## Module sui::bls12381

Group operations of BLS12-381.

@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public\_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg: The message that we test the signature against.

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G1\_XMD:SHA-256\_SSWU\_RO\_NUL\_, return true. Otherwise, return false.

@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public\_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_, return true. Otherwise, return false.

Returns e2/e1, fails if a is zero.

Returns e2 / e1, fails if scalar is zero.

Hash using DST = BLS\_SIG\_BLS12381G1\_XMD:SHA-256\_SSWU\_RO\_NUL\_

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).

Convert an Element  $< \underline{G1} >$  to uncompressed form.

Returns e2 / e1, fails if scalar is zero.

Hash using DST = BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).

Returns e2 / e1, fails if scalar is zero.

UncompressedG1 group operations /// Create a Element < G1 > from its uncompressed form

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.

## Struct

basn
***
```bash
***

```bash
```bash
@param signature: A 48-bytes signature that is a point on the $G1$ subgroup. @param public_key: A 96-bytes public key that is a point on the $G2$ subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_, return true. Otherwise, return false.

```bash	
```bash	
···	
```bash	
Returns e2/e1, fails if a is zero.	

***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
```bash			
```bash			
```bash			
```bash			

```
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
Convert an Element < \frac{G1}{} > to uncompressed form.
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
,,,
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```

```bash	
```bash	
Returns e2 / e1, fails if scalar is zero.	
```bash	
```bash	

```
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Struct
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```

```bash
```bash
@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
```bash
uasii
···
```bash
"bash
"bash" "bash

```bash	
```bash	
Returns e2/e1, fails if a is zero.	
```bash	
```bash	

```
```bash
***
```bash
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```

```
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element < \underline{G1} > to uncompressed form.
```bash
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
```

## Struct

```bash		
w		
```bash		
***		
```bash		
***		
```bash		
···		
```bash		
···		
```bash		
···		
```bash		
***		
```bash		
***		
```bash		
***		
```bash		
```bash		
```bash		
Udsii		
```bash		
···		
```bash		
***		
```bash		
***		
```bash		
w		

@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public\_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg. The message that we test the signature against.

256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
@param signature: A 96-bytes signature that is a point on the $G2$ subgroup. @param public_key: A 48-bytes public key that is a point on the $G1$ subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
```bash
```bash
···
```bash
```bash
···
```bash
```hash

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G1\_XMD:SHA-

```bash	
***	
```bash	
```bash	
Returns e2/e1, fails if a is zero.	
```bash	
···	
```bash	
```bash	
****	
```bash	
```bash	
···	
```bash	
···	
```bash	
```bash	
···	
```bash	
···	
```bash	

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element < \underline{G1} > to uncompressed form.
```bash
```bash
```

```bash	
```bash	
```bash	
***	
```bash	
····	
```bash	
····	
```bash	
···	
```bash	
***	
```bash	
***	
```bash	
····	
Returns e2 / e1, fails if scalar is zero.	
```bash	
····	
```bash	
····	
```bash	
****	
```bash	

```
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```

Hash using DST = BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_

```
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Struct
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```bash
```

```bash
```bash
@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
```bash
***
```bash

***	
```bash	
```bash	
```bash	
```bash	
****	
```bash	
Returns e2/e1, fails if a is zero.	
```bash	
****	
```bash	
```bash	
```hash	

```bash	
***	
```bash	
···	
```bash	
```bash	
```bash	
WY	
```bash	
MI.	
```bash	
***	
```bash	
W.	
```bash	
***	
```bash	
***	
Returns e2 / e1, fails if scalar is zero.	
```bash	
···	
```bash	
```bash	

```
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
,,,
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element< G1 > from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.

```bash			
***			
```bash			
***			
Struct			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			
***			
```bash			

point on the G2 subgroup. @param msg. The message that we test the signature against.

