الجمهورية العربية السورية

اللاذقية جامعة تشرين

كلية الهندسة الميكانيكية والكهربائية

قسم هندسة الاتصالات والالكترونيات

السنة الخامسة

اشراف الدكتور:

مهند عيسى

اعداد الطالبين :

يزن طالب قيراطة 2706

أدهم صلاح طه 2602

Question 1 : server.py

```python
import socket
import threading
accounts = {
    "02706": 1000.0,
    "02602": 5000.0,
    "01234": 500.0
}
def get_balance(account_number):
    return accounts[account_number]

def deposit(account_number, amount):
    accounts[account_number] += amount
    return accounts[account_number]

def withdraw(account_number, amount):
    if accounts[account_number] >= amount:
        accounts[account_number] -= amount
        return accounts[account_number]
    else:
        return "you dont have enough money"

HOST = '127.0.0.1'
PORT = 8000

server_socket = socket.socket()
server_socket.bind((HOST, PORT))
server_socket.listen(5)

print(f"Server is listening on {HOST}:{PORT}")

def handle_client(cc, cadd):
    print(f"New connection from {cadd}")

    # Authenticate the client
    account_number = cc.recv(1024).decode()
    password = cc.recv(1024).decode()
    # Verify the account number and password
    if account_number in accounts and password == "0000":
        cc.send("Authentication successful".encode())
    else:
        cc.send("Authentication failed".encode())
        cc.close()
        return

    while True:
        operation = cc.recv(1024).decode()
        if operation == "balance":
            balance = get_balance(account_number)
            cc.send(str(balance).encode())
        elif operation == "deposit":
            amount = float(cc.recv(1024).decode())
            new_balance = deposit(account_number, amount)
            cc.send(str(new_balance).encode())
        elif operation == "withdraw":
            amount = float(cc.recv(1024).decode())
            result = withdraw(account_number, amount)
            cc.send(str(result).encode())
        elif operation == "exit":
            cc.send(str(get_balance(account_number)).encode())
            cc.close()
            print("Connection with", cadd, "closed")
            break
        else:
            pass

while True:
    cc, cadd = server_socket.accept()
    client_thread = threading.Thread(target=handle_client, args=(cc, cadd))
    client_thread.start()


```

Client.py

```python
import socket

HOST = '127.0.0.1'
PORT = 8000

csock= socket.socket()
csock.connect((HOST, PORT))

account_number = input("Enter your account number: ")
password = input("Enter your password: ")

csock.send(account_number.encode())
csock.send(password.encode())

response = csock.recv(1024).decode()
if response == "Authentication successful":
    print("Authentication successful")
else:
    print("Authentication failed")
    csock.close()
    exit()

while True:
    operation = input("Enter operation (balance, deposit, withdraw, exit): ")
    csock.send(operation.encode())

    if operation == "balance":
        balance = float(csock.recv(1024).decode())
        print("Your balance is:",balance)
    elif operation == "deposit":
        amount = float(input("Enter deposit amount: "))
        csock.send(str(amount).encode())
        new_balance = float(csock.recv(1024).decode())
        print("New balance:",new_balance)
    elif operation == "withdraw":
        amount = float(input("Enter withdrawal amount: "))
        csock.send(str(amount).encode())
        result = csock.recv(1024).decode()
        print("your balance now is :",result)
    elif operation == "exit":
        final_balance = float(csock.recv(1024).decode())
        print("Final balance:",final_balance)
        csock.close()
        break
    else:
        print("Invalid operation")
```

: server.py

هذا الملف يحتوي في البداية على تعريف الحسابات المصرفية وتقديم خدمات لإدارة أرصدة الحسابات.

- تم تعريف قاموس accounts الذي يحتوي على أرقام الحسابات والأرصدة المرتبطة بها.
- تم تعريف ثلاث توابع :

- get_balance(account_number) :
يعطي الرصيد الحالي للحساب.
- deposit(account_number , amount) :
تقوم بايداع مبلغ في الحساب المحدد وتعيد الرصيد الجديد.
- withdraw(account_number , amount) :
تقوم بسحب مبلغ معين من الحساب المحدد وتعيد الرصيد الجديد او اذا كان المبلغ المطلوب سحبه اكبر من الرصيد الحالي يتم طباعة (ليس لديك مال كافي).

- ثم ننشئ خادم TCP والذي سيتعامل مع الاتصالات الواردة من العملاء.
- تم تعريف عناوين الخادم والمنفذ المستخدمين(host,port) .
- تم إنشاء مقبس خادم TCP وربطه بالعنوان والمنفذ المحددين.
- تم تعريف تابع handle_client(cc , cadd) لتعامل مع كل اتصال عميل جديد:

- تقوم بطباعة معلومات الاتصال الجديد.
- تقوم بمصادقة العميل باستخدام رقم الحساب وكلمة المرور
- بعد المصادقة الناجحة، تبدأ حلقة لتنفيذ العمليات المصرفية (الرصيد والإيداع والسحب والخروج).
- الخادم يستخدم Threading لمعالجة اتصالات العملاء المتعددة بشكل متوازٍ.

