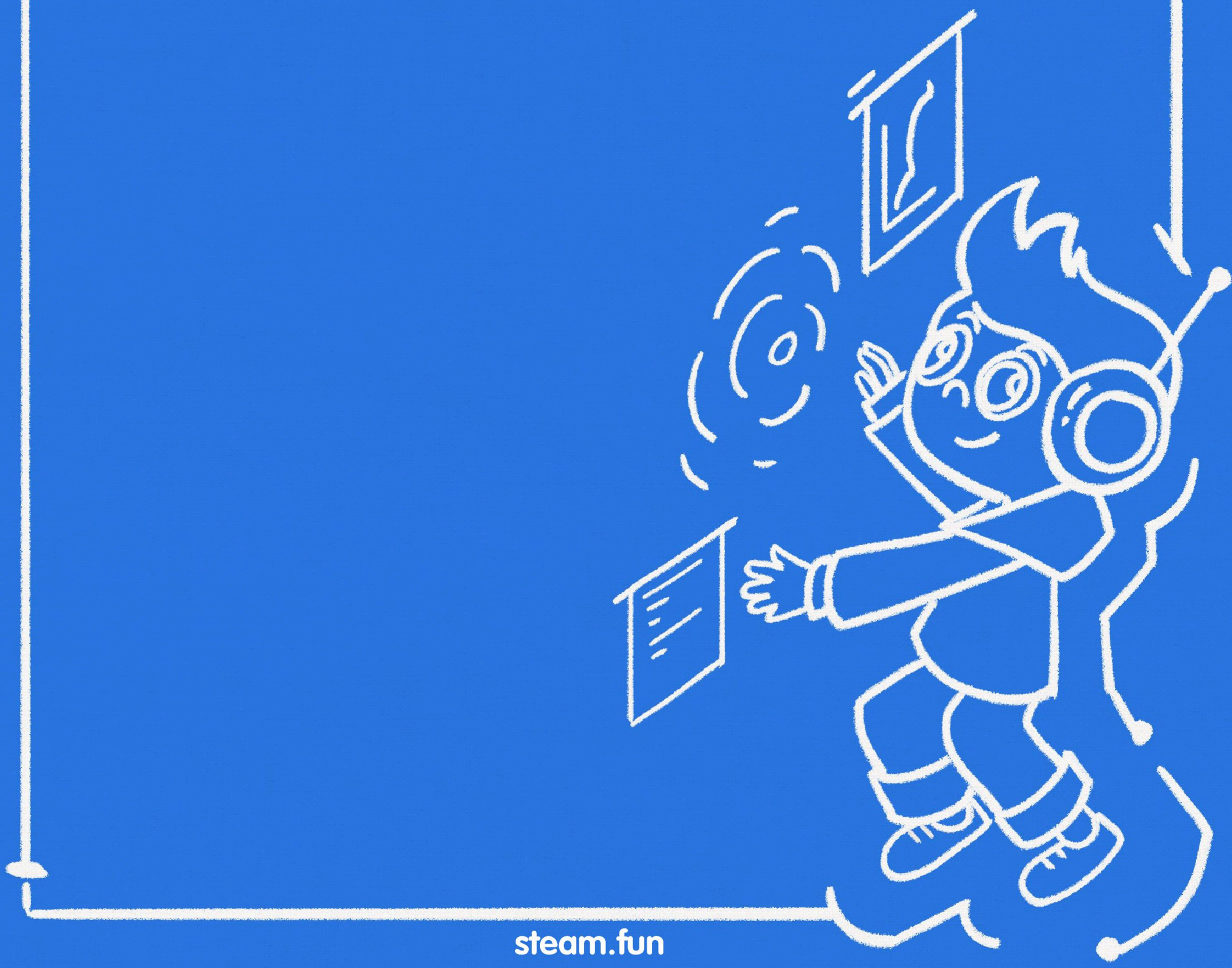
入会考验



1.探索新知

1.1 窗体出现

1. 导入 tkinter 库

```
import tkinter as tk
```

2. 创建主窗口对象，并起名为root

```
root = tk.Tk()
```

3. 配置窗口标题

```
root.title("大聪明的时钟")
```

4. 设置窗口大小

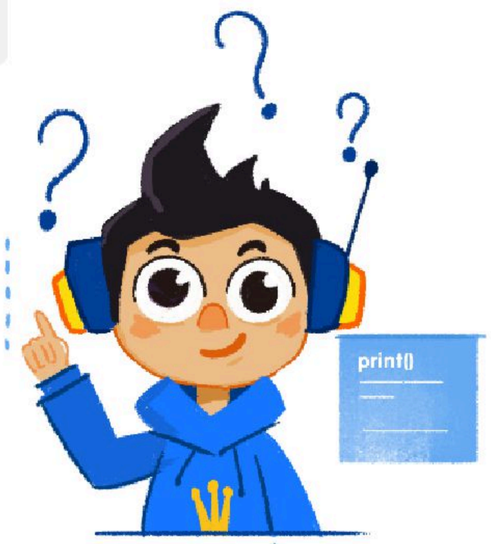
```
root.geometry('400x80+400+400')
```

5. 设置窗口不可调整

```
root.resizable(width=False, height=False)
```

【resizable】：
可调整大小

宽度和高度 False
代表什么意思？



1.1 窗体出现

指令手册

窗口.resizable(width, height)

