星座之谜

1.探索新知

1.1 窗口搭建

1. 导入tkinter库

```
import tkinter as tk
```

2. 创建主窗口

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

3. 设置窗口尺寸、位置

注意：是英文字母x，不是乘号

```
root.geometry('320x360')
```

4. 设置窗口标题

```
root.title('测测你的星座吧')
```

窗口创建好后，
为什么迟迟不出现呢？

5. 启动事件循环
（放在所有代码的最后）

```
root.mainloop()
```

1.1 窗口搭建

任务一新增代码：

```
import tkinter as tk
root = tk.Tk()
root.geometry('320x360')
root.title('测测你的星座吧')
...
root.mainloop()
```

1.2 组件布局

介绍了tkinter中三种主要的布局管理器，接下来我们详细拆分，完成【组件布局】的任务

1. 【设置动态背景】
2. 【创建标签和文本框】
3. 【创建提交按钮】

【设置动态背景】

先创建一个标签

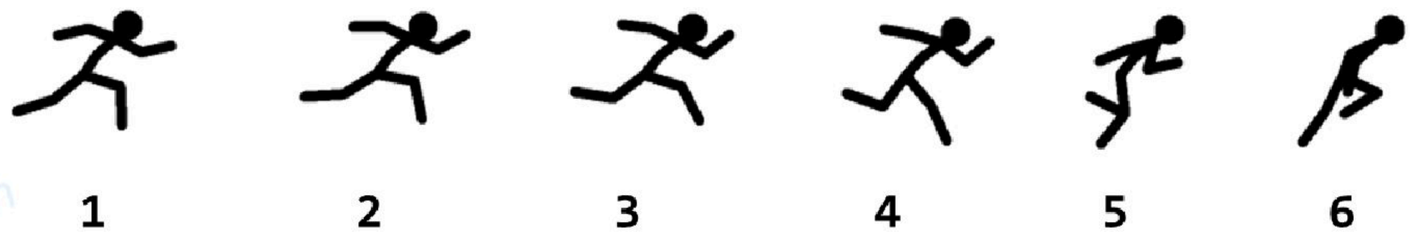
```
...
root.title("测测你的星座吧")
# 创建一个标签用于显示gif动画
label_bg = tk.Label(root)
# 将标签放在窗口顶部并充斥整个窗口
label_bg.place(x=0, y=0, relwidth=1, relheight=1)
```

标签左上角坐标

标签宽度和高度与窗口一致

1.2 组件布局

如何在标签中播放gif动画



观察上面火柴人跑步的gif图，
动图是由多帧图像组成的，那么想要在标签中播放gif动画，是否可以通过更新标签中显示的帧来实现呢？

接下来我们通过三步来实现效果：

1. 加载gif图
2. 将每帧图像存在frames列表中
3. 更新gif动画帧

1. 加载gif图

```
from PIL import Image
...
label_bg = tk.Label(root)
label_bg.place(x=0, y=0, relwidth=1, relheight=1)
image = Image.open('星图.gif')  # 打开gif图片文件
```

2. 将每帧图像存在frames列表中

想要在窗口中显示图像，必须把图片转化为PhotoImage的形式

```
from PIL import Image, ImageTk
...
image = Image.open('星图.gif')  # 打开gif图片文件
frames = []  # 用于存储gif的所有帧
while True:  # 将每一帧添加到frames列表中
    frames.append(ImageTk.PhotoImage(image))
    image.seek(len(frames))  # 跳转到gif的下一帧
```

