# 多线程

- 每0.5秒输出1
- 每1.0秒输出2
- 每1.5秒输出3
- 每2.0秒输出4
- ....

# 多线程

- 常规设计

```
import time
while True:
        time.sleep(0.5)
        print('1',end='.')
        time.sleep(0.5)
        print('1',end='.')
        print('2',end='.')
        time.sleep(0.5)
        print('1',end='.')
        time.sleep(0.5)
        print('1',end='.')
        print('2',end='.')
```

再继续插入

# 多线程

- 改进

```
import time
tick = 0
while True:
    time.sleep(0.5)
    tick += 1
    if tick%1 == 0:
        print('1', end='.')
    if tick%2==0:
        print('2', end='.')
    if tick%3==0:
        print('3', end='.')
    if tick%4==0:
        print('4')
```

非倍数呢？

# 多线程

```
import time
import threading
def task1():
        while(True):
                time.sleep(0.5)
                print('1',end='.')
def task2():
        while(True):
                time.sleep(1)
                print('2',end='.')
def task3():
        while(True):
                time.sleep(1.5)
                print('3',end='.')
def task4():
        while(True):
                time.sleep(2.0)
                print('4')
```

```
th1 = threading.Thread(target = task1)
th1.start()
th2 = threading.Thread(target = task2)
th2.start()
th3 = threading.Thread(target = task3)
th3.start()
th4 = threading.Thread(target = task4)
th4.start()
```

# 多线程

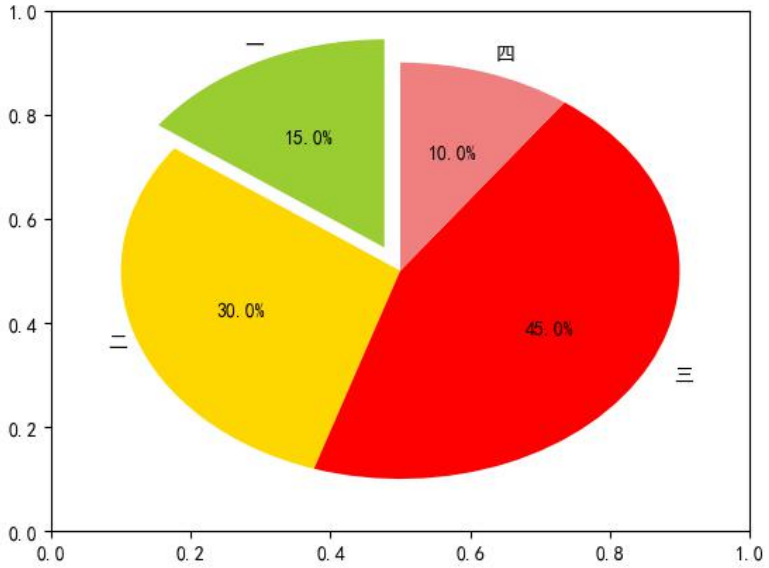
```
import time
import threading
def task(i):
        while(True):
                time.sleep(0.5*i)
                if i<4:
                        print(str(i),end='.')
                else:
                        print(str(i))
for i in range(1,5):
        threading.Thread(target = task,args=[i]).start()
```

# 简单爬虫

```
#win+R, cmd, pip install beautifulsoup4
from bs4 import BeautifulSoup
import requests
print('读取自动化学院新闻')
url = 'http://auto.hdu.edu.cn/3778/list.htm'
web_data = requests.get(url)
web_data.encoding = 'utf-8'
soup = BeautifulSoup(web_data.text, 'html.parser')
#print(soup)
for news in soup.select('.newstitle'):
        print(str(news.select('a'))[44:-5])
```

