Ira A. Fulton Schools of Engineering
Arizona State University

CSE 531: Distributed and Multiprocessor Operating Systems

gRPC Written Report

Problem Statement

This project involves the development of a distributed banking system comprising customer and branch processes. Each of these processes is expected to provide functions for depositing, withdrawing, and querying account balances. However, the primary challenge is to ensure that these processes are not only self-contained but also capable of inter-process communication to synchronize account balances across multiple branches.

Goal

The key goals are as follows:

1. **Customer and Branch Processes:**

The system necessitates the creation of two essential components: **customers** and **branches**. Customers are responsible for executing banking transactions, including deposits, withdrawals, and balance queries. Branches, on the other hand, are responsible for maintaining account balances, handling customer transactions, and communicating with other branches to keep all balances synchronized.

2. **Inter-Process Communication:**

To ensure that all branch processes have synchronized and up-to-date account balances, it is important to establish effective communication between them. This includes propagating deposit and withdrawal transactions to all relevant branches to maintain consistent balance records.

3. **Synchronization:**

A fundamental problem in this project is achieving synchronization of account balances across different branches. It involves devising a mechanism that updates balances in real-time after each customer process, maintaining consistency and accuracy across the distributed banking system.

Setup

For this project, I use gRPC and Python. gRPC is a high-performance, open-source remote procedure call (RPC) framework. It facilitates communication between distributed systems and is widely used in building efficient and scalable microservices. Python is a versatile, high-level programming language that provides simplicity and ease of use. For this project, Python 3.12

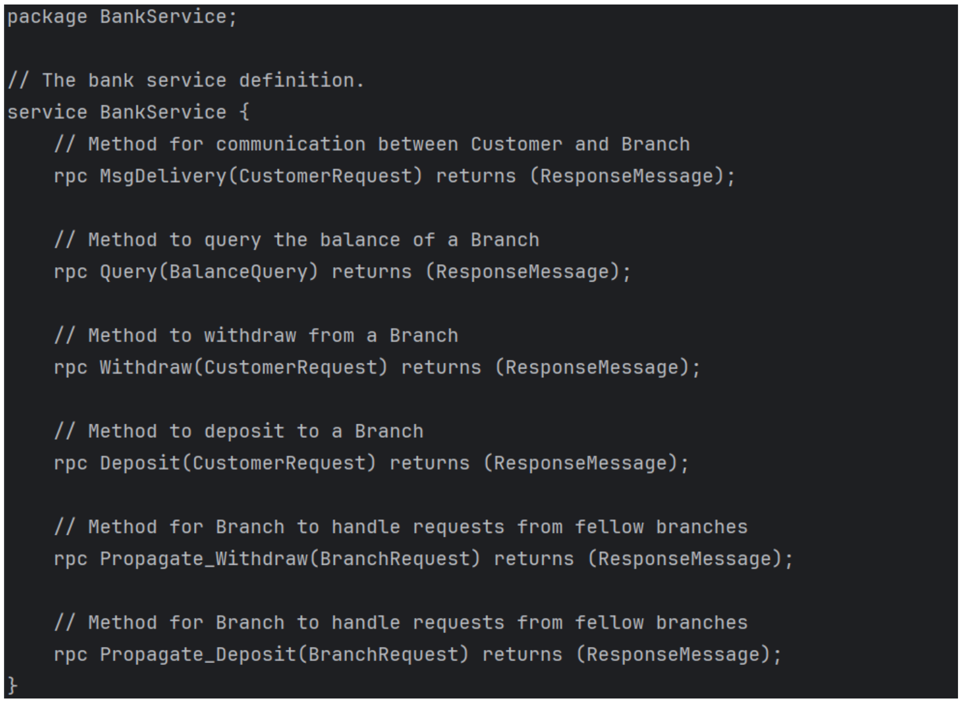
version is used, as it offers support for the latest features and libraries. Setup steps are shown as follows:

1. **Install the gRPC library by running the following command:**

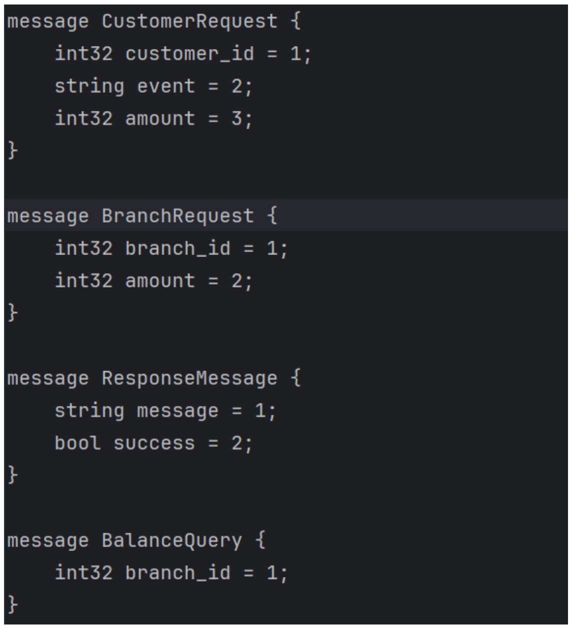
Create a Python virtual environment to manage project dependencies if needed:

2. **Define Protocol Buffers (Proto) File:**

Create a bankService.proto file that defines the service interface and message types. In the banking system, this file specifies the methods for deposit, withdraw, query, and any other relevant services.



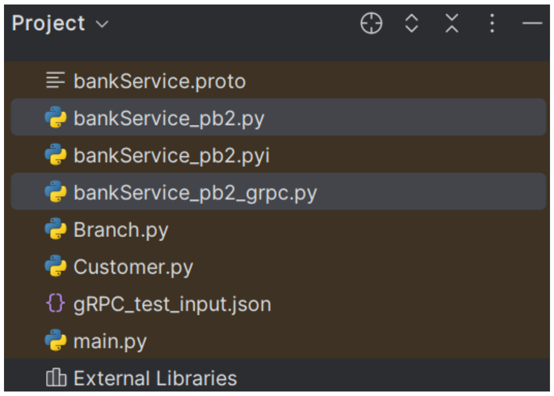
```
package BankService;

// The bank service definition.
service BankService {
    // Method for communication between Customer and Branch
    rpc MsgDelivery(CustomerRequest) returns (ResponseMessage);

    // Method to query the balance of a Branch
    rpc Query(BalanceQuery) returns (ResponseMessage);

    // Method to withdraw from a Branch
    rpc Withdraw(CustomerRequest) returns (ResponseMessage);

    // Method to deposit to a Branch
    rpc Deposit(CustomerRequest) returns (ResponseMessage);

    // Method for Branch to handle requests from fellow branches
    rpc Propagate_Withdraw(BranchRequest) returns (ResponseMessage);

    // Method for Branch to handle requests from fellow branches
    rpc Propagate_Deposit(BranchRequest) returns (ResponseMessage);
}
```



```
message CustomerRequest {
    int32 customer_id = 1;
    string event = 2;
    int32 amount = 3;
}

message BranchRequest {
    int32 branch_id = 1;
    int32 amount = 2;
}

message ResponseMessage {
    string message = 1;
    bool success = 2;
}

message BalanceQuery {
    int32 branch_id = 1;
}
```

3. **Generate gRPC Code:**

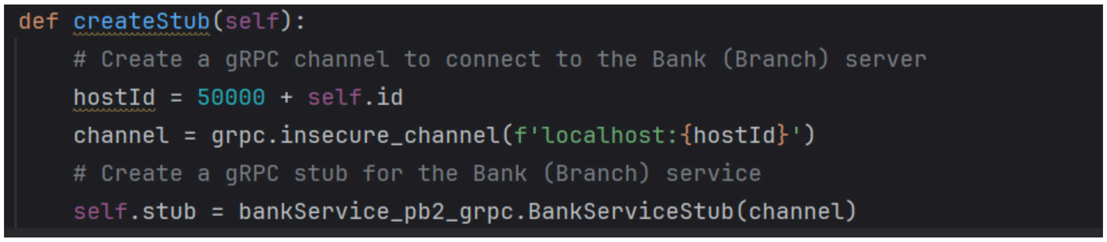
Compile the **bankService.proto** file to generate **bankService_pb2.py** and **bankService_pb2_grpc.py** by using the protocol compiler. Those two files include all the methods that can be used in class files and main process files.