【seek】：
寻找，搜索

运行看看效果吧！

运行后，报错信息如下：

```
EOFError: no more images in GIF file
EOFError: GIF 文档中不再有图像
```

gif图的多帧图像都已经被存在frames列表中，所以无法跳转到gif的下一帧，此时发生了报错。

```
try:
    while True:     # 将每一帧添加到frames列表中
        frames.append(ImageTk.PhotoImage(image))
        image.seek(len(frames))  # 跳转到gif的下一帧
except EOFError:
    pass    # 当到达gif结束时，跳出循环
```

gif结束时，报错EOFError，此时我们处理这个异常，让程序继续往下执行

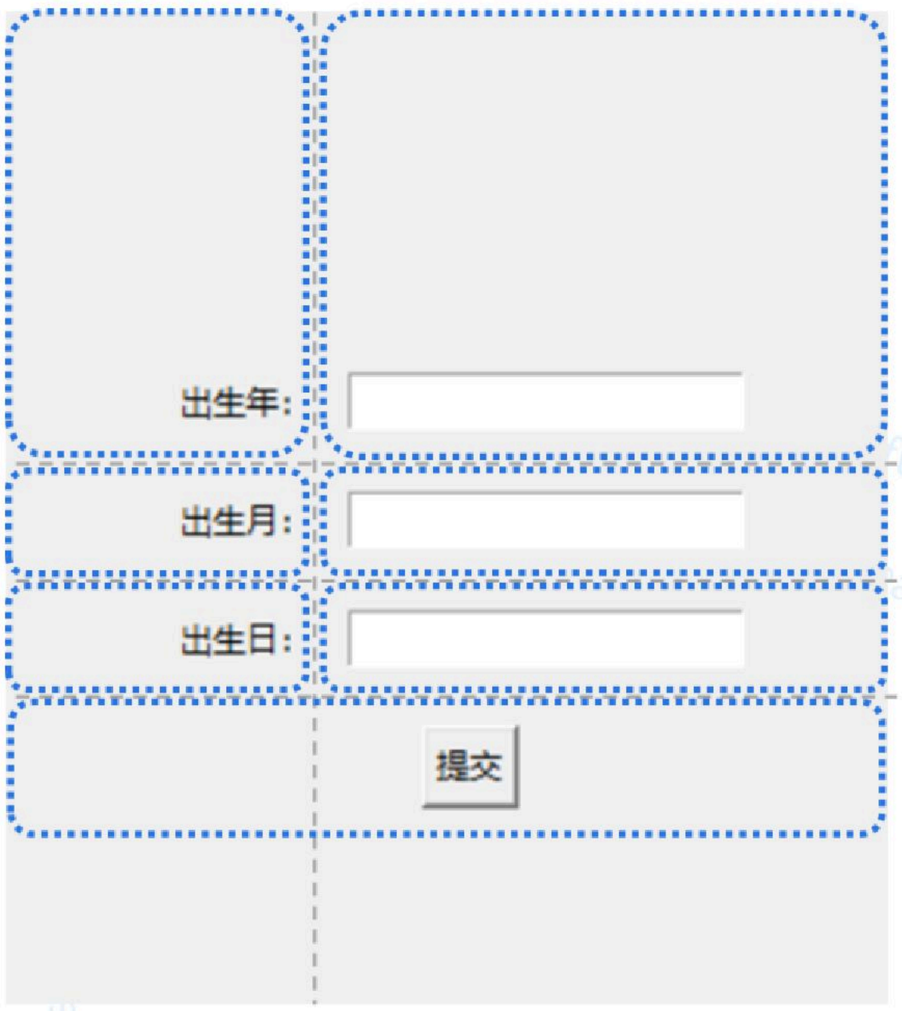
3. 更新gif动画帧

想要更新gif动画帧，也就是挨个取出frames列表中的图像，并依次更新到label_bg 标签中

```
def upd(idx):
    frame = frames[idx]  # 获取当前帧
    label_bg.configure(image=frame)  # 更新标签中显示的帧
    idx = (idx + 1) % len(frames)    # 计算下一帧的索引
    root.after(100, upd, idx)  # 100毫秒后调用自己继续更新帧
upd(0) # 开始播放gif动画
```

