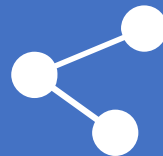


分布式仿真平台使用流程

清华大学自动化系 Tsinghua University

严 虎 Mail: yanh20@mails.tsinghua.edu.cn Wechat: w2624507753
王奕凡 Mail: wangyifa17@mails.tsinghua.edu.cn Wechat: Wyf_1995_



硬件环境

Windows/ Linux/ Mac OS X操作系统计算机一台(以Windows为例)











软件环境

Windows操作系统下运行DSP.exe必备的软件环境:

- Python3
- MySql
- pysnooperDB

步骤1：解压DSP2.0.zip

- 拷贝或下载分布式算法仿真平台程序压缩包DSP2.0.zip。
- 将压缩包置于英文目录下进行解压，包含文件应如下图：

 IoT	2020/1/31 20:41	文件夹	
 othercode	2020/11/22 22:46	文件夹	
 sample_code	2020/11/22 23:35	文件夹	
 DSP.exe	2020/1/31 19:37	应用程序	52,912 KB
 DSP.py	2020/1/31 20:25	PY 文件	1 KB
 DSP.spec	2020/1/31 19:36	SPEC 文件	1 KB
 IoTGUI.py	2020/1/30 21:12	PY 文件	53 KB
 myThread.py	2020/1/21 23:57	PY 文件	1 KB
 myThread2.py	2020/1/22 19:03	PY 文件	2 KB
 分布式仿真平台2.0说明文档.pdf	2020/11/22 23:01	Adobe Acrobat 文档	633 KB

步骤2：验证软件环境-Python

- 安装的python环境应为python3.x 32/64位，命令行输入python提示如下：

```
C:\Users\26245>python
Python 3.7.6 (tags/v3.7.6:43364a7ae0, Dec 18 2019, 23:46:00) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
```

- 若python环境不正确，到<https://www.python.org/downloads/>下载适宜电脑的版本，安装后配置环境变量路径。

步骤2：验证软件环境-MySQL

- MySQL的安装教程可参考<https://www.runoob.com/mysql/mysql-install.html>
windows 10操作系统的详细安装说明: <https://www.runoob.com/w3cnote/windows10-mysql-installer.html>
- 按说明安装后需建立调试程序所用的数据库，登录MySQL服务

```
C:\Users\26245>MySQL -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 443
Server version: 8.0.18 MySQL Community Server - GPL

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

步骤2：验证软件环境-MySQL

- 创建调试模式所用数据库TESTDB

```
mysql> CREATE DATABASE TESTDB;  
Query OK, 1 row affected (0.01 sec)
```

- 显示所有数据库

```
mysql> SHOW DATABASES;  
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sakila |  
| sys |  
| testdb |  
| world |  
+-----+
```

步骤2: 验证软件环境-PySnooperDB

- 使用pip命令安装PySnooperDB

```
C:\Users\Yvan>pip install PySnooperDB==1.0.1
Collecting PySnooperDB==1.0.1
  Downloading PySnooperDB-1.0.1.tar.gz (44 kB)
    |████████████████████████████████████████| 44 kB 656 kB/s
Requirement already satisfied: mysql-connector-python in d:\program\python38\lib\site-packages (from PySnooperDB==1.0.1) (8.0.21)
Requirement already satisfied: protobuf>=3.0.0 in d:\program\python38\lib\site-packages (from mysql-connector-python->PySnooperDB==1.0.1) (3.13.0)
Requirement already satisfied: setuptools in d:\program\python38\lib\site-packages (from protobuf>=3.0.0->mysql-connector-python->PySnooperDB==1.0.1) (49.2.1)
Requirement already satisfied: six>=1.9 in c:\users\yvan\appdata\roaming\python\python38\site-packages (from protobuf>=3.0.0->mysql-connector-python->PySnooperDB==1.0.1) (1.15.0)
Using legacy 'setup.py install' for PySnooperDB, since package 'wheel' is not installed.
Installing collected packages: PySnooperDB
  Attempting uninstall: PySnooperDB
    Found existing installation: PySnooperDB 2.0.0
    Uninstalling PySnooperDB-2.0.0:
      Successfully uninstalled PySnooperDB-2.0.0
  Running setup.py install for PySnooperDB ... done
Successfully installed PySnooperDB-1.0.1
```

步骤3：运行DSP.exe

- 运行界面如下图：

分布式仿真平台

控制按钮

节点个数: 主节点:

运行

拓扑信息:

算法程序:

退出

运行状态

运行结果

调试模式开关 ☐

调试模式信息输入

数据库用户名: 数据库密码: 数据库名: 数据表前缀名:

查看变量名:

调试模式显示控制

数据表名:

调试信息显示

步骤4：输入拓扑信息

- 输入节点个数以及拓扑结构对应的JSON数据（这里提供了一个例程的拓扑和算法在文件夹sample_code中，将拓扑模板的拓扑上传）



拓扑模板格式