# 数据可视化

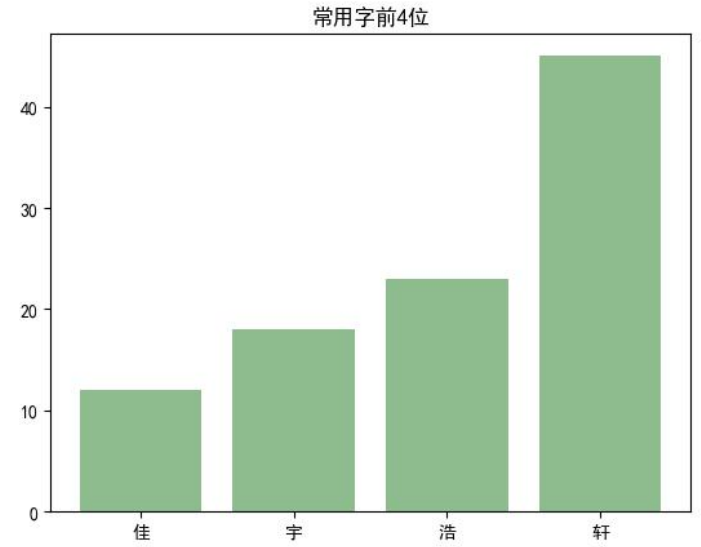
## 饼状图




```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pylab import *
#注意逆时针
labels = '一', '二', '三', '四'
sizes =  [15, 30, 45, 10]                    #占比
colors = ['yellowgreen', 'gold', '#FF0000', 'lightcoral']
explode = (0.05, 0, 0, 0)                    #分离
mpl.rcParams['font.sans-serif'] = ['SimHei']   #设置中文字体
fig = plt.figure()
ax = fig.gca()
ax.pie(sizes, explode=explode, labels=labels, colors=colors,
      autopct='%1.1f%%', shadow=False, startangle=90,
      radius=0.4, center=(0.5, 0.5), frame=True)
plt.show()
```

# 数据可视化

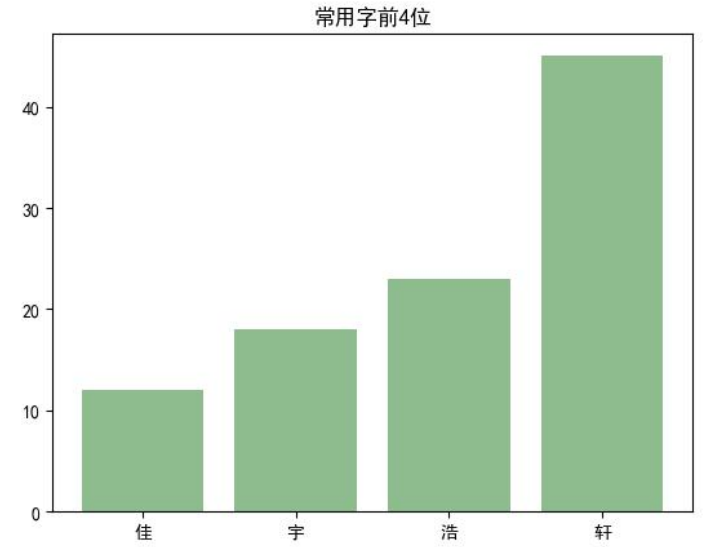
## ❖ 柱状图




```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pylab import *
labels = '佳','宇','浩','轩'
num_list = 12,18,23,45
plt.title("常用字前4位")
mpl.rcParams['font.sans-serif'] = ['SimHei']   #设置中文字体
plt.bar(labels,num_list)
plt.show()
```