1.2 组件布局

【创建标签和文本框】

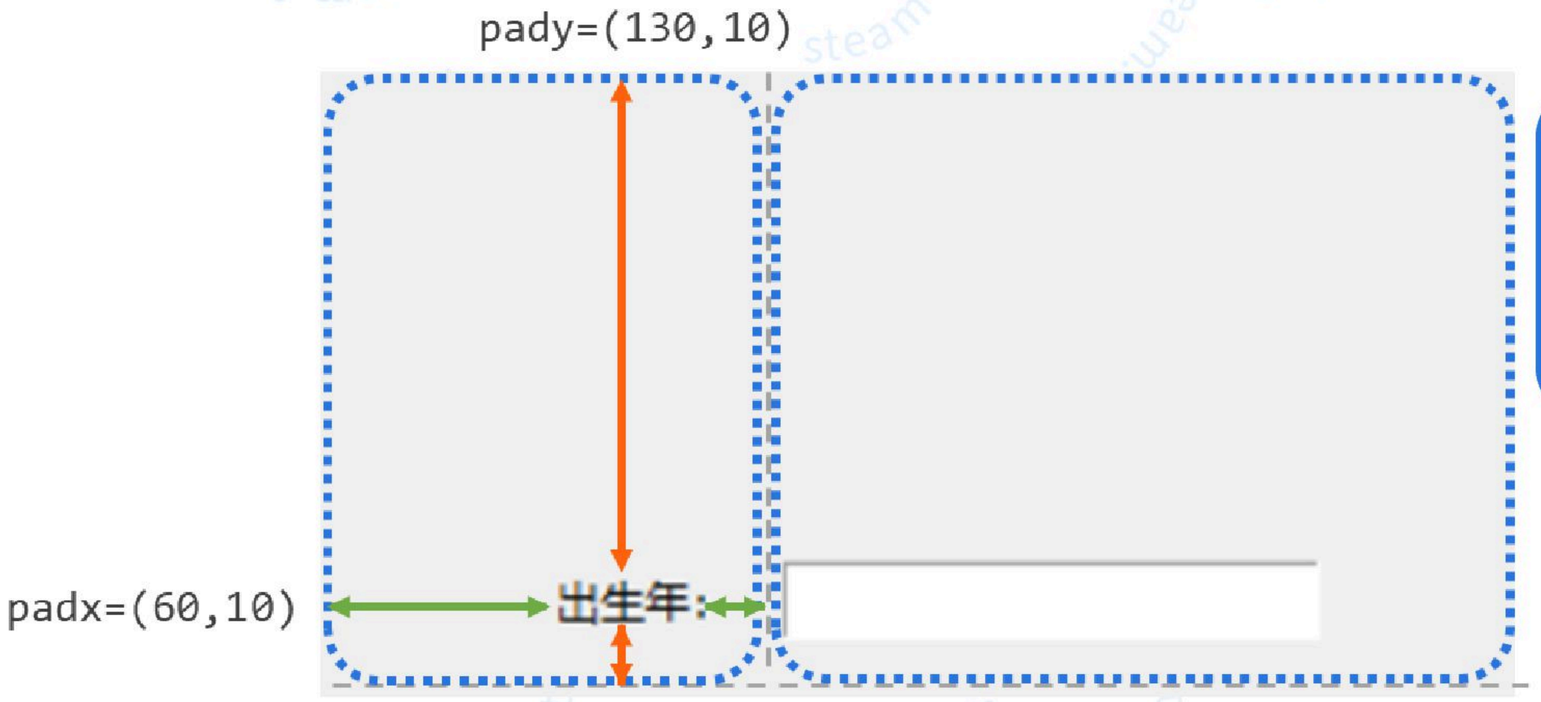


【entry】：
登记，进入

创建单行文本框

- 使用 tk.Entry(root) 创建单行文本框
- 使用 entry.get() 方法获取输入内容

标签我们创建过很多次，
那怎么创建文本框呢？



具体间距值，可
以自己尝试调整，
找到合适的间隔

```
...
upd(0)
label_year = tk.Label(root, text="出生年: ")
label_year.grid(row=0, column=0, padx=(60,10),
pady=(130,10))
entry_year = tk.Entry(root)
entry_year.grid(row=0, column=1, pady=(130,10))
```

