Test 6 - transactions

P2PKH vs P2PK [ID: 490059]

Why is P2PKH preferred over P2PK?

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	It isn't, really. P2PK is easier and better.
○	○	A key hash is easier to remember than a key.
○	○	The scripts are shorter for P2PKH.
○	○	The public key is not revealed in the UTXO.

P2PKH vs P2PK [ID: 490059]

right	wrong	
○	◉	It isn't, really. P2PK is easier and better.
○	◉	A key hash is easier to remember than a key.
◉	○	The scripts are shorter for P2PKH.
◉	○	The public key is not revealed in the UTXO.

OP_CHECKSIG [ID: 490073]

Which information (arguments and otherwise) is needed for OP_CHECKSIG to operate?

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	signature
○	○	public key
○	○	current time
○	○	transaction

OP_CHECKSIG [ID: 490073]

right	wrong	
◉	○	signature
◉	○	public key
○	◉	current time
◉	○	transaction

P2SH and Multi-Signature [ID: 490062]

Consider the following statements about P2SH and multi signature transactions.

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	Payments to a script hash can be recovered even if the script is lost.
○	○	P2SH can be used to emulate multi signature transactions.
○	○	There are fewer restrictions on redeem scripts than on locking and unlocking scripts.
○	○	One can add a new signature to an existing multi signature transaction.

P2SH and Multi-Signature [ID: 490062]

right	wrong	
○	◉	Payments to a script hash can be recovered even if the script is lost.
◉	○	P2SH can be used to emulate multi signature transactions.
◉	○	There are fewer restrictions on redeem scripts than on locking and unlocking scripts.
○	◉	One can add a new signature to an existing multi signature transaction.

OP_RETURN [ID: 490065]

Why is OP_RETURN preferred over sending bitcoin to a fictitious address?

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	More data can be stored on the blockchain using OP_RETURN than with an address.
○	○	Outputs with OP_RETURN do not fill up the pool of UTXOs of a full node.
○	○	There is no harm in using made up addresses.
○	○	OP_RETURNed outputs can be redeemed with a special script.

OP_RETURN [ID: 490065]

right	wrong	
◉	○	More data can be stored on the blockchain using OP_RETURN than with an address.
◉	○	Outputs with OP_RETURN do not fill up the pool of UTXOs of a full node.
○	◉	There is no harm in using made up addresses.
○	◉	OP_RETURNed outputs can be redeemed with a special script.

Multi Signature Transaction [ID: 490068]

- Give the locking and unlocking scripts for a multi signature transaction where N=4 and M=1. The four participants are Alice, Bob, Carl, and Dude. Suppose that Carl wants to redeem.
- Is there more than one way to write the locking and unlocking scripts?

Multi Signature Transaction [ID: 490068]

Multi Signature

- Locking Script:

  1 <PubKey_Alice> <PubKey_Bob> <PubKey _Carl> <PubKey_Dude> 4 OP_CHECKMULTISIG

- Unlocking Script

  OP_0 <Sig Carl>

Multi Signature Transaction [ID: 490068]

Multi Signature

- Locking Script:

  1 <PubKey_Alice> <PubKey_Bob> <PubKey _Carl> <PubKey_Dude> 4 OP_CHECKMULTISIG
- Unlocking Script

  OP_0 <Sig Carl>

P2SH:

- Locking Script:

  OP_HASH160 <20-byte hash of redeem script> <OP_EQUALVERIFY>
- Unlocking Script:

  Sig_Carl <redeem script>
- Redeem script:

  1 <PubKey_Alice> <PubKey_Bob> <PubKey _Carl> <PubKey_Dude> 4 OP_CHECKMULTISIG

Test 7

SPV [ID: 491640]

What does SPV stand for?

1.
2.
3.

- Simplified Payment Verification

full node vs SPV node [ID: 491645]

Consider these statements about full nodes and SPV nodes.

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	SPV nodes need not store the full blocks of the blockchain
○	○	SPV nodes store the headers of all blocks
○	○	There is no way to obtain only the block headers form a node
○	○	SPV nodes locally verify all transactions

full node vs SPV node [ID: 491645]

right	wrong	
◉	○	SPV nodes need not store the full blocks of the blockchain
◉	○	SPV nodes store the headers of all blocks
○	◉	There is no way to obtain only the block headers form a node
○	◉	SPV nodes locally verify all transactions

bloom filter - hash functions [ID: 491651]

Let $m = 15$and consider the following hash functions

h_1(x) = $(57x + 13)$ **% 15**

h_2(x) = $(12x + 3)$ **% 15**

Are these hash functions usable for a Bloom filter?