# 数据可视化

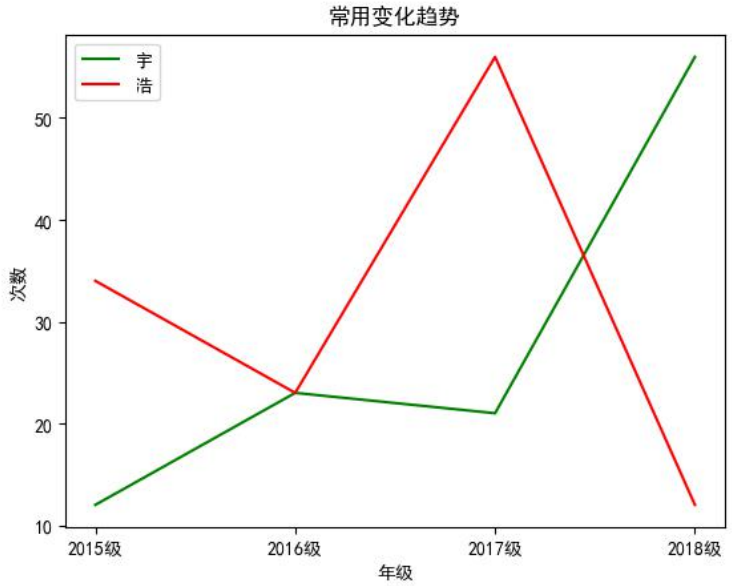
## ❖ 柱状图-不变顺序




```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pylab import *
#labels = ['佳','宇','浩','轩']
labels = [3,4,2,1]
num_list = [23,12,18,45]
mpl.rcParams['font.sans-serif'] = ['SimHei']   #设置中文字体
fig=plt.figure(figsize=(10,5))
axes=fig.add_subplot(1,1,1)
axes.bar(range(len(labels)),num_list)          #序号从小到大
axes.set_xticks(range(len(labels)))                 #设置柱个数
axes.set_xticklabels(labels)                        #添加柱标
plt.show()
```

# 数据可视化

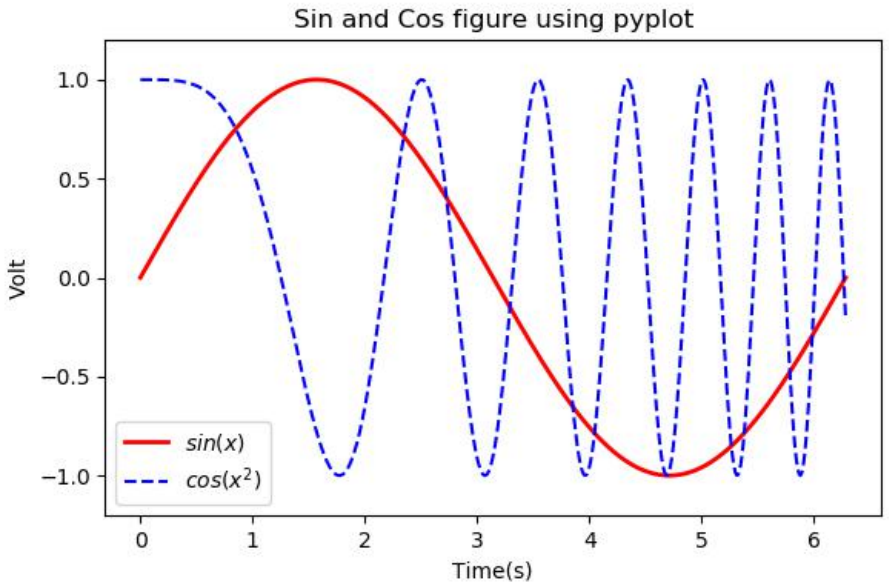
## ❖ 折线图




```
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
from pylab import *
x=['2015级','2016级','2017级','2018级']
y1=[12,23,21,56]
y2=[34,23,56,12]
num_list = 18,23,12,45
plt.title("常用字变化趋势")
mpl.rcParams['font.sans-serif'] = ['SimHei']   #设置中文字体
plt.plot(x,y1, color='green', label='宇')
plt.plot(x,y2, color='red',   label='浩')
plt.legend()                                   #设置图例（左上角）
plt.xlabel('年级')
plt.ylabel('次数')
plt.show()
```