- Tkinter 库中的一个方法，用于设置窗口是否可调整大小
- resizable() 方法接受两个布尔值参数：width 和 height。分别表示窗口是否可以在水平方向和垂直方向上调整大小
- 如果参数为 True ，则允许在该方向上调整大小；如果为 False ，则禁止在该方向上调整大小

6. 让窗口活起来

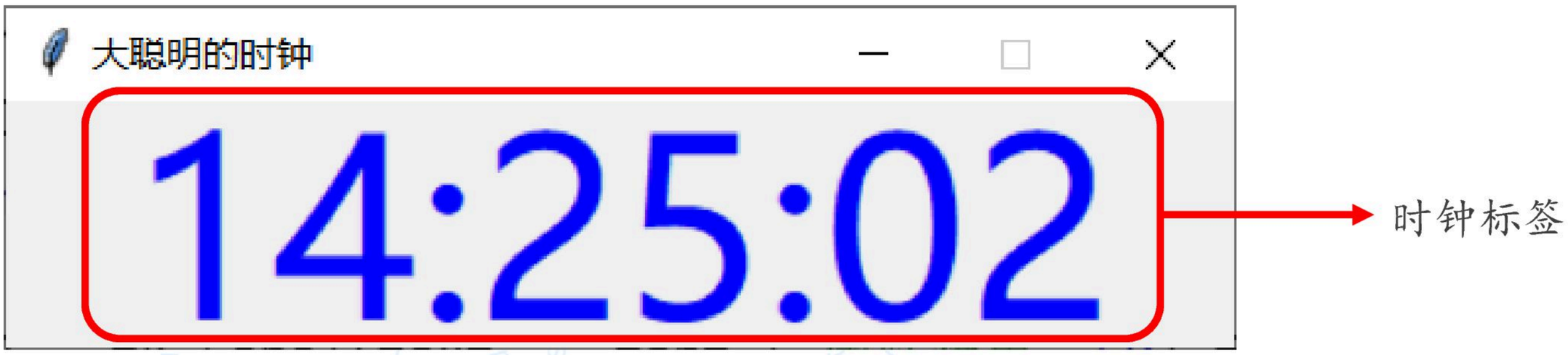
```
root.mainloop()
```

效果如下：

任务一新增代码如下：

```
import tkinter as tk
root = tk.Tk()
root.title('大聪明的时钟')
root.geometry('400x80+400+400')
root.resizable(width=False, height=False)
root.mainloop()
```

1.2 创建时钟标签



还记得怎么在窗口中创建标签吗？

【标签的文本内容】
时间是不停更新的，我们先设置文本为'666'测试使用，后续记得删除

【文本颜色】
这里用的是蓝色，也可以设置其他颜色

```
label = tk.Label(text='666', font=('微软雅黑',60), fg='blue')
```

【font】：
字体

【文本的字体、字号】
这里用的是“微软雅黑”字体，字号是60

```
label.pack()
```

任务二新增代码如下：

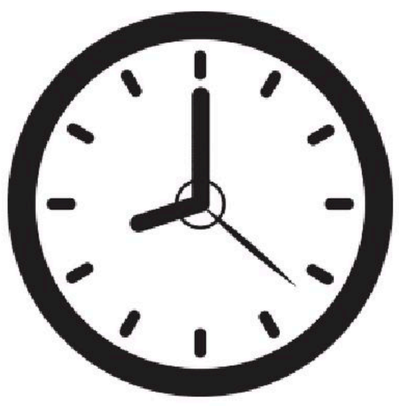
```
label = tk.Label(text='666', font=('微软雅黑',60), fg='blue')
label.pack()
```

1.3 时间更新

标签准备好后，接下来就需要对时间进行更新啦！我们创建一个"update_clock"函数，来实现更新时间的功能

```
def update_clock():
    pass
update_clock()
```

可以发现，时间在不停更新，要求与计算机时间一致，该如何获取本地时间呢？



time 库

time库是Python中处理时间的标准库，是最基础的时间处理库。

1. 导入time库

```
import time
```

2. 时间格式化

```
now = time.strftime('%H:%M:%S')
```

1.3 时间更新

指令手册

time.strftime(format[, t])

- format 是一个格式化模板字符串，必须提供，用来定义输出效果；t 表示要格式化的时间元组，如果不提供，则默认使用当前时间
- %H　24小时制的小时
- %M　分钟
- %S　秒

