Team Name: Trustless_Trust

Team Members: - Nityam Gupta(230001058)
- Nitin (230001059)
- Ajay Sonawani (230001004)

Part 1: Legacy (P2PKH) Address Transactions

For this part, we have created 2 Python codes, Legacy_1.py and Legacy_2.py

- Legacy_1.py

Outputs :

```
Legacy Addresses:
Sender: mir9Z1g54bZQdDp3SKndzZfPrLH2RQyDDC
Receiver: mfrVyD5imKR9EzuvJrDZvVk1BIxzTDUB4h
Change: mfnS29rRqeBFybrD7z7UFCvAs36Zbiufh9

Mining some initial blocks to fund sender address ...

Balance of Sender: 9641.86041242 BTC
UTXO of Sender: 60.0 BTC

Enter the amount to send from sender to receiver (max 59.9999000 BTC): 50

Creating a raw transaction from Sender to Reciever ...

Unsigned raw transaction hex:
0200000001621d364af14538a65f743dde4e1a825d70764b956677a723e82a96b2d5c26b580000000000fdffffff0200943
577000000001976a914bed867320f53016a64caa842778ff4f098e8255b88acf036deb2000000001976a914befee71dba42
db838583b6089c2d63d28e1312e188ac00000000

Decoding raw transaction to extract the challenge script ...

Extracted ScriptPubKey: 76a914bed836920f53016a64caa842778ff4f098e8255b88ac
Script size: 28 vbytes

Signing the transaction Sender -> Reciever ...

Signed transaction hex:
0200000001621d364af14538a65f743dde4e1d825d70764b956677a723e82a96b2d5c26b58000000006a47384402204b48d
3d301eda3772681cb246dee760379b9fe40ecbae24e6a6b6e78062420f02201670d5107755373a3a3f6ac187be8ceeea2c
210472822d43026d8666fef33b012103c8f34d7379c5998143ac1716cd67bba325a477ec5594ea721cf5b897ab5992e5f
ffffffff0200943577000000001976a914bed836920f53016a64caa842778ff4f098e8255b88acf036d0b2000000001976a9
14befee71dba42db838583b6089c2d63d28e1312e188ac00000000

Broadcasting the transaction Sender -> Reciever ...

Transaction ID (Sender -> Reciever): 82f14104bf6757a7a08d90ce6b16aad7f69e6d10bf42d4df569831388356efed
Transaction size: 231 vbytes

Unloaded wallet: Synergy_Legacy
```

Explanation :

a) Create a new wallet (or load existing wallet) named Synergy_Legacy

b) Generate 3 legacy addresses A, B and C

c) Mine some initial blocks in order to fund address A

d) Display the UTXO balance of A once it is funded

e) Ask the user the amount to be transferred from A to B, satisfying the condition:
$0 < \text{Amount} \leq \text{UTXO}(A) - \text{Mining fee}$

f) Create a raw transaction transferring coins from A to B

g) Decode the raw transaction to extract the challenge script for freshly created UTXO of B, i.e. ScriptPubKey and also display its size in vbytes

h) Sign the transaction A → B and broadcast it on the network

i) Display the transaction ID and transaction size (in vbytes)