# 数据可视化

- 正余弦曲线

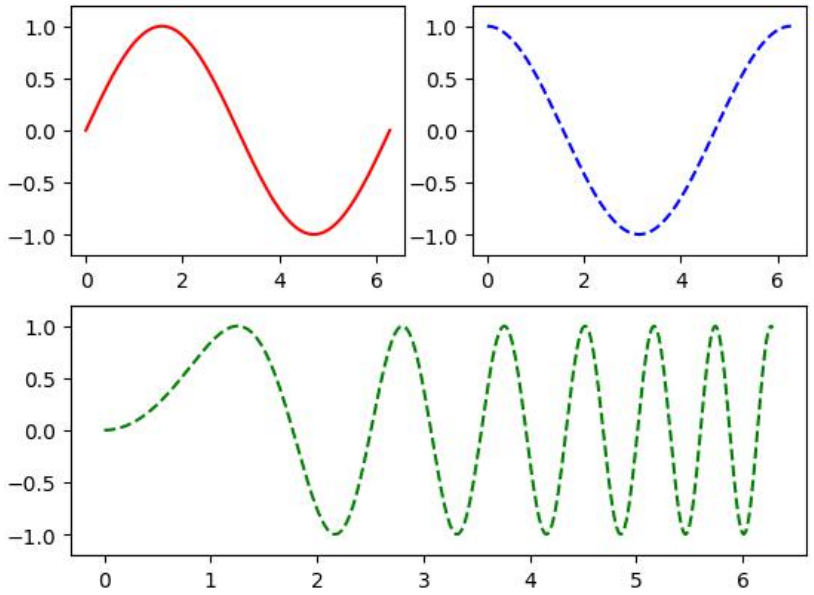



```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 2*np.pi, 500)
y = np.sin(x)
z = np.cos(x*x)
plt.figure(figsize=(8,4))
plt.plot(x,y, label='$sin(x)$',color='red',linewidth=2)
plt.plot(x,z, 'b--',label='$cos(x^2)$')
plt.xlabel('Time(s)')
plt.ylabel('Volt')
plt.title('Sin and Cos figure using pyplot')
plt.ylim(-1.2,1.2)
plt.legend()
plt.savefig('sin_cos.png',dpi=120)
plt.show()
```

# 数据可视化

- 多个正余弦曲线

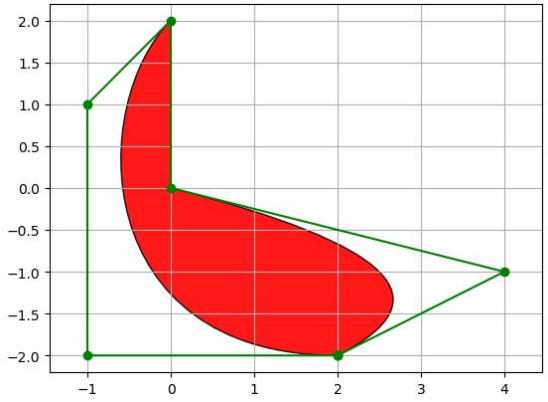
```
import numpy as np
import matplotlib.pyplot as plt
x = np.linspace(0, 2*np.pi, 500)
y1, y2, y3= np.sin(x), np.cos(x),
np.sin(x*x)
plt.figure(1)
ax1 = plt.subplot(2,2,1)
ax2 = plt.subplot(2,2,2)
ax3 = plt.subplot(2,1,2)
plt.sca(ax1)
plt.plot(x,y1,color='red')
```




```
plt.ylim(-1.2,1.2)
plt.sca(ax2)
plt.plot(x,y2,'b--')
plt.ylim(-1.2,1.2)
plt.sca(ax3)
plt.plot(x,y3,'g--')
plt.ylim(-1.2,1.2)
plt.legend()
plt.show()
```