Implementation Processes

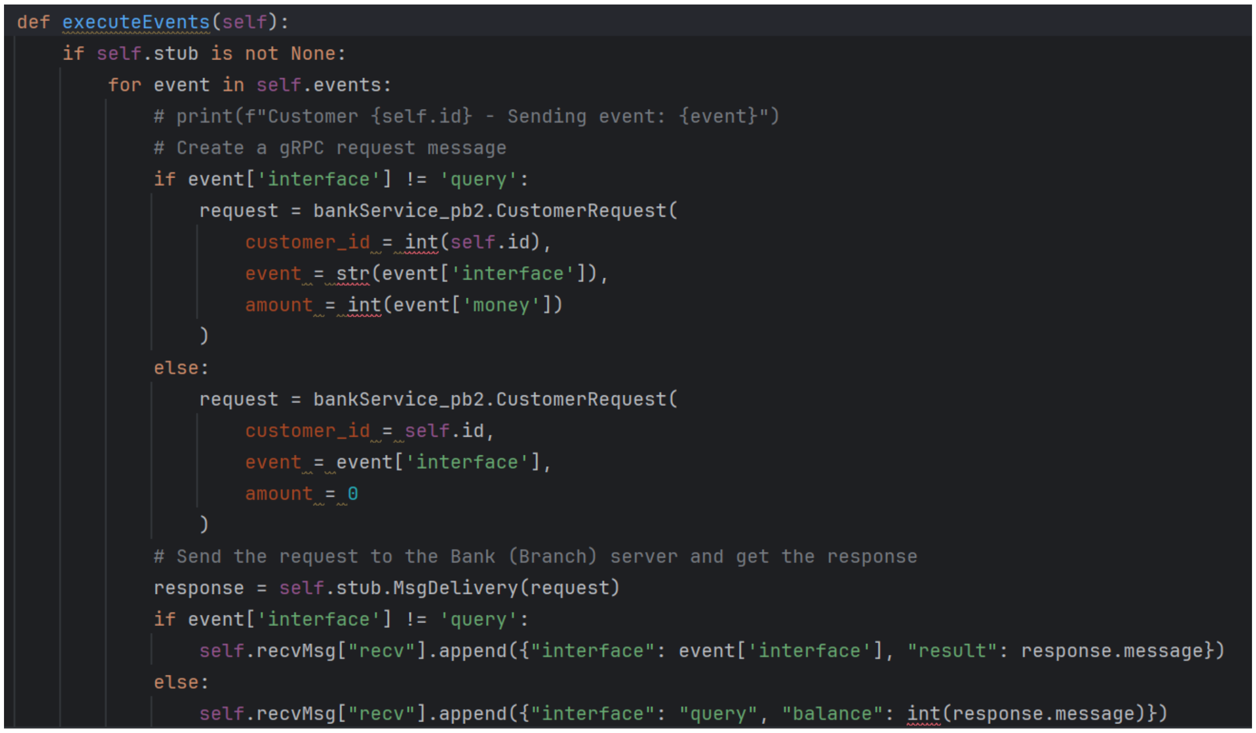
1. **Processes Between Customer and Branch:**

The Customer class is responsible for customer processes. It creates a gRPC stub to connect to the Bank (Branch) server using a specified channel.



```
def createStub(self):
    # Create a gRPC channel to connect to the Bank (Branch) server
    hostId = 50000 + self.id
    channel = grpc.insecure_channel(f'localhost:{hostId}')
    # Create a gRPC stub for the Bank (Branch) service
    self.stub = bankService_pb2_grpc.BankServiceStub(channel)
```

The executeEvents method of the Customer class sends customer events, such as withdrawals, deposits, and queries, to the Bank (Branch) server. It creates gRPC request messages based on the events and sends them to the server.