- Legacy_2.py

Outputs :

```
PS C:\Users\Subhas Kr Gupta\AppData\Roaming\Bitcoin> & "C:/Users/Subhas Kr Gupta/AppData/Local/Programs/Python/Python313/python.exe" "c:/Users/Subhas Kr Gupta/AppData/Roaming/Bitcoin/dup2.py"
Wallet 'Synergy_Legacy' is already loaded.

Sender Address: mfrVyD5imKR9EzuvJrDZvVk1BJxzTDuB4h
Receiver Address: mfnS29rRqeBFybrD7z7UFCvAs36Zbiufh9

Fetching the UTXO list ...

UTXO of Sender:
TXID: 82f14104bf6757a7a08d90ce6b16aad7f69e6d10bf42d4df569831388356efed
Vout: 0
Amount: 50.00000000 BTC

Enter the amount to send from sender to receiver (max 49.99990000 BTC): 0.0001

Creating the transaction from Sender to Receiver ...

Unsigned raw transaction hex:
0200000001edef568338319856dfd442bf106d9ef6d7aa166bce908da0a75767bf0441f1820000000000fdffffff0210270000
000000001976a91402ed6240e32c6a936fd683cbbf12a91b6ac753ac88ace0a3052a010000001976a91403b258246c4f7cd0f4
30f623edb129e07d76a49088ac00000000

Signing the transaction Sender → Receiver ...

Signed transaction hex:
0200000001edef568338319856dfd442bf106d9ef6d7aa166bce908da0a75767bf0441f182000000006a47304402205b30fcbe
a3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b0702200c2508d9f197a05a9889198e1019969e151c27a660
e73c56e2a8c2a899f2ea24012103c58267579b20423941bfefd838855d314247bab9b3f320a58db16d5a97499957fdffffff02
10270000000000001976a91402ed6240e32c6a936fd683cbbf12a91b6ac753ac88ace0a3052a010000001976a91403b258246c
4f7cd0f430f623edb129e07d76a49088ac00000000
```

```
Signing the transaction Sender → Receiver ...

Signed transaction hex:
0200000001edef568338319856dfd442bf106d9ef6d7aa166bce908da0a75767bf0441f182000000006a47304402205b30fcbe
a3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b0702200c2508d9f197a05a9889198e1019969e151c27a660
e73c56e2a8c2a899f2ea24012103c58267579b20423941bfefd838855d314247bab9b3f320a58db16d5a97499957fdffffff02
10270000000000001976a91402ed6240e32c6a936fd683cbbf12a91b6ac753ac88ace0a3052a010000001976a91403b258246c
4f7cd0f430f623edb129e07d76a49088ac00000000

Broadcasting the transaction Sender → Receiver ...

Transaction ID (Sender → Receiver): ccfe008a27c0dc7d614497b12649bdfda08ced24f7fa046adeeccf323575a6a2
Transaction size: 225 vbytes

Decoding raw transaction to extract the response script ...

Extracted ScriptSig:
47304402205b30fcbea3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b0702200c2508d9f197a05a9889198e
1019969e151c27a660e73c56e2a8c2a899f2ea24012103c58267579b20423941bfefd838855d314247bab9b3f320a58db16d5a
97499957
Script size: 106 vbytes

Unloaded wallet: Synergy_Legacy
PS C:\Users\Subhas Kr Gupta\AppData\Roaming\Bitcoin>
```

Explanation :

a) Load the wallet Synergy_Legacy

b) Fetch the legacy addresses B and C created by Legacy_1.py

c) Fetch and display the UTXO details of B from the transaction A → B

d) Create a new transaction B → C funded by this UTXO balance by following the same procedure as that for the transaction A → B

e) Display the transaction ID and transaction size (in vbytes)

f) Decode the transaction B → C to extract the response script to unlock the UTXO balance of B, i.e. ScriptSig and also display its size in vbytes

g) Unload the wallet at the end

Analysis of transaction

- Transaction A → B

Transaction ID:
82f14104bf6757a7a08d90ce6b16aad7f69e6d10bf42d4df569831388356efed

Transaction size: 231 vbytes
-> Transfer of 50 BTC from A to B
-> The output (UTXO) of this transaction is stored in Address B's wallet as:

1. vout : 0
2. Amount : 50 BTC
3. ScriptPubKey : 76a914bed836920f53016a64caa842778ff4f098e8255b88ac
4. Script Size : 28 vbytes

▪ Transaction B → C

Transaction ID:
ccfe008a27c0dc7d614497b12649bdfda08ced24f7fa046adeeccf323575a6a2

Transaction size: 225 vbytes
- Transfer of .001 BTC from B to C
- The input for this transaction is the UTXO from the previous transaction as:

Referred Transaction ID :
82f14104bf6757a7a8d90ce6b16aad7f69e6d10bf42d4df569831388356efed