# 数据可视化

## ❖ 贝塞尔曲线

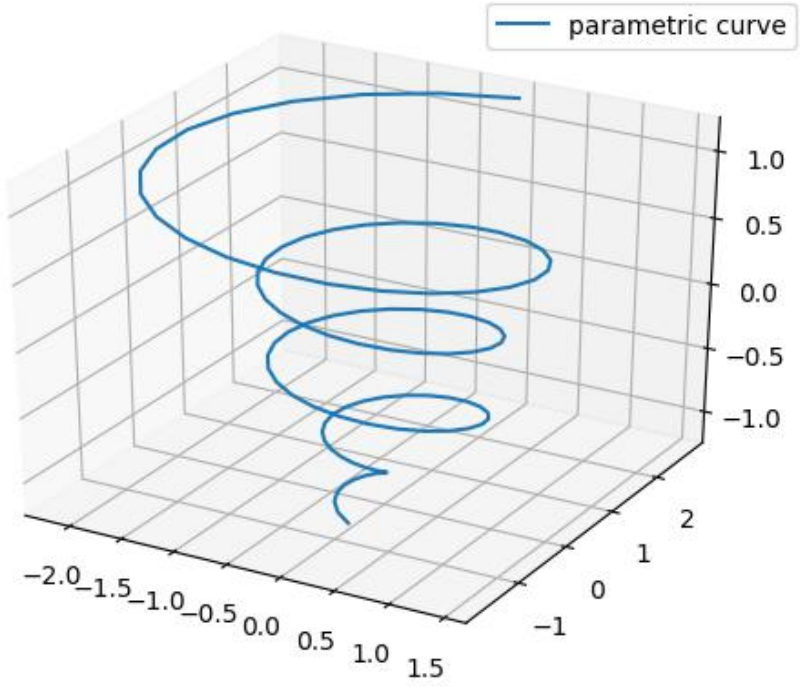



```
#定义绘图指令与控制点坐标
#其中MOVETO表示将绘制起点移动到指定坐标
#CURVE4表示使用4个控制点绘制3次贝塞尔曲线
#CURVE3表示使用3个控制点绘制2次贝塞尔曲线
#LINETO表示从当前位置绘制直线到指定位置
#CLOSEPOLY表示从当前位置绘制直线到指定位置并闭合多边形
from matplotlib.path import Path
from matplotlib.patches import PathPatch
import matplotlib.pyplot as plt
fig, ax = plt.subplots()
path_data = [
        (Path.MOVETO, (2, -2)),
        (Path.CURVE4, (-1, -2)),
        (Path.CURVE4, (-1, 1)),
        (Path.CURVE4, (0, 2)),
        (Path.LINETO, (0, 0)),
        (Path.CURVE3, (4, -1)),
        (Path.CURVE3, (2, -2)),
        (Path.CLOSEPOLY, (2, -2)),
      ]
codes, verts = zip(*path_data)
path = Path(verts, codes)
patch = PathPatch(path, facecolor='r', alpha=0.9)
ax.add_patch(patch)
x, y = zip(*path.vertices)
line, = ax.plot(x, y, 'go-')
ax.grid()
ax.axis('equal')
plt.show()
```

# 数据可视化

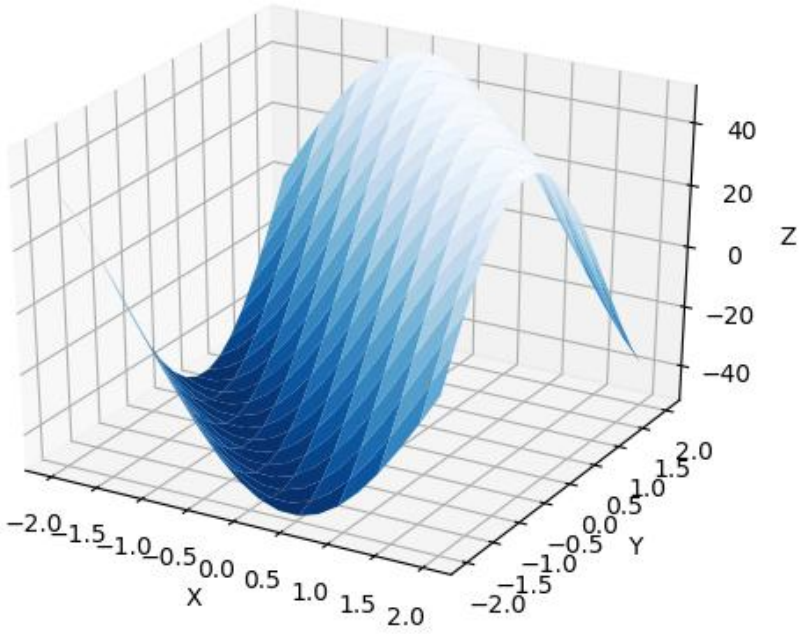
## ❖ 3D曲线

```
import matplotlib as mpl
from mpl_toolkits.mplot3d import Axes3D
import numpy as np
import matplotlib.pyplot as plt
mpl.rcParams['legend.fontsize'] = 10
fig = plt.figure()
ax = fig.gca(projection='3d')
theta = np.linspace(-4 * np.pi, 4 * np.pi, 100)
z = np.linspace(-4, 4, 100)*0.3
r = z**3 + 1
x = r * np.sin(theta)
y = r * np.cos(theta)
ax.plot(x, y, z, label='parametric curve')
ax.legend()
plt.show()
```