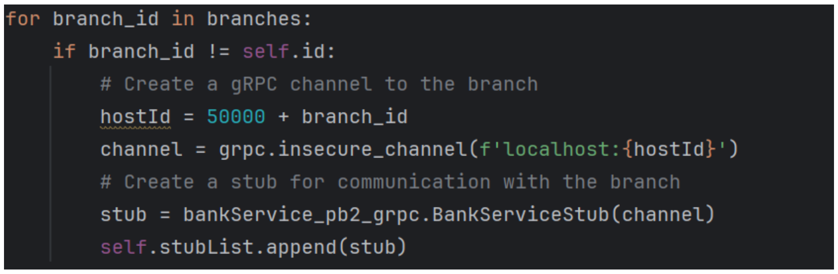


```python
def executeEvents(self):
    if self.stub is not None:
        for event in self.events:
            # print(f"Customer {self.id} - Sending event: {event}")
            # Create a gRPC request message
            if event['interface'] != 'query':
                request = bankService_pb2.CustomerRequest(
                    customer_id = int(self.id),
                    event = str(event['interface']),
                    amount = int(event['money'])
                )
            else:
                request = bankService_pb2.CustomerRequest(
                    customer_id = self.id,
                    event = event['interface'],
                    amount = 0
                )
            # Send the request to the Bank (Branch) server and get the response
            response = self.stub.MsgDelivery(request)
            if event['interface'] != 'query':
                self.recvMsg["recv"].append({"interface": event['interface'], "result": response.message})
            else:
                self.recvMsg["recv"].append({"interface": "query", "balance": int(response.message)})
```

The customer processes communicate with the Bank (Branch) server by invoking the MsgDelivery method of the server using the gRPC stub. This method processes customer requests and returns responses, which are stored in the recvMsg list for debugging purposes.

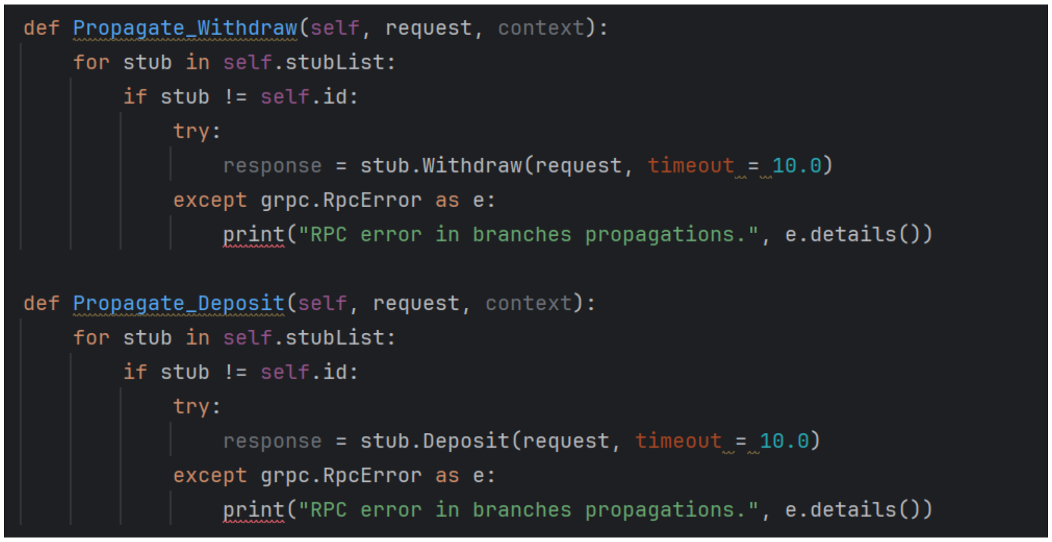
2. **Processes Between Branch and Branch:**

The Branch class is responsible for branch processes. It creates gRPC channels to connect with other branch servers. These channels are used to create gRPC stubs for communication.