下面是一个例子：

```
# 提供 format 和 t 参数，使用指定的时间元组
now = time.localtime()  # 获取当前时间的时间元组
formatted_time = time.strftime("%H:%M:%S", now)
print(formatted_time)
```

如何将格式化后的时间字符串放到窗口中呢？


```
label.config(text=now)
```

1.3 时间更新

```
def update_clock():
    now = time.strftime('%H:%M:%S')
    label.config (text=now)
update_clock()
```

虽然能够在窗口显示时间了，不过显示的是固定的时间，如何让时间更新呢？

我有一个好主意，我在【更新时间】的函数中再次调用【更新时间】的函数，就跟套娃一样，不停递归就好了嘛！

不停递归，不会停止！

```
def update_clock():
    now = time.strftime('%H:%M:%S')
    label.config(text=now)
    update_clock()

update_clock()
```

1.3 时间更新

控制台

```
  File "C:\Program Files (x86)\SteamCode\resources\app\python-env\win\l
    return self._configure('configure', cnf, kw)
  File "C:\Program Files (x86)\SteamCode\resources\app\python-env\win\l
    cnf = _cnfmerge((cnf, kw))
  File "C:\Program Files (x86)\SteamCode\resources\app\python-env\win\l
    elif isinstance(cnfs, (type(None), str)):
RecursionError: maximum recursion depth exceeded in __instancecheck__
```

意思是“超出以下条件的最大递归深度”，总之，就是递归太快太深，电脑脑子不够用啦，出故障瓦特啦(●ˇ∀ˇ●)，看来这个方法不行，怎么办呢？

这里需要一个新的指令来帮忙，每隔一段时间调用一次更新时间的函数，这样电脑就能愉快地执行，不会脑子瓦特啦！

指令手册

窗口名.after(延迟时间，延迟后调用函数)

- Tkinter 中的一个方法，用于在指定的时间后调用一个函数
- 延迟时间，单位是毫秒。1秒=1000毫秒

1.3 时间更新

```
def update_clock():
    now = time.strftime('%H:%M:%S')
    label.config(text=now)
    root.after(1000, update_clock)

update_clock()
```

- 每隔1000毫秒（也就是1秒），调用一次时间更新的函数

任务三新增代码如下：

```
import time
# 记得删除测试用的“666”，text为空效果更好
label = tk.Label(text='', font=('微软雅黑',60), fg='blue')
label.pack()
# 更新时间的函数
def update_clock():
    now = time.strftime('%H:%M:%S')
    label.config(text=now)
    # 每1秒（1000毫秒）调用一次update_clock函数
    root.after(1000,update_clock)
update_clock()
```

完整代码

```
# 导入tkinter、time库
import tkinter as tk
import time

# 创建主窗口
root = tk.Tk()
root.title('大聪明的时钟')
root.geometry('400x80+400+400')
root.resizable(width=False,height=False)

# 创建时间标签
label = tk.Label(text='',font=('微软雅黑',60),fg='blue')
label.pack()

# 定义更新时间的函数
def update_clock():
    now = time.strftime('%H:%M:%S')
    label.config(text=now)
    root.after(1000,update_clock)  # 每1秒（1000毫秒）调用一次update_clock函数

# 更新时间
update_clock()

# 运行主循环，等待用户交互
root.mainloop()
```

2.强化练习

1. root.geometry('400x80+400+400')中的400x80表示（　　）

A．窗口的宽度和高度　　B．窗口的左上角坐标

C．窗口的右下角坐标　　D．窗口的颜色代码

2. root.resizable(width=False, height=False)的作用是？（　　）

A．允许用户调整窗口的宽度和高度　　B．禁止用户调整窗口的宽度和高度

C．只能调整窗口的宽度　　D．只能调整窗口的高度

3. time.strftime('%H:%M:%S')中的%H表示（　　）

A．小时（24小时制）　　B．分钟

C．秒　　D．年份

4. root.after(1000,update_clock)中的1000表示（　　）

A．1秒　　B．1000秒

C．1毫秒　　D．10秒

5. label.config(text=now)的作用是（　　）

A．创建一个新的标签　　B．更新标签的文本内容

C．删除标签　　D．移动标签的位置

3.术语箱

resizable	可调整大小
font	字体
strftime	时间格式化为字符串
update	更新
clock	时钟

4.课后挑战

挑战一：

运用学到的知识对作品进行如下调整：

1. 设置窗口为宽度可调整，高度不可调整
2. 设置标签字体为“宋体”
3. 设置标签背景色

（黑、红、黄任选一）

黑色“black”，bg="black"

红色“red”，bg="red"

黄色“yellow”，bg="yellow"

挑战二：

运用学到的知识实现如视频所示效果：

提示：

1. 创建名言标签并配置在窗口中
2. 设置合适的窗口尺寸

（或者尝试使用默认参数）