

**东莞城市学院**

实验报告

课程名称： 软件测试

专业班级：

学生学号：

学生姓名：

上课老师：

上课学期：

成 绩：

**实验评语：**

该生在本次实验中表现良好。成功完成了“第二日”问题的编程任务。在等价类划分和测试用例设计环节，展现了对等价类方法的初步掌握，能够合理地对输入情况进行分类，并设计出具有一定代表性的测试用例。整体实验过程认真投入，态度积极，值得肯定。希望继续保持这种良好的学习态度，进一步提升软件测试能力。

2025年5月8日

实验一 黑盒测试（1）

**一、实验目的：**

1.编程实现“第二日”问题，为后续的测试做准备。

2.通过“第二日”问题的等价类划分，掌握等价类方法及测试用例的设计。

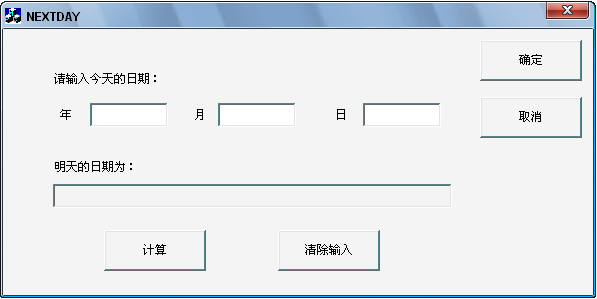
**二、实验学时**

2学时。

**三、实验设备**

PC电脑，安装有能够编译运行常用程序语言（C、C++、C#、JAVA等），安装有文档编辑软件（Office、WPS等）。

**四、实验要求**

1.实验结束，需要提交实验报告；

2.实验报告命名为“学号后2位-姓名-实验1”，例如：25-张三-实验1.docx;

**五、实验内容**

任务1：编程实现“第二日”问题（编程语言不限）。

根据用户输入的有效日期(格式为年-月-日)，自动计算下一天的日期。有效日期1800年1月1日到2050年12月31日。

上机编程并调试完成，填写表格。

|  |  |
| --- | --- |
| **“第二日”问题编程** | |
| **编程所用语言** | Python |
| **源代码** | import tkinter as tk from tkinter import messagebox from datetime import datetime  class DateCalculator:  def \_\_init\_\_(self, master):  self.master = master  master.title("第二日日期计算器")  master.geometry("380x250")   # 输入框架  input\_frame = tk.Frame(master)  input\_frame.pack(pady=10)   # 年输入  self.year\_label = tk.Label(input\_frame, text="年：")  self.year\_entry = tk.Entry(input\_frame, width=10)  self.year\_label.grid(row=0, column=0, padx=5)  self.year\_entry.grid(row=0, column=1, padx=5)   # 月输入  self.month\_label = tk.Label(input\_frame, text="月：")  self.month\_entry = tk.Entry(input\_frame, width=10)  self.month\_label.grid(row=1, column=0, padx=5)  self.month\_entry.grid(row=1, column=1, padx=5)   # 日输入  self.day\_label = tk.Label(input\_frame, text="日：")  self.day\_entry = tk.Entry(input\_frame, width=10)  self.day\_label.grid(row=2, column=0, padx=5)  self.day\_entry.grid(row=2, column=1, padx=5)   # 按钮框架  button\_frame = tk.Frame(master)  button\_frame.pack(pady=15)   self.confirm\_btn = tk.Button(button\_frame, text="确定", command=self.validate\_date, width=8)  self.calc\_btn = tk.Button(button\_frame, text="计算", command=self.calculate\_next\_day, width=8)  self.clear\_btn = tk.Button(button\_frame, text="清除", command=self.clear\_input, width=8)  self.cancel\_btn = tk.Button(button\_frame, text="取消", command=master.quit, width=8)   self.confirm\_btn.grid(row=0, column=0, padx=5)  self.calc\_btn.grid(row=0, column=1, padx=5)  self.clear\_btn.grid(row=0, column=2, padx=5)  self.cancel\_btn.grid(row=0, column=3, padx=5)   # 结果标签  self.result\_label = tk.Label(master, text="", fg="blue")  self.result\_label.pack(pady=10)   # 初始化有效日期  self.valid\_date = None   def validate\_date(self):  *"""验证输入的日期有效性"""* try:  year = int(self.year\_entry.get())  month = int(self.month\_entry.get())  day = int(self.day\_entry.get())  except ValueError:  messagebox.showerror("错误", "请输入有效的数字")  return   if not (1800 <= year <= 2050):  messagebox.showerror("错误", "年份必须在1800-2050之间")  return   try:  datetime(year, month, day)  self.valid\_date = (year, month, day)  messagebox.showinfo("成功", "日期验证通过！")  except ValueError:  messagebox.showerror("错误", "无效的日期")  self.valid\_date = None   def calculate\_next\_day(self):  *"""计算下一天日期"""* if not self.valid\_date:  messagebox.showerror("错误", "请先验证日期有效性")  return   year, month, day = self.valid\_date  try:  current\_date = datetime(year, month, day)  next\_date = current\_date.replace(day=current\_date.day + 1)  self.result\_label.config(text=f"第二日日期：{next\_date.year}-{next\_date.month}-{next\_date.day}")  except ValueError:  # 处理月末特殊情况  try:  next\_date = datetime(year, month + 1, 1)  except ValueError:  next\_date = datetime(year + 1, 1, 1)  self.result\_label.config(text=f"第二日日期：{next\_date.year}-{next\_date.month}-{next\_date.day}")   def clear\_input(self):  *"""清除所有输入"""* self.year\_entry.delete(0, tk.END)  self.month\_entry.delete(0, tk.END)  self.day\_entry.delete(0, tk.END)  self.valid\_date = None  self.result\_label.config(text="")   if \_\_name\_\_ == "\_\_main\_\_":  root = tk.Tk()  app = DateCalculator(root)  root.mainloop() |
| **源代码截图** |  |

