```python
#versione 1.0 multithread
#nome del file : pagellaServerMulti_V01.py
import socket
from threading import Thread
import json
import pprint

SERVER_ADDRESS = '127.0.0.1'
SERVER_PORT = 22001


def ricevi_comandi1(sock_service):
    print("avviato")
    while True:
        data=sock_service.recv(1024)
        if not data:
            break
        data=data.decode()
        data=json.loads(data)
        # completare:
        #1. recuperare studente, materia, voto e assenze
        studente=data['studente']
        materia=data['materia']
        voto=int(data['voto'])
        assenze=int(data['assenze'])

        #2. restituire studente, materia e una valutazione testuale sapendo che :
        # voto < 4 Gravemente insufficiente
        # voto <= 5 Insufficiente
        # voto = 6 Sufficiente
        # voto = 7 Discreto
        # voto compreso tra 8 e 9 Buono
        # voto = 10 Ottimo
        if voto<4:
            valutazione="Gravemente insufficiente"
        elif voto<=5:
            valutazione="Insufficiente"
        elif voto==6:
            valutazione="Sufficiente"
        elif voto==7:
            valutazione="Discreto"
        elif voto>=8 and voto<=9:
            valutazione="Buono"
        elif voto==10:
            valutazione="Ottimo"
        messaggio={'studente':studente,
        'materia':materia,
        'valutazione':valutazione}
        print("Dati inviati al client:")
        print(messaggio)
        messaggio=json.dumps(messaggio)
        sock_service.sendall(messaggio.encode("UTF-8"))
    sock_service.close()
```

```python
def ricevi_comandi2(sock_service):
    print("avviato")
    while True:
        data=sock_service.recv(1024)
        if not data:
            break
        data=data.decode()
        data=json.loads(data)
        # completare:
        #1. recuperare studente e pagella
        studente=data['studente']
        pagella=data['pagella']
        #2. restituire studente, media dei voti e somma delle assenze :
        assenze=0
        media=0
        for i,p in enumerate(pagella):
            media+=int(p[1])
            assenze+=int(p[2])
        media=media/i
        messaggio={'studente':studente,
        'media':media,
        'assenze':assenze}
        print("Dati inviati al client:")
        print(messaggio)
        messaggio=json.dumps(messaggio)
        sock_service.sendall(messaggio.encode("UTF-8"))
    sock_service.close()
```

```python
def ricevi_comandi3(sock_service):
    print("avviato")
    while True:
        data=sock_service.recv(1024)
        if not data:
            break
        data=data.decode()
        data=json.loads(data)
        # completare:
        #1. recuperare il tabellone
        pp=pprint.PrettyPrinter(indent=4)
        #2. restituire una lista di dizionari : studente, media dei voti e somma delle assenze
        tabellone=[]
        for stud in data:
            pagella=data[stud]
            assenze=0
            media=0
            for i,p in enumerate(pagella):
                media+=int(p[1])
                assenze+=int(p[2])
            media=media/i
            messaggio={'studente':stud,
            'media':media,
            'assenze':assenze}
            tabellone.append(messaggio)
        print("Dati inviati al client:")
        pp.pprint(tabellone)
        messaggio=tabellone
        messaggio=json.dumps(messaggio)
        sock_service.sendall(messaggio.encode("UTF-8"))
    sock_service.close()
```

```
115  def ricevi_connessioni(sock_listen):
116      while True:
117          sock_service, addr_client = sock_listen.accept()
118          print("\nConnessione ricevuta da " + str(addr_client))
119          print("\nCreo un thread per servire le richieste ")
120          try:
121              Thread(target=ricevi_comandi1,args=(sock_service,addr_client)).start()
122              #Thread(target=ricevi_comandi2,args=(sock_service,addr_client)).start()
123              #Thread(target=ricevi_comandi3,args=(sock_service,addr_client)).start()
124          except:
125              print("il thread non si avvia")
126              sock_listen.close()
127
128  def avvia_server(SERVER_ADDRESS,SERVER_PORT):
129      sock_listen=socket.socket()
130      sock_listen.setsockopt(socket.SOL_SOCKET,socket.SO_REUSEADDR,1)
131      sock_listen.bind((SERVER_ADDRESS,SERVER_PORT))
132      sock_listen.listen(5)
133      print("Server in ascolto su %s." %str((SERVER_ADDRESS,SERVER_PORT)))
134      ricevi_connessioni(sock_listen)
135
136  if __name__=='__main__':
137      avvia_server(SERVER_ADDRESS,SERVER_PORT)
```