1.2 组件布局

接着继续完成后面两行标签和文本框的配置：

```
label_month = tk.Label(root, text="出生月: ")
label_month.grid(row=1, column=0, padx=(60,10), pady=10)
entry_month = tk.Entry(root)
entry_month.grid(row=1, column=1, pady=10)

label_day = tk.Label(root, text="出生日: ")
label_day.grid(row=2, column=0, padx=(60,10), pady=(10,20))
entry_day = tk.Entry(root)
entry_day.grid(row=2, column=1, pady=(10,20))
```

【创建提交按钮】

```
...
entry_day.grid(row=2, column=1, pady=(10,20))
def click():
    pass  # 空函数，下个任务再补充
btn = tk.Button(root, text="提交", command=click)
btn.grid(row=3, columnspan=2, padx=(80,10))
```

自由调整设置合适的位置、间距

任务二新增代码：

```
from PIL import Image, ImageTk
...
label_bg = tk.Label(root) # 创建一个标签用于显示gif动画
label_bg.place(x=0, y=0, relwidth=1, relheight=1) # 标签充斥整个窗口
# 加载gif图片并将每一帧存储在frames列表中
image = Image.open('星图.gif')  # 打开gif图片文件
frames = []  # 用于存储gif的所有帧
```

```
try:
    while True:  # 将每一帧添加到frames列表中
        frames.append(ImageTk.PhotoImage(image))
        image.seek(len(frames))  # 跳转到gif的下一帧
except EOFError:
    pass  # 当到达gif结束时，跳出循环
def upd(idx):  # 用于更新gif动画帧的函数
    frame = frames[idx] # 获取当前帧
    label_bg.configure(image=frame) # 更新标签中显示的帧
    idx = (idx + 1) % len(frames)  # 计算下一帧的索引
    root.after(100, upd, idx)  # 100毫秒后调用自己继续更新帧
upd(0) # 开始播放gif动画

# 创建标签和文本框
label_year = tk.Label(root, text="出生年: ")
label_year.grid(row=0, column=0, padx=(60,10),
pady=(130,10))
entry_year = tk.Entry(root)
entry_year.grid(row=0, column=1, pady=(130,10))
label_month = tk.Label(root, text="出生月: ")
label_month.grid(row=1, column=0, padx=(60,10), pady=10)
entry_month = tk.Entry(root)
entry_month.grid(row=1, column=1, pady=10)
label_day = tk.Label(root, text="出生日: ")
label_day.grid(row=2, column=0, padx=(60,10),
pady=(10,20))
entry_day = tk.Entry(root)
entry_day.grid(row=2, column=1, pady=(10,20))

def click():
    pass  # 空函数，下个任务再补充
# 创建提交按钮
btn = tk.Button(root, text="提交", command=click)
btn.grid(row=3, columnspan=2, padx=(80,10))

root.mainloop()
```

1.3 点击按钮解锁星座

点击提交按钮后，需要获取用户输入的出生年月日，验证输入的内容是否有效，如果有效，则返回用户对应的星座

首先，还记得如何获取文本框输入的内容吗？

```
def click():
    year = entry_year.get()
    month = entry_month.get()
    day = entry_day.get()
```

获取的year、month、day是数字还是字符串呢？自己试试吧！
（测试完记得删除测试积木）

```
def click():
    year = entry_year.get()
    month = entry_month.get()
    day = entry_day.get()
    print(type(year)) # 测试用
```