# 数据可视化

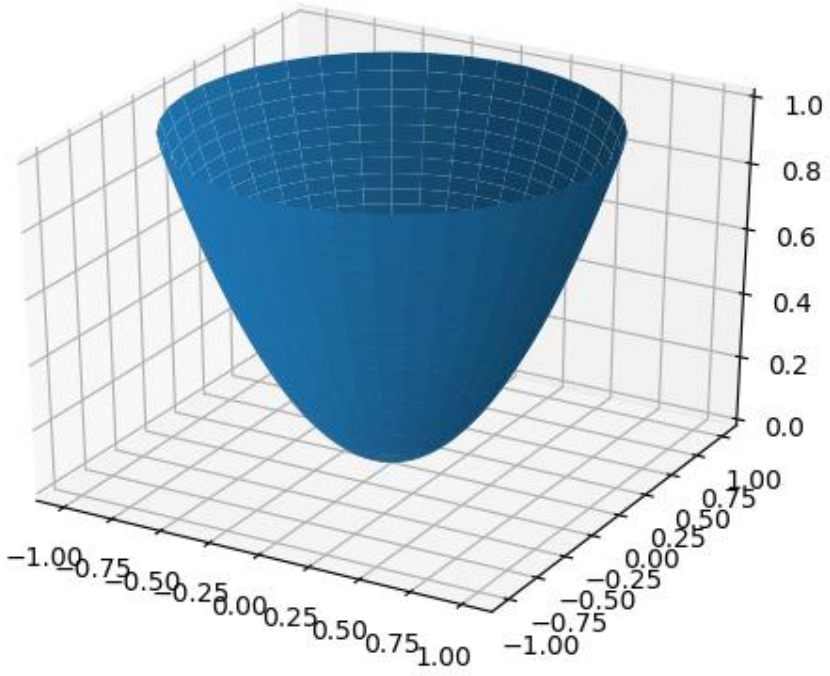
- 3D曲面




```
import numpy as np
import matplotlib.pyplot as plt
import mpl_toolkits.mplot3d
x,y = np.mgrid[-2:2:20j, -2:2:20j]
z = 50 * np.sin(x+y)
ax = plt.subplot(111, projection='3d')
ax.plot_surface(x,y,z,rstride=2,
cstride=1, cmap=plt.cm.Blues_r)
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.show()
```

