**实验报告**

|  |  |  |
| --- | --- | --- |
| 课程名称：python程序设计实验课 | 班级：软件1912 | 实验日期：2021-5-27 |
| 姓名：邹一泓 | 学号：20192076 | 指导老师：杨帆 |
| 实验名称：循环语句 | 实验序号：09 | 实验成绩： |
| 1. 实验目的： 2. 动态显示数字时钟 3. 键盘事件响应函数（贴瓷砖 ） 4. 编写一个程序，提示用户输入以恶搞整数，然后反向显示一个整数 5. 猴子吃桃问题：猴子第一天摘下若干个桃子，当即吃了一半，还不瘾，又多吃了一个第二天早上又将剩下的桃子吃掉一半，又多吃了一个。以后每天早上都吃了前一天剩下的一半零一个。到第10天早上想再吃时，见只剩下一个桃子了。求第一天共摘了多少。   二、实验环境：  Python3.9.2 win10  三、实验步骤：  1.  from turtle import \* from datetime import \*  # 移动到指定位置 def skip(step):  penup()  forward(step)  pendown()  # 画指针 def drawpointer(name, length):  reset()  skip(-length \* 0.1)  begin\_poly()  forward(length \* 1.1)  end\_poly()  handForm = get\_poly()  register\_shape(name, handForm)  # 初始化 def init():  global hrpointer, minpointer, secpointer, weektext  # 设置turtle Logo模式，朝北  mode("logo")  drawpointer("hrpointer", 90)  drawpointer("minpointer", 130)  drawpointer("secpointer", 140)  hrpointer = Turtle()  hrpointer.shape("hrpointer")  minpointer = Turtle()  minpointer.shape("minpointer")  secpointer = Turtle()  secpointer.shape("secpointer")  secpointer.pencolor("red")  for pointer in hrpointer, minpointer, secpointer:  pointer.shapesize(3, 1, 1)  pointer.speed(0)  # 文字输出  weektext = Turtle()  weektext.hideturtle()  weektext.penup()  # 设置表盘形状 def setupClock(radius):  reset()  # clockPanel = Turtle()  pensize(7)  pencolor("blue")  for i in range(60):  skip(radius)  if i % 5 == 0:  forward(20)  skip(-radius - 20)  else:  dot(5)  skip(-radius)  right(6)   # 星期文本  def Week(t):  week = ["星期一", "星期二", "星期三", "星期四", "星期五", "星期六", "星期日"]  return week[t.weekday()]  # 日期文本 def Date(t):  y = t.year  m = t.month  d = t.day  return "%s %d %d" % (y, m, d)  # 绘制表针动态显示 def tick():  t = datetime.today()  second = t.second + t.microsecond \* 0.0000001  minute = t.minute + second / 60.0  hour = t.hour + minute / 60.0  secpointer.setheading(6 \* second)  minpointer.setheading(6 \* minute)  hrpointer.setheading(30 \* hour)  tracer(False)  weektext.forward(65)  weektext.write(Week(t), align="center", font=("Courier", 14, "bold"))  weektext.back(130)  weektext.write(Date(t), align="center", font=("Courier", 14, "bold"))  weektext.home()  tracer(True)  # 间隔100ms调用一次  ontimer(tick, 100)  def main():  tracer(False)  init()  setupClock(160)  tracer(True)  tick()  mainloop()  if \_\_name\_\_ == '\_\_main\_\_':  main()  secpointer.shape("secpointer")  secpointer.pencolor("red")  for pointer in hrpointer, minpointer, secpointer:  pointer.shapesize(3, 1, 1)  pointer.speed(0)  # 文字输出  weektext = Turtle()  weektext.hideturtle()  weektext.penup()  # 设置表盘形状 def setupClock(radius):  reset()  # clockPanel = Turtle()  pensize(7)  pencolor("blue")  for i in range(60):  skip(radius)  if i % 5 == 0:  forward(20)  skip(-radius - 20)  else:  dot(5)  skip(-radius)  right(6)   # 星期文本  def Week(t):  week = ["星期一", "星期二", "星期三", "星期四", "星期五", "星期六", "星期日"]  return week[t.weekday()]  # 日期文本 def Date(t):  y = t.year  m = t.month  d = t.day  return "%s %d %d" % (y, m, d)  # 绘制表针动态显示 def tick():  t = datetime.today()  second = t.second + t.microsecond \* 0.0000001  minute = t.minute + second / 60.0  hour = t.hour + minute / 60.0  secpointer.setheading(6 \* second)  minpointer.setheading(6 \* minute)  hrpointer.setheading(30 \* hour)  tracer(False)  weektext.forward(65)  weektext.write(Week(t), align="center", font=("Courier", 14, "bold"))  weektext.back(130)  weektext.write(Date(t), align="center", font=("Courier", 14, "bold"))  weektext.home()  tracer(True)  # 间隔100ms调用一次  ontimer(tick, 100)  def main():  tracer(False)  init()  setupClock(160)  tracer(True)  tick()  mainloop()  if \_\_name\_\_ == '\_\_main\_\_':  main()  2.  