控制台

```
<class 'str'>
```

如何确认用户输入的是否为有效的数字？

1996　　有效的数字

1996年　　不是有效的数字

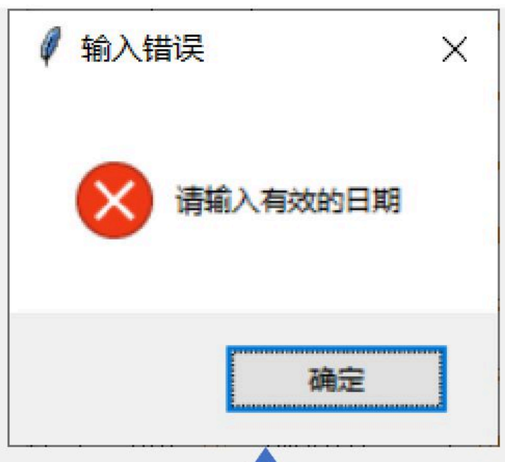
就不告诉你　　不是有效的数字

【digit】：数字

isdigit() 是字符串方法，用于检查字符串是否仅包含数字字符（即 0-9）。如果字符串中的所有字符都是数字字符，则返回 True，否则返回 False。

1.3 点击按钮解锁星座

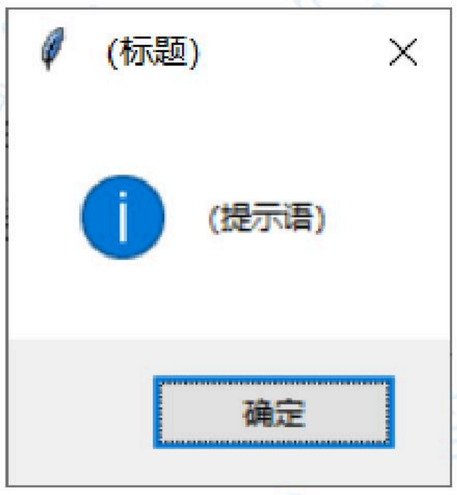
```
def click():
    year = entry_year.get()
    month = entry_month.get()
    day = entry_day.get()
    # 检查用户输入的年、月、日是否为有效的数字
    if not year.isdigit() or not month.isdigit() or not day.isdigit():
        # <弹出对话框，提示输入错误，提示输入有效日期>
```



messagebox 是 Tkinter 库中的一个模块，用于在应用程序中显示各种类型的消息对话框，例如信息对话框、警告对话框、错误对话框等。

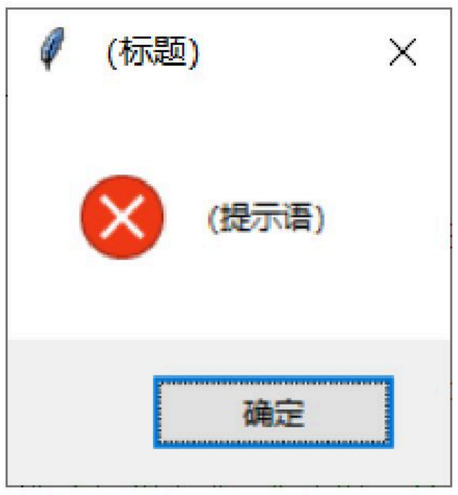
```
from tkinter import messagebox
```

给大家介绍几个messagebox 模块常用的函数，对话框如图所示



信息对话框

警告对话框



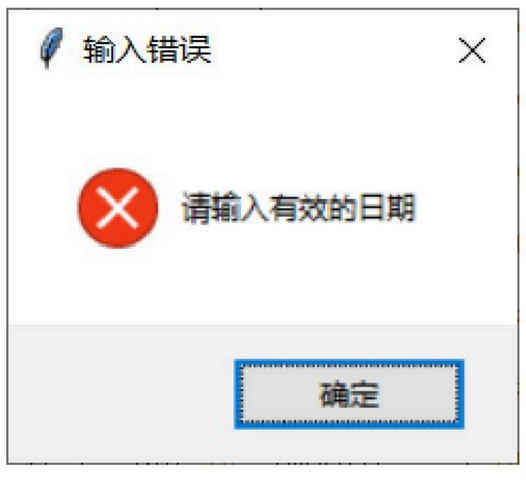
错误对话框

messagebox.showinfo(title, message) ：显示信息对话框

messagebox.showwarning(title, message) ：显示警告对话框

messagebox.showerror(title, message) ：显示错误对话框

```
# <弹出对话框，提示输入错误，提示输入有效日期>
messagebox.showerror("输入错误", "请输入有效的日期")
```