Referred Output Index (vout) : 0

UTXO Balance unlocked : 50 BTC (.001 BTC sent to C, remaining coins back to B)

Challenge Script (ScriptPubKey) :
76a914bed836920f53016a64caa842778ff4f098e8255b88ac

Response Script (ScriptSig) :
47304402205b30fcbea3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b
0702200c2508d9f197a05a98891981019969e151c27a660e73c56e2a8c2a899f2ea2401
2103c58267579b20423941bfefd838855d314247bab9b3f320a58db16d5a97499957

Response Script Size : 106 vbytes

Structure of Legacy scripts

▪ Response Script (ScriptSig)

47304402205b30fcbea3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b
0702200c2508d9f197a05a98891981019969e151c27a660e73c56e2a8c2a899f2ea2401
2103c58267579b20423941bfefd838855d314247bab9b3f320a58db16d5a97499957

-> This script provides a cryptographic proof (signature + public key) to satisfy the conditions set by the ScriptPubKey

Length of signature : 47

ECDSA signature (proving ownership of Address B’s private key) :
304402205b30fcbea3abcd44798d0836a39d08bc51fe5acd1fe2f731976d0377b2dc5b07
02200c2508d9f197a05a98891981019969e151

Length of public key : 21

Compressed public key of Address B :
b20423941bfefd838855d314247bab9b3f320a58db16d5a97499957

``

▪ Challenge Script (ScriptPubKey)

76a914bed836920f53016a64caa842778ff4f098e8255b88ac

Duplicate the public key (OP_DUP) : 76

Hash the duplicated public key using SHA-256 + RIPEMD-160 (OP_HASH160): a9

Push 20 bytes (length of the hashed public key) : 14

20-byte hash of Address B’s public key :
bed836920f53016a64ca a842778ff4f098e8255b

Verify the computed hash matches the embedded hash (OP_EQUALVERIFY) : 88

Validate the cryptographic signature(OP_CHECKSIG) : ac

Part 2: P2SH-SegWit Address Transactions

Output :

```
SegWit Addresses:
X: 2N5XyPqGZbKZxF3Aq1s4kYqfH1MQ6FJYvZC
Y: 2MysA9wZv9hUn7qXZ9m1CQV6Y5Wp3Kx5Bf3
Z: 2NDf1Vt6qGQhK7LmPzBXyRJ2KwTzXvD7JXG

Mining some initial blocks to fund address X ...

Balance of X: 100.00000000 BTC
UTXO of X: 60.00000000 BTC

Enter the amount to send from  X → Y (max 59.99990000 BTC): 50

Creating a raw transaction from  X → Y ...

Decoding the transaction X → Y to extract challenge script ...

Extracted ScriptPubKey: a914e2366dcc691d7984c9b6d951a54deeeef63ed89387
Script size: 22 vbytes

Signing the transaction  X → Y ...

Broadcasting the transaction  X → Y ...

Transaction ID ( X → Y): 8b6f32359440fca32d97a49bc6f92586b04aafa67a7c8645593754825b2
Transaction size: 161 vbytes
```

```
Fetching the UTXO list ...

UTXO of Y:
TXID: 8b6f32359440fca32d97a49bc6f92586b04aafa67a7c8645593754825b2
Vout: 0
Amount: 50.0000000

Enter the amount to send from Y to Z (max 49.99990000 BTC): 0.0001
Creating a raw transaction from Y to Z ...

Signing the transaction Y → Z ...

Broadcasting the transaction Y → Z ...

Transaction ID (Y → Z): 2a34129c8997cbbdd2e6211ec1454f03f7c22c67a0cef83e57d47cc702af2808
Transaction size: 164 vbytes

Decoding the transaction Y → Z to extract response script

Extracted ScriptSig: 160014a405007a7d990b3ea801908a7e0256536b47b395
Script size: 20 vbytes

Unloaded wallet: Synergy_SegWit
```