- ○ Yes, no problem
- ○ No, two hash functions are not enough
- ○ No, the hash functions are not independent

bloom filter - hash functions [ID: 491651]

- Yes, no problem
- No, the hash functions are not independent
- No, two hash functions are not enough

Using Bloom filter [ID: 491659]

Let m=15 and consider the following hash functions

h1(x) = (3x+1) % 15

h2(x) = (14x-3) % 15

h3(x) = (7x+6) % 15

Enter the numbers 1, 2, 3 into the empty Bloom filter with these parameters.

Write down the resulting bitvector using the format BITVECTOR=000000000000000 (change 0 to 1 for bits that are set, the leftmost bit has index 0).

For each of the numbers 4, 5, 6, 7, 8, 9 determine whether the filter contains them! Write your answers in the form 4:YES or 4:NO (if 4 is/is not contained) etc up to 9:YES or 9:NO.

Generally: Do not use spaces inside an answer!

The order of the answers doesn't matter.

Using Bloom filter [ID: 491659]

$h1(x) = (3x+1) \% 15$

$h2(x) = (14x-3) \% 15$

$h3(x) = (7x+6) \% 15$

- $h1(1) = 4$, $h2(1) = 11$, $h3(1) = 13$
- $h1(2) = 7$, $h2(2) = 10$, $h3(2) = 5$
- $h1(3) = 10$, $h2(3) = 9$, $h3(3) = 12$

Using Bloom filter [ID: 491659]

- h1(1) = 4, h2(1) = 11, h3(1) = 13
- h1(2) = 7, h2(2) = 10 , h3(2) = 5
- h1(3) = 10, h2(3) = 9, h3(3) = 12

BITVECTOR=00001101011111O

Using Bloom filter [ID: 491659]

- h1(4) = 13, h2(4) = 8, h3(4) = 4

  BITVECTOR=00001101**0**111110

- NO

Using Bloom filter [ID: 491659]

- h1(7) = 7, h2(7) = 5, h3(7) = 10

  BITVECTOR=0000110101111110

- YES

Using Bloom filter [ID: 491659]

Score is granted based on the occurrence of the following keywords:

- BITVECTOR=00001101011110 for 8 Points
- 4:NO for 2 Points
- 5:NO for 2 Points
- 6:NO for 2 Points
- 7:YES for 2 Points
- 8:NO for 2 Points
- 9:NO for 2 Points

answers of a Bloom filter [ID: 491687]

The check operation of a Bloom filter for set M applied to x returns True or False, but what does that mean?

Element x is definitely in M.	Check returns True
Element x may be in M.	Check returns False
Element x may not be in M.	Not applicable
Element x is definitely not in M.	

answers of a Bloom filter [ID: 491687]

Element x may be in M.	matches	Check returns True
Element x is definitely not in M.	matches	Check returns False
Element x may not be in M.	matches	Not applicable
Element x is definitely in M.	matches	Not applicable

Deterministic Wallet [ID: 491690]

Which of the following statements about deterministic wallets are correct?

You have to decide on every statement: [right] or [wrong]

right	wrong	
○	○	All keys need to be pregenerated.
○	○	One number is sufficient to remember all keys in the wallet.
○	○	Matching secret keys and public keys can be generated independently from an index.
○	○	GIven a public key from the wallet, one can calculate the next public key.

Deterministic Wallet [ID: 491690]

right	wrong	
○	●	All keys need to be pregenerated.
●	○	One number is sufficient to remember all keys in the wallet.
●	○	Matching secret keys and public keys can be generated independently from an index.
○	●	GIven a public key from the wallet, one can calculate the next public key.