任务2：“第二日”等价类划分。

基于“第二日”问题，运用等价划分的方法进行测试用例设计。

1.针对以下等价类划分，给出其对应典型值并写出测试用例集合。

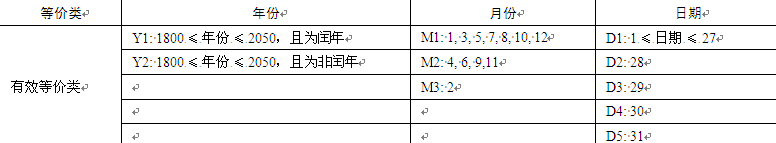
|  |  |  |  |
| --- | --- | --- | --- |
| 等价类 | 年 份 | 月 份 | 日 期 |
| 有效等价类 | Y1:1800≤年份≤2050 | M1:1≤月份≤12 | D1:1≤日期≤31 |
| 无效等价类 | Y2:年份<1800 | M2:月份<1 | D2:日期<1 |
| Y3:年份>2050 | M3:月份>12 | D3:日期>31 |

（1）典型值

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 典型值 | 编号 | 典型值 | 编号 | 典型值 |
| Y1 | 2000 | M1 | 6 | D1 | 15 |
| Y2 | 1799 | M2 | 0 | D2 | 0 |
| Y3 | 2051 | M3 | 13 | D3 | 32 |

（2）测试用例集合

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **输入数据** | **预期输出** | **等价组合** |
| 001 | 2000-6-15 | 2000-6-15 | Y1+M1+D1 |
| 002 | 1799-6-15 | 错误:年份必须>=1800 | Y2+M1+D1 |
| 003 | 2051-6-15 | 错误:年份必须<=2500 | Y3+M1+D1 |
| 004 | 2000-0-15 | 错误:月份必须在1~12之间 | Y1+M2+D1 |
| 005 | 2000-13-15 | 错误:月份必须在1~12之间 | Y1+M3+D1 |
| 006 | 2000-6-0 | 错误:日期必须>=1 | Y1+M1+D2 |
| 007 | 2000-6-32 | 错误:日期必须<=31 | Y1+M1+D3 |

2、针对以下等价类划分，给出其对应典型值并写出测试用例集合。

（1）典型值

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编号 | 典型值 | 编号 | 典型值 | 编号 | 典型值 |
| Y1 | 2020 | M1 | 3(31天月份) | D1 | 15 |
| Y2 | 2021 | M2 | 6(30天月份) | D2 | 28 |
|  | | M3 | 2(2月) | D3 | 29 |
|  | | | | D4 | 30 |
|  | | | | D5 | 31 |

(2)测试用例集合（表格不够请自行添加）

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **输入数据** | **预期输出** | **等价组合** |
| 001 | 2020-3-15 | 2020-3-16 | Y1+M1+D1 |
| 002 | 2020-2-28 | 2020-2-29 | Y1+M3+D2 |
| 003 | 2021-2-28 | 2021-3-1 | Y2+M3+D2 |
| 004 | 2020-2-29 | 2020-3-1 | Y1+M3+D3 |
| 005 | 2021-4-30 | 2021-5-1 | Y2+M2+D4 |
| 006 | 2020-12-31 | 2021-1-1 | Y1+M1+D5 |
| 007 | 2021-6-30 | 2021-7-1 | Y2+M2+D5 |
| 008 | 2020-7-31 | 2020-8-1 | Y1+M1+D5 |
| 009 | 2021-11-30 | 2021-12-1 | Y2+M2+D4 |
| 010 | 2021-10-31 | 2020-11-1 | Y1+M1+D5 |
| 011 | 1800-1-1 | 1800-1-2 | Y1+M1+D1 |
| 012 | 2050-12-31 | 2051-1-1 | Y1+M1+D5 |