```
[
  {
    "_comment": "_comment中的文字仅作为注释，实际导入中可删去。拓扑请使用json数据格式进行导入，具体形式为每个节点描述组成的数组。",
    "_comment2": "单个节点具体格式如下所示，ID为1~n的编号；IP为该节点IP；PORT为给该节点指定的6个通信服务器与1个任务服务器的端口数组；adjID为该节点邻接节点数组，要求对称性；adjDirection为邻居节点对应的端口编号，需在1~6中取值；datalist为该节点初始化时需传入的数据，格式自定义。接下来为实例，指定一个正方形中的连接关系"
  },
  {
    "ID": 1,
    "IP": "localhost",
    "PORT": [10000, 10001, 10002, 10003, 10004, 10005, 10006],
    "adjID": [2, 3],
    "adjDirection": [1, 2],
    "datalist": {}
  },
  {
    "ID": 2,
    "IP": "localhost",
    "PORT": [10007, 10008, 10009, 10010, 10011, 10012, 10013],
    "adjID": [1, 4],
    "adjDirection": [3, 2],
    "datalist": {}
  }
]
```



拓扑输入

```
[
  {
    "ID": 3,
    "IP": "localhost",
    "PORT": [10014, 10015, 10016, 10017, 10018, 10019, 10020],
    "adjID": [1, 4],
    "adjDirection": [4, 1],
    "datalist": {}
  },
  {
    "ID": 4,
    "IP": "localhost",
    "PORT": [10021, 10022, 10023, 10024, 10025, 10026, 10027],
    "adjID": [2, 3],
    "adjDirection": [4, 3],
    "datalist": {}
  }
]
```

确定 取消

步骤5：导入算法程序

- 根据模板格式编写python算法程序，并进行上传（上传FindMaxTemper.py文件）

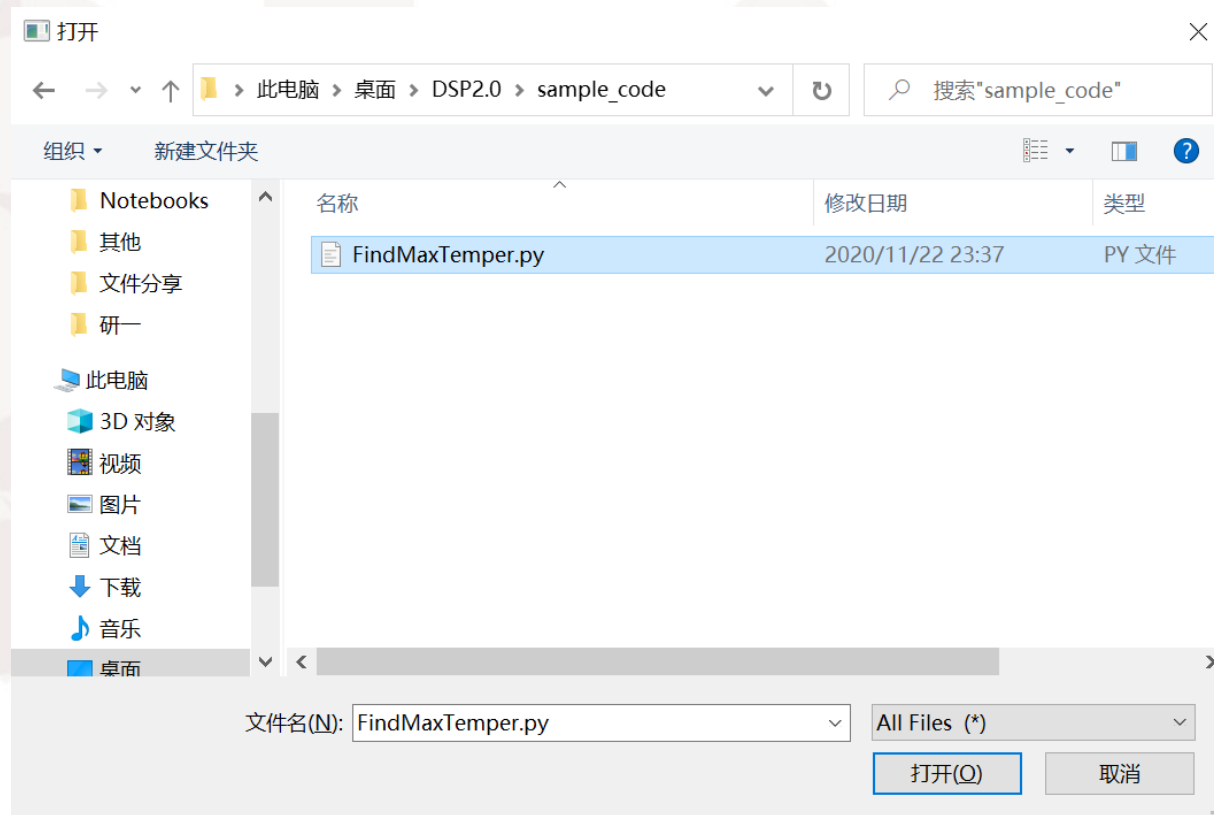