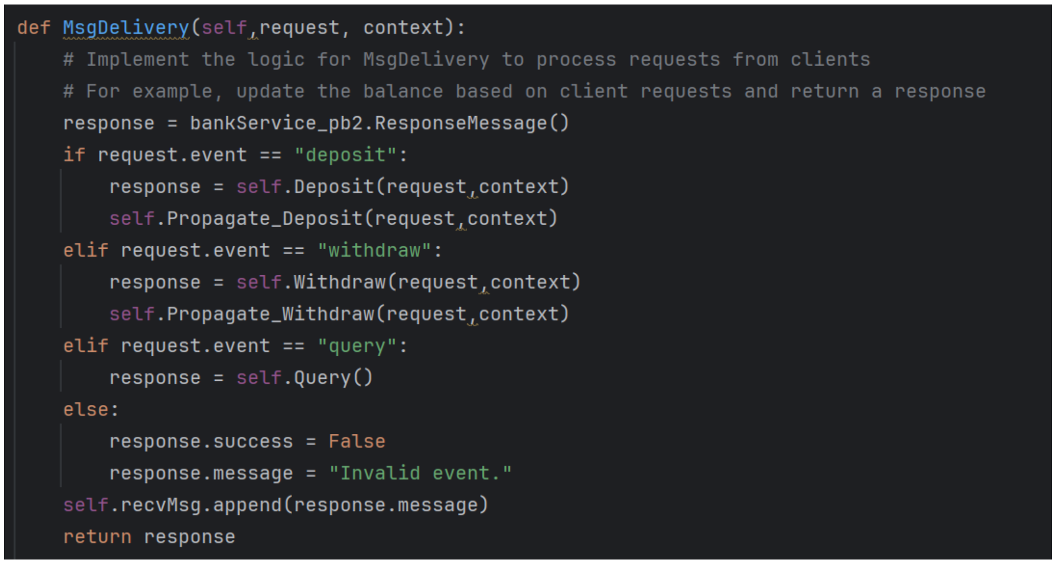
```python
for branch_id in branches:
    if branch_id != self.id:
        # Create a gRPC channel to the branch
        hostId = 50000 + branch_id
        channel = grpc.insecure_channel(f'localhost:{hostId}')
        # Create a stub for communication with the branch
        stub = bankService_pb2_grpc.BankServiceStub(channel)
        self.stubList.append(stub)
```

The Propagate_Withdraw and Propagate_Deposit methods in the Branch class are responsible for propagating withdrawal and deposit requests to other branch servers. These methods use gRPC to send requests to other branch servers using their respective stubs.



```
def Propagate_Withdraw(self, request, context):
    for stub in self.stubList:
        if stub != self.id:
            try:
                response = stub.Withdraw(request, timeout = 10.0)
            except grpc.RpcError as e:
                print("RPC error in branches propagations.", e.details())

def Propagate_Deposit(self, request, context):
    for stub in self.stubList:
        if stub != self.id:
            try:
                response = stub.Deposit(request, timeout = 10.0)
            except grpc.RpcError as e:
                print("RPC error in branches propagations.", e.details())
```

The MsgDelivery method of the Branch class handles requests from customers and processes them by updating the balance based on customer requests. It also propagates deposit and withdrawal requests to other branch servers for synchronization.



```
def MsgDelivery(self,request, context):
    # Implement the logic for MsgDelivery to process requests from clients
    # For example, update the balance based on client requests and return a response
    response = bankService_pb2.ResponseMessage()
    if request.event == "deposit":
        response = self.Deposit(request,context)
        self.Propagate_Deposit(request,context)
    elif request.event == "withdraw":
        response = self.Withdraw(request,context)
        self.Propagate_Withdraw(request,context)
    elif request.event == "query":
        response = self.Query()
    else:
        response.success = False
        response.message = "Invalid event."
    self.recvMsg.append(response.message)
    return response
```

3. Execution

The main.py script is the entry point of your project and coordinates the execution of customer and branch processes.

It reads input data from a JSON file named **'gRPC_test_input.json'** to determine the configuration of branches and customers. The file contains information about branch IDs, initial balances, and events for each customer. It then iterates through the customer data and identifies branch-related information.

For each branch defined in the input data, it creates a gRPC server and starts the gRPC server for each branch on a specified port based on the branch's ID and adds the server instance to the **branch_servers** list.
It then iterates the customer data. For each customer defined in the input data, it creates a Customer object, passing its ID and a list of events. It initializes a gRPC stub for the customer which connects the customer to the branch server.
The it loops through each customer in **customerList** and executes the events defined in the input data by calling the **customer.executeEvents()** method.
After all customer processes are completed, the script stops the gRPC servers for the branches to terminate the simulation.
Finally, it prints the received messages for each customer, which include information about the events they executed, and the responses received.