For this part, we have created a single Python code SegWit.py which will:

a) Create a new wallet (or load existing wallet) named Synergy_SegWit

b) Generate 3 SegWit addresses A, B and C

c) Mine some initial blocks in order to fund address A

d) Display the UTXO balance of A once it is funded

e) Ask the user the amount to be transferred from A to B, satisfying the condition: $0 < \text{Amount} \leq \text{UTXO(A)} - \text{Mining fee}$

f) Create a raw transaction transferring coins from A to B

g) Decode the raw transaction to extract the challenge script for freshly created UTXO of B, i.e. ScriptPubKey and also display its size in vbytes

h) Sign the transaction A → B and broadcast it on the network

i) Display the transaction ID and transaction size (in vbytes)

j) Fetch and display the UTXO details of B from the transaction A → B

k) Create a new transaction B → C funded by this UTXO balance by following the same procedure as that for the transaction A → B

l) Display the transaction ID and transaction size (in vbytes)

m) Decode the transaction B → C to extract the response script to unlock the UTXO balance of B, i.e. ScriptSig and also display its size in vbytes

n) Unload the wallet at the end

Analysis of transactions

- Transaction A → B

Transaction ID:
8b6f32359440fca32d97a49bc6f92586b04aafa67a7c8645593754825b2

Transaction size: 161 vbytes
- Transfer of 50 BTC from A to B
- The output (UTXO) of this transaction is stored in Address B's wallet as:

vout : 0

Amount : 50 BTC

ScriptPubKey : a914e2366dcc691d7984c9b6d951a54deeeef63ed89387

Script Size : 22 vbytes

- Transaction B → C

Transaction ID:
2a34129c8997cbbdd2e6211ec1454f03f7c22c67a0cef83e57d47cc702af2808

Transaction size: 164 vbytes
- Transfer of .0001 BTC from B to C
- The input for this transaction is the UTXO from the previous transaction as:

Referred Transaction ID :
8b6f32359440fca32d97a49bc6f92586b04aafa67a7c8645593754825b2

Referred Output Index (vout) : 0

UTXO Balance unlocked : 50 BTC (.0001 BTC sent to C, remaining coins back to B)

Challenge Script (ScriptPubKey) :
a914e2366dcc691d7984c9b6d951a54deeeef63ed89387

Response Script (ScriptSig) :
160014a405007a7d990b3ea801908a7e0256536b47b395

Response Script Size : 20 vbytes

Structure of SegWit scripts

▪ Response Script (ScriptSig)

“160014a405007a7d990b3ea801908a7e0256536b47b395”

->This script provides a cryptographic proof (signature + public key) to satisfy the conditions set by the ScriptPubKey

length of the witness program : 16

Witness program :
0014a405007a7d990b3ea8 01908a7e0256536b47b395

▪ Challenge Script (ScriptPubKey)

“a914e2366dcc691d7984c9b6d951a54deeeef63ed89387”

->This script locks funds to a SegWit-compatible redeem script hash. The actual spending requires validation of witness data (signature + public key)

Hash the redeem script using SHA256 + RIPEMD-160 : a9

Push 20 bytes (length of the hashed redeem script) : 14

20-byte hash of the redeem script (witness program) :
e2366dcc691d7984c9b6 d951a54deeeef63ed893

Verify the computed hash matches the embedded hash : 87

Part 3: Analysis and Explanation

Size Comparison :

Size (in vbytes)	Legacy Addresses	SegWit Addresses
Transaction size	225	161
ScriptPubKey size	28	23
ScriptSig size	106	20

Evidently, SegWit addresses led to a reduction in transaction size and script size

Script Structure Comparison :

Legacy Addresses :

1. Signatures and public keys are embedded directly in the transaction's ScriptSig, bloating the transaction size

2. Both the sender and receiver’s public key hashes are stored in the transaction body

SegWit Addresses :

1. Critical validation data (signatures, public keys) is stored in a separate witness field, not counted as heavily toward transaction size
2. Only the redeem script hash is embedded in the transaction body, reducing redundancy