1.3 点击按钮解锁星座

确认用户输入的是有效数字后，接下来要确认月、日是否在有效范围内。

```
    if not year.isdigit() or not month.isdigit() or not day.isdigit():
        messagebox.showerror("输入错误", "请输入有效的日期")
        return
    month = int(month) # 将月份转换为整数
    day = int(day)  # 将日期转换为整数
    # 检查日期是否在有效范围内
    if month < 1 or month > 12 or day < 1 or day > 31:
        messagebox.showerror("输入错误", "请输入有效的日期")
```

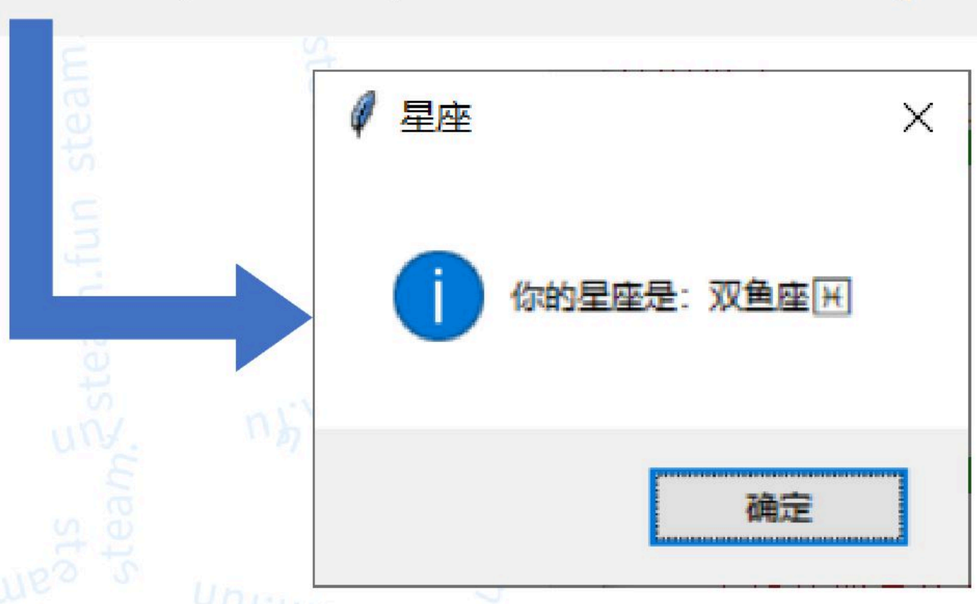
将month、day转换为数字类型

有效范围：
1 <= month <=12
1 <= day <= 31

排除无效输入后，接下来需要由已知的month、day返回对应的星座。

输入month、day参数，返回对应星座的check函数已经准备好啦！直接调用即可

```
    # 检查日期是否在有效范围内
    if month < 1 or month > 12 or day < 1 or day > 31:
        messagebox.showerror("输入错误", "请输入有效的日期")
    # 调用check函数获取星座信息
    result = check(month, day)
    # 显示用户星座信息
    messagebox.showinfo("星座", f"你的星座是：{result}")
```



1.3 点击按钮解锁星座

任务三新增代码：

```
from tkinter import messagebox
...
def click():
    year = entry_year.get() # 获取用户输入的出生年份
    month = entry_month.get()  # 获取用户输入的出生月份
    day = entry_day.get()  # 获取用户输入的出生日

    # 检查用户输入的年、月、日是否为有效的数字
    if not year.isdigit() or not month.isdigit() or not day.isdigit():
        messagebox.showerror("输入错误", "请输入有效的日期")
    month = int(month) # 将月份转换为整数
    day = int(day)  # 将日期转换为整数
    # 检查日期是否在有效范围内
    if month < 1 or month > 12 or day < 1 or day > 31:
        messagebox.showerror("输入错误", "请输入有效的日期")
    # 调用check函数获取星座信息
    result = check(month, day)
    # 显示用户星座信息
    messagebox.showinfo("星座", f"你的星座是: {result}")
```