```
# import需求模块

# 用户自定义函数区
"""
用户编码区域
"""

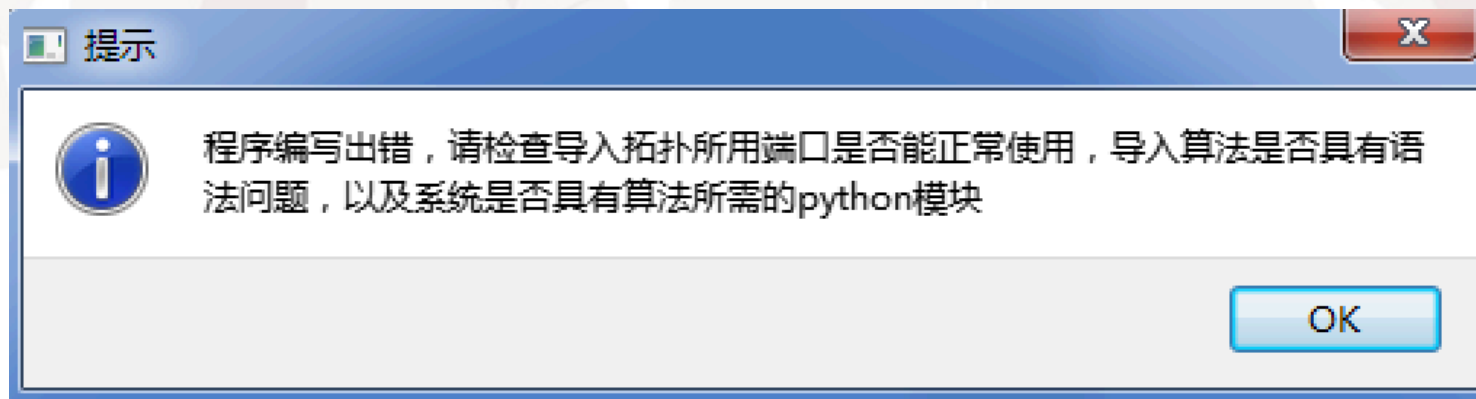
# 定义算法主函数taskFunction(self, id, adjDirection, datalist)，四个形参分别为
# 节点类，节点ID，节点邻居对应的方向以及初始化时键入的数据
def taskFunction(self, id, adjDirection, datalist):
    # 在算法设计中，可与邻居节点进行通信，函数为self.transmitData(data)，
    # data为需通信数据，可为字符串、对象、数组等JSON格式能解析的数据类型
    # 返回数据feedback = self.transmitData(data)为二元数组，第一分量表示
    # 返回邻居方向adjDirection，第二分量表示相应数据，与第一分量中方向一一对应
    # 通信的异步函数为self.sendDataToDirection(direction, data)，direction
    # 为需要传递数据的方向，data为需通信数据，可为字符串、对象、数组等JSON格式
    # 能解析的数据类型，调用该函数时可通过self.adjData获取邻居传递过来的数据信息，
    # 内容与方向adjDirection一一对应
    # 异步通信函数请勿与同步通信函数同时使用，使用异步通信函数时需要考虑
    # 到程序执行的异步性与延时，慎用！
    # 调用异步函数时可以使用self.syncNode()函数使得程序实现同步
    # 在算法设计中，可在运行状态区域打印出自己调试所需要的算法运行信息，
    # 函数为self.sendUDP(str)，str为字符串格式的数据