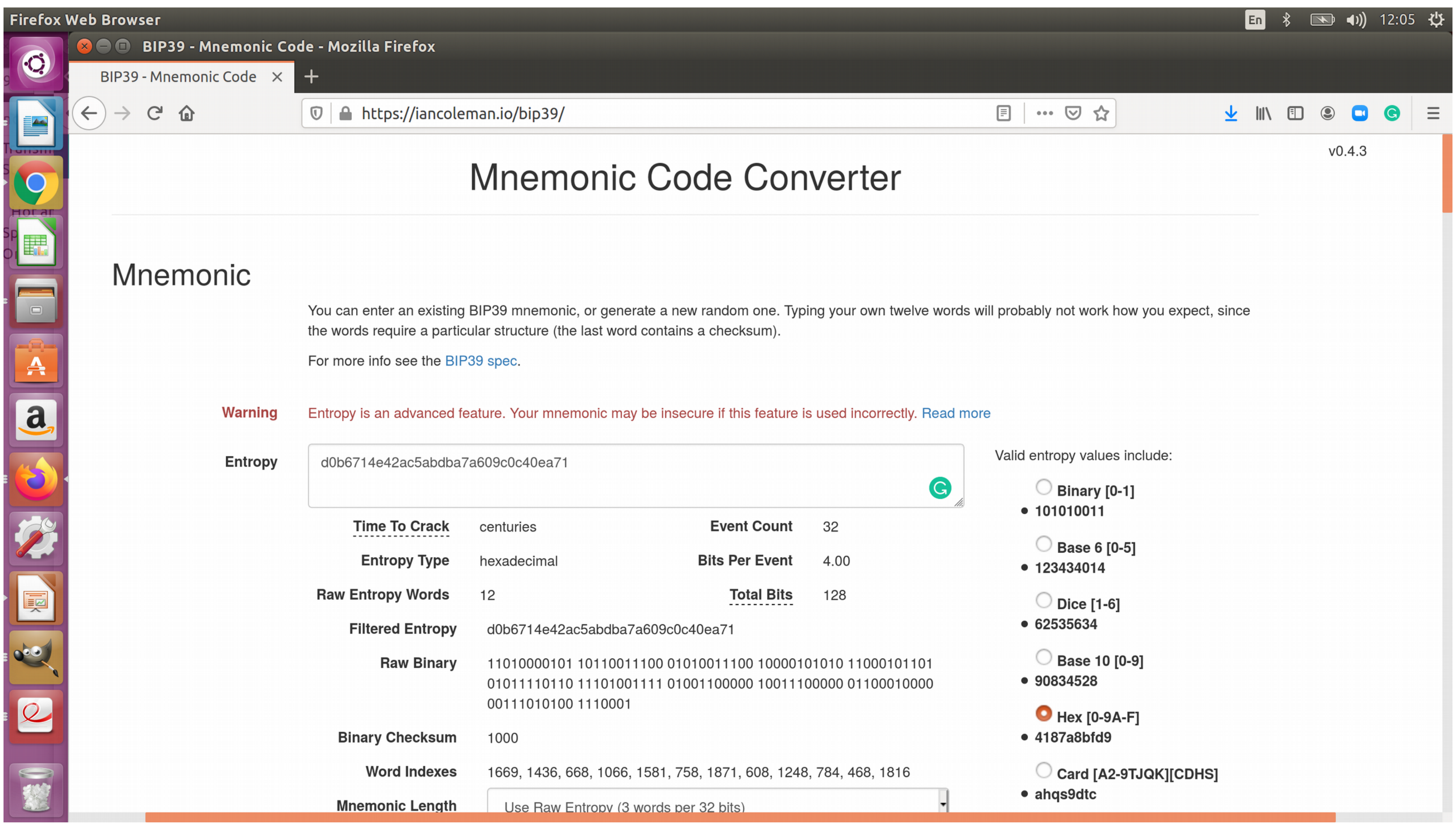
Mnemonic code words [ID: 491693]

- Give the number (128 bits) represented by this list of mnemonic code words (BIP 0039):

  spawn receive fatal luxury shield gain try equal orchard ginger dentist toast

Mnemonic code words [ID: 491693]

- spawn receive fatal luxury shield gain try equal orchard ginger dentist toast



Mnemonic code words [ID: 491693]