Results

The printed messages show each customer's id and received message after each event has been processed. It shows all withdraw or deposit events have been processed successfully for each customer, and also the updated balance of the branch the customer is associated with. See the output results below.

```
{'id': 1, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 410}]}
{'id': 2, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 420}]}
{'id': 3, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 430}]}
{'id': 4, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 440}]}
{'id': 5, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 450}]}
{'id': 6, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 460}]}
{'id': 7, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 470}]}
{'id': 8, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 480}]}
{'id': 9, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 490}]}
{'id': 10, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 500}]}
{'id': 11, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 510}]}
{'id': 12, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 520}]}
{'id': 13, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 530}]}
{'id': 14, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 540}]}
{'id': 15, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 550}]}
{'id': 16, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 560}]}
{'id': 17, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 570}]}
{'id': 18, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 580}]}
{'id': 19, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 590}]}
{'id': 20, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 600}]}
{'id': 21, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 610}]}
{'id': 22, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 620}]}
```

```
{'id': 23, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 630}]}
{'id': 24, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 640}]}
{'id': 25, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 650}]}
{'id': 26, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 660}]}
{'id': 27, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 670}]}
{'id': 28, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 680}]}
{'id': 29, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 690}]}
{'id': 30, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 700}]}
{'id': 31, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 710}]}
{'id': 32, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 720}]}
{'id': 33, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 730}]}
{'id': 34, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 740}]}
{'id': 35, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 750}]}
{'id': 36, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 760}]}
{'id': 37, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 770}]}
{'id': 38, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 780}]}
{'id': 39, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 790}]}
{'id': 40, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 800}]}
{'id': 41, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 810}]}
{'id': 42, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 820}]}
{'id': 43, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 830}]}
{'id': 44, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 840}]}
{'id': 45, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 850}]}
{'id': 46, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 860}]}
{'id': 47, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 870}]}
{'id': 48, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 880}]}
{'id': 49, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 890}]}
{'id': 50, 'recv': [{'interface': 'deposit', 'result': 'success'}, {'interface': 'query', 'balance': 900}]}
{'id': 51, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 890}]}
{'id': 52, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 880}]}
{'id': 53, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 870}]}
{'id': 54, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 860}]}
{'id': 55, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 850}]}
{'id': 56, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 840}]}
{'id': 57, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 830}]}
{'id': 58, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 820}]}
{'id': 59, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 810}]}
{'id': 60, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 800}]}
{'id': 61, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 790}]}
{'id': 62, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 780}]}
{'id': 63, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 770}]}
{'id': 64, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 760}]}
{'id': 65, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 750}]}
```

```
{'id': 66, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 740}]}
{'id': 67, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 730}]}
{'id': 68, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 720}]}
{'id': 69, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 710}]}
{'id': 70, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 700}]}
{'id': 71, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 690}]}
{'id': 72, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 680}]}
{'id': 73, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 670}]}
{'id': 74, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 660}]}
{'id': 75, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 650}]}
{'id': 76, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 640}]}
{'id': 77, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 630}]}
{'id': 78, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 620}]}
{'id': 79, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 610}]}
{'id': 80, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 600}]}
{'id': 81, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 590}]}
{'id': 82, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 580}]}
{'id': 83, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 570}]}
{'id': 84, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 560}]}
{'id': 85, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 550}]}
{'id': 86, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 540}]}
{'id': 87, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 530}]}
{'id': 88, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 520}]}
{'id': 89, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 510}]}
{'id': 90, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 500}]}
{'id': 91, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 490}]}
{'id': 92, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 480}]}
{'id': 93, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 470}]}
{'id': 94, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 460}]}
{'id': 95, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 450}]}
{'id': 96, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 440}]}
{'id': 97, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 430}]}
{'id': 98, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 420}]}
{'id': 99, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 410}]}
{'id': 100, 'recv': [{'interface': 'withdraw', 'result': 'success'}, {'interface': 'query', 'balance': 400}]}
```