    value = []
    # value为返回值，可为任意格式
```



步骤6：运行节点，仿真算法

- 确保前几步完成后，点击运行按钮，等待算法结果
- 若运行状态没有节点信息输出，则是仿真平台节点没有成功启动；若出现下图提示，则节点启动失败。
- 运行结果中返回的各节点的value值若不为空，说明算法运行成功，反之说明算法运行失败，可于运行状态中查看程序错误信息，进行调试。



常见错误

1. 运行程序时出现“程序编写出错”的提示。

原因：该问题出现的原因主要有以下几点：系统python环境不正确，拓扑信息中的端口被其他程序占用导致无法使用，导入的算法本身具有语法问题，系统python不具备算法使用的部分模块。

2. 算法中调用self.sendUDP时运行状态中无法收到输出信息。

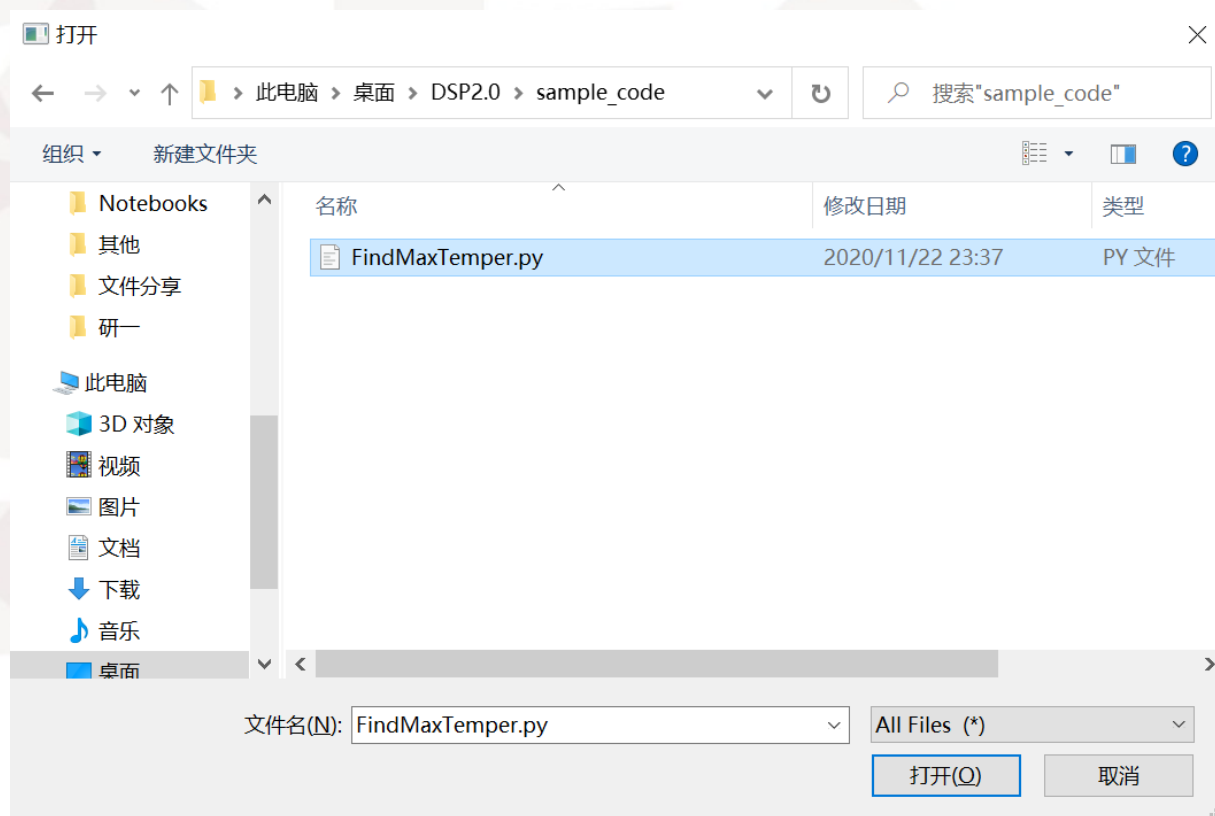
原因：请将整个DSP2.0文件夹置于英文目录下运行。

The background features a cluster of overlapping squares in various shades of light beige and cream, creating a textured, layered effect. The squares are of different sizes and are slightly rotated, giving a sense of depth and movement.

调试模式的使用流程

步骤1：输入节点数量，导入拓扑和算法程序

- 同正常运行的步骤1~步骤5，输入节点数量，导入拓扑和算法程序。



步骤2：输入数据库相关信息

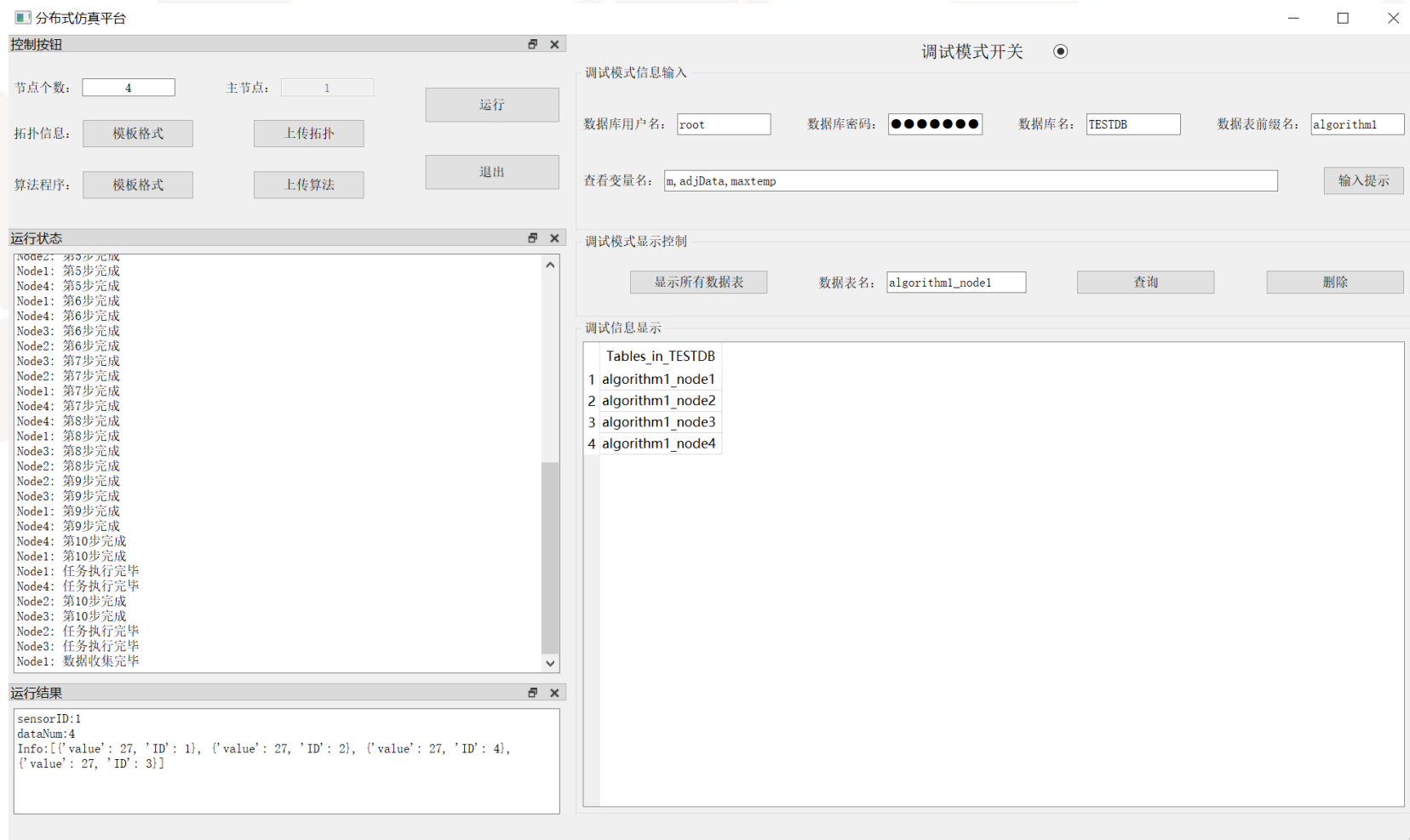
- 打开“调试模式开关”，输入数据库用户名、数据库密码、数据库名、数据表前缀名，点击“显示所有数据表”检查输入信息是否有误。
- 右图“调试信息显示”中正确显示数据库中的所有数据表名，所填信息无误。

