```python
import socket
from threading import Thread
SERVER_ADDRESS = '127.0.0.1'
SERVER_PORT = 22225
class Client():
    """
    Questa classe rappresenta una persona che opera come client
    """
    def connessione_server(self,address,port):
        """
        Metodo per stabilire la connessione con il server
        """
        sock_service = socket.socket()
        sock_service.connect((address, port))
        print("Connesso a " + str((address, port)))
        return sock_service

    def invia_comandi(self,sock_service):
        """
        Metodo per inviare le richieste di servizio e ricevere le risposte
        """
        while True:
            try:
                n1 = input("Inserisci il primo numero: ")
                n2 = input("Inserisci il secondo numero: ")
                oper = input("Inserisci l'operazione da effettuare(piu / meno / per / diviso): ")
                dati=f"{oper};{n1};{n2}"
            except EOFError:
                print("\nOkay. Exit")
                break
            if not dati:
                print("Non puoi inviare una stringa vuota!")
                continue
            if dati == '0':
                print("Chiudo la connessione con il server!")
                sock_service.close()
                break

            dati = dati.encode()
            sock_service.sendall(dati)
            dati = sock_service.recv(2048)
            if not dati:
                print("Server non risponde. Exit")
                break
            dati = dati.decode()
            print("Ricevuto dal server:")
            print(dati + '\n')

c1=Client()
sock_serv=c1.connessione_server(SERVER_ADDRESS,SERVER_PORT)
c1.invia_comandi(sock_serv)
```

```python
import socket
from threading import Thread
SERVER_ADDRESS='127.0.0.1'
SERVER_PORT=22225
class Server():
    """
    Questa classe rappresenta un server
    """
    def __init__(self, address, port):
        self.address = address
        self.port=port

    def avvia_server(self):
        """
        Metodo per aprirsi e mettersi in ascolto aspettando richieste da servire
        """
        sock_listen = socket.socket()
        sock_listen.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
        sock_listen.bind((self.address, self.port))
        sock_listen.listen(5)
        print("Server in ascolto su %s." % str((self.address, self.port)))
        return sock_listen

    def accetta_connessioni(self,sock_listen):
        """
        Metodo per accettare richieste di servizio ed assegnare un Thread ad ognuna di esse
        """
        while True:
            sock_service, addr_client = sock_listen.accept()
            print("\nConnessione ricevuta da " + str(addr_client))
            print("\nCreo un thread per servire le richieste ")
            try:
                Thread(target=self.ricevi_comandi, args=(sock_service,addr_client)).start()
            except:
                print("il thread non si avvia")
                sock_listen.close()

    def ricevi_comandi(self,sock_service,addr_client):
        """
        Metodo per ricevere i comandi e servire le richieste ricevute
        """
        print("avviato")
        while True:
            dati = sock_service.recv(2048)
            if not dati:
                print("Fine dati dal client. Reset")
                break
            dati = dati.decode()
            print("Ricevuto: '%s'" % dati)
            if dati=='0':
                print("Chiudo la connessione con " + str(addr_client))
                break
            risultato=0
            oper,n1,n2= dati.split(";")
            if oper=="piu":
                risultato=int(n1)+int(n2)

            if oper=="meno":
                risultato=int(n1)-int(n2)

            if oper=="per":
                risultato=int(n1)*int(n2)

            if oper=="diviso":
                risultato=int(n1)/int(n2)

            dati = f"Risposta a : {str(addr_client)}. Il risultato dell'operazione({n1} {oper} {n2}) è :{risultato} "
            dati = dati.encode()
            sock_service.send(dati)
        sock_service.close()

s1=Server(SERVER_ADDRESS,SERVER_PORT)
sock_lis=s1.avvia_server()
s1.accetta_connessioni(sock_lis)

```