# 数据可视化

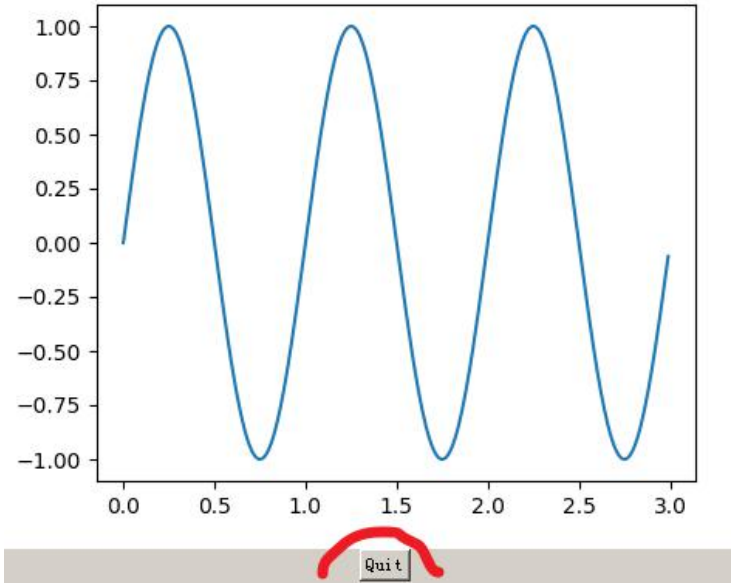
- 3D曲面




```
import pylab as pl
import matplotlib.pyplot as plt
import numpy as np
import mpl_toolkits.mplot3d
rho, theta = np.mgrid[0:1:40j,
0:2*np.pi:40j]
z = rho**2
x = rho*np.cos(theta)
y = rho*np.sin(theta)
ax = pl.subplot(111, projection='3d')
ax.plot_surface(x,y,z)
pl.show()
```

# 数据可视化

- TK结合plot




```
import sys
import tkinter as Tk
import matplotlib
from numpy import arange, sin, pi
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from matplotlib.backend_bases import key_press_handler
from matplotlib.figure import Figure
matplotlib.use('TkAgg')
root = Tk.Tk()
root.title("matplotlib in TK")
#设置图形尺寸与质量
f = Figure(figsize=(5, 4), dpi=100)
a = f.add_subplot(111)
t = arange(0.0, 3, 0.01)
s = sin(2*pi*t)
a.plot(t, s)                #绘制图形
#把绘制的图形显示到tkinter窗口上
canvas = FigureCanvasTkAgg(f, master=root)
canvas.show()
canvas.get_tk_widget().pack(side=Tk.TOP, fill=Tk.BOTH)
#按钮单击事件处理函数
def _quit():
    #结束事件主循环，并销毁应用程序窗口
    root.quit()
    root.destroy()
button = Tk.Button(master=root, text='Quit', command=_quit)
button.pack(side=Tk.BOTTOM)
Tk.mainloop()
```

# 自动问答机器人

- 服务端

```
import socket
words = {'how are you?':'Fine,thank you.',        'bye':'Bye'}
HOST ,PORT = '', 50007
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.bind((HOST, PORT))   #绑定socket
s.listen(1)            #开始监听
print('Listening at port:',PORT)
conn, addr = s.accept() #阻塞，等待连接
print('Connected by', addr)
while True:
    data = conn.recv(1024).decode()
    if not data:
        break
    print('Received message:', data)
    conn.sendall(words.get(data, 'Nothing').encode())
conn.close()
```

# 自动问答机器人

- 客户端

```
import socket

HOST = '127.0.0.1'          #服务端主机IP地址
PORT = 50007                #服务端主机端口号
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))     #连接连接
while True:
    c = input('Input the content you want to send:')
    s.sendall(c.encode())   #发送数据
    data = s.recv(1024).decode()     #从客户端接收数据
    print('Received:', data)
    if c.lower() == 'bye':
        break
s.close()                   #关闭连接
```

# 多线程+自动问答机器人

```
def conTask(x):
    while True:
        data = x.recv(1024).decode()
        if not data:
            x.close()
            break
        print('Received message:', data)
        x.sendall(words.get(data, 'Nothing').encode())

while True:
    #阻塞，等待连接；连接后启动
    conn, addr = s.accept()
    threading.Thread(target = conTask, args=[conn]).start()
```