import turtle  unit\_length = 100 width = 4 height = 4 origin\_x = 0 # 由于移动瓷砖需要修改原点坐标，因此使用变量，而不使用元组 origin\_y = 0 rotate = 0 # 旋转次数 edge\_point\_list = [] center\_point\_list = []  def draw\_L():  edge\_point\_list.clear()  turtle.reset() # turtle 清除窗口  turtle.penup()  turtle.goto(origin\_x, origin\_y) # 光标移动到原点坐标  turtle.right(90 \* (rotate % 4)) # 根据旋转次数计算旋转角度  turtle.pendown()  turtle.fillcolor("blue")  turtle.begin\_fill()  turtle.forward(100)  turtle.right(90)  turtle.forward(100)  edge\_point\_list.append(turtle.position())  turtle.right(90)  turtle.forward(100 \* 2)  edge\_point\_list.append(turtle.position())  turtle.right(90)  turtle.forward(100 \* 2)  edge\_point\_list.append(turtle.position())  turtle.right(90)  turtle.forward(100)  turtle.right(90)  turtle.forward(100)  turtle.end\_fill()  center\_point\_list.clear()  center\_point\_list.append(((origin\_x + edge\_point\_list[0][0]) \* 0.5, (origin\_y + edge\_point\_list[0][1]) \* 0.5))  center\_point\_list.append(((origin\_x + edge\_point\_list[1][0]) \* 0.5, (origin\_y + edge\_point\_list[1][1]) \* 0.5))  center\_point\_list.append(((origin\_x + edge\_point\_list[2][0]) \* 0.5, (origin\_y + edge\_point\_list[2][1]) \* 0.5))  print(edge\_point\_list)  print(center\_point\_list)   def move\_up\_tiling():  global origin\_x  global origin\_y  origin\_x = origin\_x  origin\_y = origin\_y + 1 \* unit\_length  draw\_L()  def move\_down\_tiling():  global origin\_x  global origin\_y  origin\_x = origin\_x  origin\_y = origin\_y - 1 \* unit\_length  draw\_L()  def move\_left\_tiling():  global origin\_x  global origin\_y  origin\_x = origin\_x - 1 \* unit\_length  origin\_y = origin\_y  draw\_L()  def move\_right\_tiling():  global origin\_x  global origin\_y  origin\_x = origin\_x + 1 \* unit\_length  origin\_y = origin\_y  draw\_L()  def rotate\_tiling():  global rotate  rotate = rotate + 1  draw\_L()  turtle.setup((width + 5) \* unit\_length, (height + 2) \* unit\_length)  def main():  win = turtle.Screen()  win.tracer(0) # 不显示绘制轨迹  win.onkey(draw\_L, 't') # 按T键绘制瓷砖  win.onkey(move\_up\_tiling, 'Up') # 按向上键，向上移动  win.onkey(move\_down\_tiling, 'Down') # 按向下键，向下移动  win.onkey(move\_left\_tiling, 'Left') # 按向左键，向左移动  win.onkey(move\_right\_tiling, 'Right') # 按向右键，向右移动  win.onkey(rotate\_tiling, 'r') # 按 R 键，顺时针旋转 90 度  win.listen() # 窗口监听  win.mainloop()  if \_\_name\_\_ == '\_\_main\_\_':  main()  secpointer.shape("secpointer")  secpointer.pencolor("red")  for pointer in hrpointer, minpointer, secpointer:  pointer.shapesize(3, 1, 1)  pointer.speed(0)  # 文字输出  weektext = Turtle()  weektext.hideturtle()  weektext.penup()  # 设置表盘形状 def setupClock(radius):  reset()  # clockPanel = Turtle()  pensize(7)  pencolor("blue")  for i in range(60):  skip(radius)  if i % 5 == 0:  forward(20)  skip(-radius - 20)  else:  dot(5)  skip(-radius)  right(6)  # 星期文本  def Week(t):  week = ["星期一", "星期二", "星期三", "星期四", "星期五", "星期六", "星期日"]  return week[t.weekday()]  # 日期文本 def Date(t):  y = t.year  m = t.month  d = t.day  return "%s %d %d" % (y, m, d)  # 绘制表针动态显示 def tick():  t = datetime.today()  second = t.second + t.microsecond \* 0.0000001  minute = t.minute + second / 60.0  hour = t.hour + minute / 60.0  secpointer.setheading(6 \* second)  minpointer.setheading(6 \* minute)  hrpointer.setheading(30 \* hour)  tracer(False)  weektext.forward(65)  weektext.write(Week(t), align="center", font=("Courier", 14, "bold"))  weektext.back(130)  weektext.write(Date(t), align="center", font=("Courier", 14, "bold"))  weektext.home()  tracer(True)  # 间隔100ms调用一次  ontimer(tick, 100)  def main():  tracer(False)  init()  setupClock(160)  tracer(True)  tick()  mainloop()  if \_\_name\_\_ == '\_\_main\_\_':  main()  3.  def reserve(num):  i = 0  num1 = num  while True:  if num1 // 10 == 0:  break  i += 1  num1 = num1 // 10  sum = 0  while i >= 0:  sum = sum + (num % 10) \* (10 \*\* i)  num = num // 10  i = i - 1  return sum  def main():  num = int(input("请输入一个数:"))  print(reserve(num))  if \_\_name\_\_ == '\_\_main\_\_':  main()  4.  def monkey\_tao(n):  if n > 10 or n < 1:  return  elif n == 10:  return 1  else:  return (monkey\_tao(n + 1) + 1) \* 2  def main():  print(monkey\_tao(1))  if \_\_name\_\_ == '\_\_main\_\_':  main()  四、实验结果：  1.    2.    3.    4.    五、实验心得：  人生苦短我用python. | | |