- spawn receive fatal luxury shield gain try equal orchard ginger dentist toast

bip-0039 English wordlist

[redacted]	0028 address	0055 almost	0082 any	0109 assault	0136 axis	0163 begir
0002 ability	0029 adjust	0056 alone	0083 apart	0110 asset	[redacted]	0164 beha
0003 able	0030 admit	0057 alpha	0084 apology	0111 assist	0138 bachelor	0165 behir
0004 about	0031 adult	0058 already	0085 appear	0112 assume	0139 bacon	0166 belie
0005 above	0032 advance	0059 also	0086 apple	0113 asthma	0140 badge	0167 belov
0006 absent	0033 advice	0060 alter	0087 approve	0114 athlete	0141 bag	0168 belt
0007 absorb	0034 aerobic	0061 always	0088 april	0115 atom	0142 balance	0169 bencl
0008 abstract	0035 affair	0062 amateur	0089 arch	0116 attack	0143 balcony	0170 bene
0009 absurd	0036 afford	0063 amazing	0090 arctic	0117 attend	0144 ball	0171 best
0010 abuse	0037 afraid	0064 among	0091 area	0118 attitude	0145 bamboo	0172 betra
0011 access	0038 again	0065 amount	0092 arena	0119 attract	0146 banana	0173 bette
0012 accident	0039 age	0066 amused	0093 argue	0120 auction	0147 banner	0174 betw
0013 account	0040 agent	0067 analyst	0094 arm	0121 audit	0148 bar	0175 beyo
0014 accuse	0041 agree	0068 anchor	0095 armed	0122 august	0149 barely	0176 bicyc
0015 achieve	0042 ahead	0069 ancient	0096 armor	0123 aunt	0150 bargain	0177 bid
0016 acid	0043 aim	0070 anger	0097 army	0124 author	0151 barrel	0178 bike
0017 acoustic	0044 air	0071 angle	0098 around	0125 auto	0152 base	0179 bind
0018 acquire	0045 airport	0072 angry	0099 arrange	0126 autumn	0153 basic	0180 biolo
0019 across	0046 aisle	0073 animal	0100 arrest	0127 average	0154 basket	0181 bird
0020 act	0047 alarm	0074 ankle	0101 arrive	0128 avocado	0155 battle	0182 birth
0021 action	0048 album	0075 announce	0102 arrow	0129 avoid	0156 beach	0183 bitter
0022 actor	0049 alcohol	0076 annual	0103 art	0130 awake	0157 bean	0184 black
0023 actress	0050 alert	0077 another	0104 artefact	0131 aware	0158 beauty	0185 blade
0024 actual	0051 alien	0078 answer	0105 artist	0132 away	0159 because	0186 blam
0025 adapt	0052 all	0079 antenna	0106 artwork	0133 awesome	0160 become	0187 blank
0026 add	0053 alley	0080 antique	0107 ask	0134 awful	0161 beef	0188 blast
0027 addict	0054 allow	0081 anxiety	0108 aspect	0135 awkward	0162 before	0189 bleak

Mnemonic code words [ID: 491693]

- Give the number (128 bits) represented by this list of mnemonic code words (BIP 0039):

  spawn receive fatal luxury shield gain try equal orchard ginger dentist toast

- D0b6714e42ac5abdba7a609c0c40ea71
- 11010000101 10110011100 01010011100 10000101010 11000101101 01011110110 11101001111 01001100000 10011100000 01100010000 00111010100 1110001

Key reconstruction from two fragments [ID: 491696]

Max has distributed fragments of his secret key using the linear interpolation scheme, where the key can be reconstructed from any two fragments. His calculations are done modulo 2971. Alice found the following two fragments

- (3, 900)
- (1771, 1000)

Help Alice to calculate the secret and the random number used to hide the secret.

Write your answers in the format SECRET=1111 and RANDOM=1111 where you replace 1111 by the respective answer. Do not use spaces. Use leading zeroes if necessary.

Key reconstruction from two fragments [ID: 491696]

- b – a = (s + j * m mod p) – (s + i * m mod p)
- b – a = (j – i) m mod p
- n * p + (b - a) = (j – i) * m

- n * 2971 + (1000 – 900) = 1771 * m
- m = (n * 2971 + 100) / 1771

e.g

```
def findm (p, a, b, i, j):
    n = 1
    x = b - a
    y = j - i
    while n < p:
        if (p*n + x) % y == 0:
        return ((p*n + x) / y)
        n = n + 1
m = findm(2971, 900, 1000, 3, 1771)
```

Key reconstruction from two fragments [ID: 491696]

- $a = s + i * m \bmod p$ and $b = s + j * m \bmod p$
- $n_1 * p + a = s + i * m$ and $b = s + j * m \bmod p$
- $s = n_1 * p + a – i * m$ and $s = n_2 *p + b – j * m$
- $s = n_1 * p + a – i * m$ and $(a – b) + n_1 * p + m (j – i) \% p == 0$

```
def finds (m, p, a, b, i, j):
  n = 1
  x = b - a
  y = j - i
  while n < p:
    if (-x + p*n + m*y ) % p == 0 and (p*n + a - i
    * m) > 0:
    return (p*n + a - i * m)
    n = n + 1
s = finds(1499, 2971, 900, 1000, 3, 1771)
print s
```

Key reconstruction from two fragments [ID: 491696]

Score is granted based on the occurrence of the following keywords:

- SECRET=2345 for 10 Points
- RANDOM=1499 for 10 Points