# 网络聊天室

- 服务端

```
import socket
import threading
HOST ,PORT = '', 50007
s = socket.socket(socket.AF_INET, 、
          socket.SOCK_STREAM)
s.bind((HOST, PORT))    #绑定socket
s.listen(1)             #开始监听
print('Listening at port:',PORT)
conlist=[]
def sendMsg(msg):
    global conlist
    for c in conlist:
        c.sendall(msg.encode())
```

```
def conTask(c):
    while True:
        data = c.recv(1024).decode()
        if not data:
            break
        print('Rx:', data)
        sendMsg(data)
while True:
    #阻塞，等待连接；连接后启动
    conn, addr = s.accept()
    conlist.append(conn)
    threading.Thread(target = conTask,
args=[conn]).start()
```

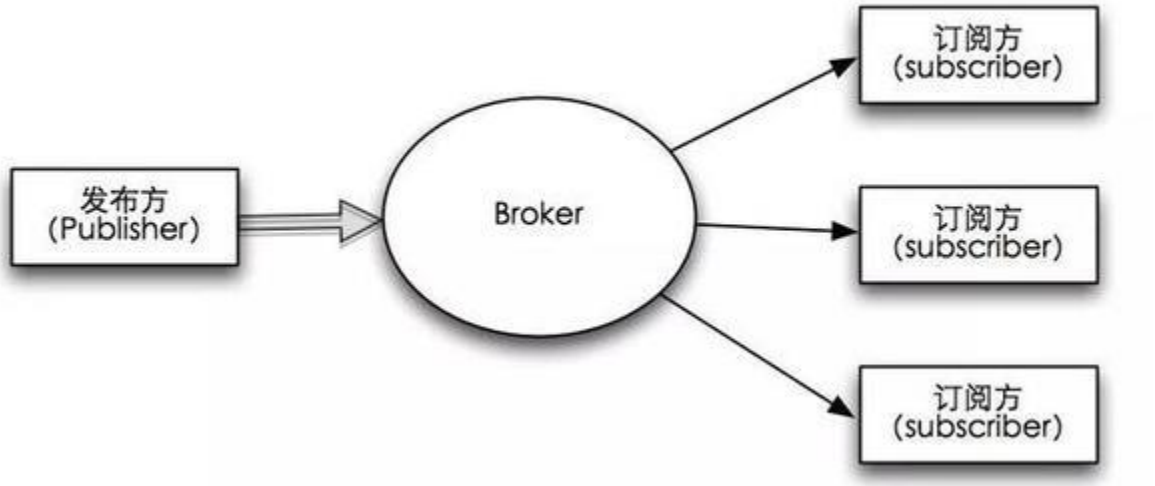
# 网络聊天室

- 客户端

```
import socket
import threading
HOST = '127.0.0.1'          #服务端主机IP地址
PORT = 50007                #服务端主机端口号
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect((HOST, PORT))     #连接连接
def sendMsg(s):
        while True:
            msg = input('>')
            s.sendall(msg.encode())   #发送数据
threading.Thread(target = sendMsg, args=[s]).start()
while True:
    data = s.recv(1024).decode()     #从客户端接收数据
    print('Rx:', data)
```

# MQTT—Message Queuing Telemetry Transport

MQTT(消息队列遥测传输)是ISO 标准
(ISO/IEC PRF 20922)下基于发布/订阅范式的消息协议。



# 群聊天--MQTT服务

```
import paho.mqtt.client as mqtt
import json
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))
    client.subscribe("dddd")
    client.publish("chat", json.dumps({"user": user, "say": "Hello!"}))
def on_message(client, userdata, msg):
    payload = json.loads(msg.payload.decode())
    print(payload.get("user")+":"+payload.get("say"))
```

# 群聊天--MQTT服务

```
if __name__ == '__main__':
    client = mqtt.Client()
    client.username_pw_set("admin", "password") # 必须设置，否则会返回4
    client.on_connect = on_connect
    client.on_message = on_message
    HOST = "115.200.220.105"
    client.connect(HOST, 1883, 60)
    print('connect ok!')
    #client.loop_forever()
    user = input("请输入名称:") #应该是用户名
    client.user_data_set(user)
    client.loop_start()
    while True:
        str = input()
        if str:
            client.publish("dddd", json.dumps({"user": user, "say": str}))
```