: client.py

- هذا الملف ينشئ عميل TCP ويتفاعل مع المستخدم لتنفيذ العمليات المصرفية.
- تم إنشاء مقبس عميل TCP والاتصال بالخادم.
- يطلب من المستخدم إدخال رقم الحساب وكلمة المرور للمصادقة.
- بعد المصادقة الناجحة، يبدأ حلقة للسماح للمستخدم بتنفيذ العمليات المصرفية (الرصيد والإيداع والسحب والخروج).
- يرسل طلبات العمليات إلى الخادم وتستقبل الردود وتعرضها على المستخدم.
- عند اختيار "خروج"، يستقبل الرصيد النهائي من الخادم قبل إغلاق الاتصال.

خرج ال Server :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Windows.10> & C:/Python312/python.exe c:/Users/Windows.10/Desktop/server.py
Server is listening on 127.0.0.1:8000
New connection from ('127.0.0.1', 49691)
Connection with ('127.0.0.1', 49691) closed
New connection from ('127.0.0.1', 49692)
Connection with ('127.0.0.1', 49692) closed
New connection from ('127.0.0.1', 49693)
Connection with ('127.0.0.1', 49693) closed
```

الخرج لثلاثة client :

Client1 :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Windows.10> & C:/Python312/python.exe c:/Users/Windows.10/Desktop/client.py
Enter your account number: 02602
Enter your password: 0000
Authentication successful
Enter operation (balance, deposit, withdraw, exit): balance
Your balance is: 5000.0
Enter operation (balance, deposit, withdraw, exit): deposit
Enter deposit amount: 200
New balance: 5200.0
Enter operation (balance, deposit, withdraw, exit): withdraw
Enter withdrawal amount: 200
your balance now is : 5000.0
Enter operation (balance, deposit, withdraw, exit): exit
Final balance: 5000.0
PS C:\Users\Windows.10>
```

Client2 :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Windows.10> & C:/Python312/python.exe c:/Users/Windows.10/Desktop/client.py
Enter your account number: 02706
Enter your password: 0000
Authentication successful
Enter operation (balance, deposit, withdraw, exit): balance
Your balance is: 1000.0
Enter operation (balance, deposit, withdraw, exit): deposit
Enter deposit amount: 500
New balance: 1500.0
Enter operation (balance, deposit, withdraw, exit): withdraw
Enter withdrawal amount: 300
your balance now is : 1200.0
Enter operation (balance, deposit, withdraw, exit): exit
Final balance: 1200.0
PS C:\Users\Windows.10>
```

Client3 :

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

PS C:\Users\Windows.10> & C:/Python312/python.exe c:/Users/Windows.10/Desktop/client.py
Enter your account number: 01234
Enter your password: 0000
Authentication successful
Enter operation (balance, deposit, withdraw, exit): balance
Your balance is: 500.0
Enter operation (balance, deposit, withdraw, exit): deposit
Enter deposit amount: 300
New balance: 800.0
Enter operation (balance, deposit, withdraw, exit): withdraw
Enter withdrawal amount: 200
your balance now is : 600.0
Enter operation (balance, deposit, withdraw, exit): exit
Final balance: 600.0
PS C:\Users\Windows.10>
```

Question 2:

From 140 Python Projects with Source Code
o https://medium.datadriveninvestor.com/140-python-projects-with-source-code-fa12c9e2aeac

Animal Quiz Games: https://thecleverprogrammer.com/2021/01/03/create-a-quiz-game-with-python/

```python
def check_guess(guess, answer):
    global score
    still_guessing = True
    attempt = 0
    while still_guessing and attempt < 3:
        if guess.lower() == answer.lower():
            print("Correct Answer")
            score = score + 1
            still_guessing = False
        else:
            if attempt < 2:
                guess = input("Sorry Wrong Answer, try again")
            attempt = attempt + 1
    if attempt == 3:
        print("The Correct answer is ",answer )

score = 0
print("Guess the Animal")
guess1 = input("Which bear lives at the North Pole? ")
check_guess(guess1, "polar bear")
guess2 = input("Which is the fastest land animal? ")
check_guess(guess2, "Cheetah")
guess3 = input("Which is the larget animal? ")
check_guess(guess3, "Blue Whale")
print("Your Score is "+ str(score))
```

في البداية نقوم بتعريف تابع check_guess يأخذ بارمترين هما إجابة المستخدم والاجابة الصحيحة ويقوم بمقارنة الاجابتين اذا كانت صحيحة يزيد السكور بمقدار واحد ويخرج من الحلقة while واذا كانت خاطئة يطلب من المستخدم ادخال إجابة أخرى واذا اخطأ ثلاث مرات يعطيه الإجابة الصحصحة

وبعدها يطلب من المستخدم الاجابة على الأسئلة وتتم المقارنة بواسطة التابع check_guess وفي النهاية تتم طباعة السكور الذي يحوي عدد الإجابات الصحيحة

خرج السؤال الثانى :