The screenshot displays a 'Debug Mode' (调试模式) interface. At the top, there is a 'Debug Mode Switch' (调试模式开关) set to 'On' (开启). Below this, the 'Debug Mode Information Input' (调试模式信息输入) section contains four input fields: 'Database Username' (数据库用户名) with 'root', 'Database Password' (数据库密码) with masked characters, 'Database Name' (数据库名) with 'TESTDB', and 'Table Prefix' (数据表前缀名) with 'algorithm1'. A 'View Variable Name' (查看变量名) field contains 'm, adjData, maxtemp' with an 'Input Hint' (输入提示) button. The 'Debug Mode Display Control' (调试模式显示控制) section has a 'Show All Data Tables' (显示所有数据表) button, a 'Table Name' (数据表名) field with 'algorithm1_node1', and 'Query' (查询) and 'Delete' (删除) buttons. The 'Debug Information Display' (调试信息显示) section shows a table titled 'Tables in TESTDB' with four rows of table names.

Tables in TESTDB	
1	algorithm1_node1
2	algorithm1_node2
3	algorithm1_node3
4	algorithm1_node4

步骤3：输入查看变量，运行算法

- 输入查看变量名m,adjData,maxtemp，点击运行按钮，等待算法结果。



步骤4：查看变量变化情况，进行调试