完整代码

```
import tkinter as tk
from PIL import Image, ImageTk
from tkinter import messagebox

# 判断星座的函数(预置函数)
def check(month, day):
    ...

# 创建Tkinter窗口
root = tk.Tk()
root.geometry('320x360')
root.title("测测你的星座吧")

# 创建一个标签用于显示gif动画
label_bg = tk.Label(root)
label_bg.place(x=0, y=0, relwidth=1, relheight=1) # 标签充斥整个窗口
# 加载gif图片并将每一帧存储在frames列表中
image = Image.open('星图.gif')  # 打开gif图片文件
frames = []  # 用于存储gif的所有帧
try:
    while True:  # 将每一帧添加到frames列表中
        frames.append(ImageTk.PhotoImage(image))
        image.seek(len(frames))  # 跳转到gif的下一帧
except EOFError:
    pass  # 当到达gif结束时，跳出循环
# 用于更新gif动画帧的函数
def upd(idx):
    frame = frames[idx] # 获取当前帧
    label_bg.configure(image=frame) # 更新标签中显示的帧
    idx = (idx + 1) % len(frames)  # 计算下一帧的索引
    root.after(100, upd, idx)  # 100毫秒后调用自己继续更新帧
upd(0) # 开始播放gif动画
```

完整代码

```
# 创建标签和文本框
label_year = tk.Label(root, text="出生年: ")
label_year.grid(row=0, column=0, padx=(60,10), pady=(130,10))
entry_year = tk.Entry(root)
entry_year.grid(row=0, column=1, pady=(130,10))

label_month = tk.Label(root, text="出生月: ")
label_month.grid(row=1, column=0, padx=(60,10), pady=10)
entry_month = tk.Entry(root)
entry_month.grid(row=1, column=1, pady=10)

label_day = tk.Label(root, text="出生日: ")
label_day.grid(row=2, column=0, padx=(60,10), pady=10)
entry_day = tk.Entry(root)
entry_day.grid(row=2, column=1, pady=10)
```

2.强化练习

1. 以下哪种控件用于在 Tkinter 中创建一个单行文本输入框？（　　）

A. tk.Label

B. tk.Button

C. tk.Entry

D. tk.Text

2. 哪个模块用于在 Tkinter 程序中显示对话框？（　　）

A. tkmessage

B. msgbox

C. messagebox

D. tkdialog

3. 下面哪一行代码将标签控件放置在窗口中，并充满整个窗口？（　　）

A. entry_day.grid(row=2, column=1, pady=10)

B. label_bg.place(x=0, y=0, relwidth=1, relheight=1)

C. btn.grid(row=3, columnspan=2, padx=(80,10), pady=10)

D. label_day.grid(row=2, column=0, padx=(60,10), pady=10)

2.强化练习

4. 哪个库用于处理和展示 GIF 图像？（　　）

A. Tkinter

B. Pillow

C. Numpy

D. Matplotlib

5. 为什么在 GIF 动画帧更新函数 upd 中使用 idx = (idx + 1) % len(frames)？（　　）

A. 确保不断增大的索引

B. 使用 idx += 1 也可以

C. 让动画循环播放

D. 动态计算帧的总数

3.术语箱

seek	寻找，搜索	entry	登录，进入
grid	网格	messagebox	对话框
place	放置，位置	digit	数字
column	列		

4.课后挑战

生肖计算助手

要求：

1．窗口内有标签、单行文本输入框、提交按钮
2．输入正确的年份（1900-2100）后会弹出对话框显示生肖
3．输入的年份不在范围内或者不是数字格式，会弹出错误对话框
注意：判断生肖的函数check(year)已经预置好，可以直接调用；组件的位置可以自行调整，合理美观即可。