- 程序运行结束后，点击“显示所有数据表”，将程序生成的数据表名输入“数据表名”中，点击查询获取数据表信息，点击删除删除该数据表。
- 可依据变量的变化情况进行相应的调试。

调试模式开关 ☒

调试模式信息输入

数据库用户名: 数据库密码: 数据库名: 数据表前缀名:

查看变量名:

调试模式显示控制

数据表名:

调试信息显示

	code	codeline	maxtemp	m	adjData
1	maxtemp = temp	12	25	None	None
2	for m in range(10):	13	25	0	None
3	adjData = data[1]	16	25	0	[18, 6]
4	for m in range(10):	13	25	1	[18, 6]
5	adjData = data[1]	16	25	1	[27, 27]
6	maxtemp = adjData[i]	19	27	1	[27, 27]
7	for m in range(10):	13	27	2	[27, 27]
8	for m in range(10):	13	27	3	[27, 27]
9	for m in range(10):	13	27	4	[27, 27]
10	for m in range(10):	13	27	5	[27, 27]
11	for m in range(10):	13	27	6	[27, 27]
12	for m in range(10):	13	27	7	[27, 27]
13	for m in range(10):	13	27	8	[27, 27]
14	for m in range(10):	13	27	9	[27, 27]

The background features a cluster of overlapping squares in various shades of light beige and cream, creating a textured, layered effect. The squares are of different sizes and are slightly rotated, giving a sense of depth and movement.

谢谢聆听！