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G1\_XMD:SHA-256 SSWU RO NUL, return true. Otherwise, return false. ```bash \*\*\* ```bash @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public\_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against. If the signature is a valid signature of the message and public key according to BLS SIG BLS12381G2 XMD:SHA-256\_SSWU\_RO\_NUL\_, return true. Otherwise, return false. ```bash ```bash \*\*\* ```bash ```bash \*\*\* ```bash ```bash ```bash \*\*\* ```bash ```bash ```bash ... ```bash ```bash

```bash	
***	
```bash	
····	
```bash	
····	
```bash	
····	
Returns e2/e1, fails if a is zero.	
```bash	
····	
```bash	
****	
```bash	
····	
```bash	
****	
```bash	
***	
```bash	
w	

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
Convert an Element < \frac{G1}{} > to uncompressed form.
```bash
***
```

```bash		
***		
```bash		
***		
```bash		
***		
```bash		
***		
```bash		
····		
```bash		
***		
```bash		
····		
```bash		
***		
```bash		
***		
```bash		
```bash		
```bash		
```bash		
Returns e2 / e1, fails if scalar is zero.		
"bash		
ousii · · · ·		
```bash		
···		
```bash		
***		

```
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
```bash
```bash
,,,
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```

```
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Constants
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```

```bash
```bash
@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg: The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
```bash
"bash ""  @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash
""bash ""  @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash ""
""bash ""  @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash ""bash
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash ""bash ""
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param nsg. The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash "" "bash "" "bash
""bash "" @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg: The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  ""bash ""bash ""bash ""
""bash ""  @param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg. The message that we test the signature against.  If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_, return true. Otherwise, return false.  "bash ""bash ""bash ""bash ""bash

```bash	
```bash	
Returns e2/e1, fails if a is zero.	
```bash	
```bash	

```bash	
```bash	
***	
```bash	
```bash	
***	
```bash	
****	
```bash	
****	
```bash	
····	
```bash	
****	
```bash	
***	
```bash	
***	
```bash	
***	
```bash	
····	
Returns e2 / e1, fails if scalar is zero.	
```bash	
····	
```bash	
****	
```bash	
····	
```bash	

```
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
```bash
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
```bash
```bash
***
```bash
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
```

...

## **Function**

@param signature: A 48-bytes signature that is a point on the G1 subgroup. @param public\_key: A 96-bytes public key that is a point on the G2 subgroup. @param msg: The message that we test the signature against.

If the signature is a valid signature of the message and public key according to BLS\_SIG\_BLS12381G1\_XMD:SHA-256\_SSWU\_RO\_NUL\_, return true. Otherwise, return false.

```
"bash
"bash
""
```

@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public\_key: A 48-bytes public key that is a point on the G1 subgroup. @param msg. The message that we test the signature against.

If the signature is a valid signature of the message and public key according to  $BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL$ , return true. Otherwise, return false.

```
256_SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
***
```bash
```bash
```bash
```bash
```bash
```bash
```bash
...
```bash
```bash
```bash
```

Wheels	
```bash	
Returns e2/e1, fails if a is zero.	
```bash	
```bash	

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
...
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```

## Convert an Element $< \underline{G1} >$ to uncompressed form. ```bash \*\*\* ```bash ```bash \*\*\* ```bash ```bash \*\*\* ```bash \*\*\* ```bash \*\*\* ```bash \*\*\* Returns e2 / e1, fails if scalar is zero. ```bash \*\*\* ```bash

```
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```

```
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
```bash
...
```bash
UncompressedG1 group operations /// Create a Element< G1 > from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
@param signature: A 96-bytes signature that is a point on the G2 subgroup. @param public key: A 48-bytes public key that is a
point on the G1 subgroup. @param msg. The message that we test the signature against.
If the signature is a valid signature of the message and public key according to BLS_SIG_BLS12381G2_XMD:SHA-
256 SSWU_RO_NUL_, return true. Otherwise, return false.
```bash
```bash
,,,
```bash
```

```
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```

```
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq \underline{G1} \geq to uncompressed form.
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.

```bash \*\*\* ```bash \*\*\*

\*\*\*

\*\*\*

**Function** ```bash \*\*\* ```bash \*\*\* ```bash ```bash \*\*\*

```bash ```bash

```bash \*\*\*

```bash \*\*\* ```bash

```bash \*\*\* ```bash

\*\*\* ```bash \*\*\*

```bash

| Returns e2/e1, fails if a is zero. |  |
|------------------------------------|--|
| ```bash                            |  |
| WY                                 |  |
| ```bash                            |  |
| ····                               |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |
| ···                                |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |
| WY                                 |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |
|                                    |  |
| ```bash                            |  |

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element < \frac{G1}{} > to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
```

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts

with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
···
```bash
···
```bash
****
```bash
···
```bash
···
```bash
···
```bash
```bash
```bash
···
```bash
···
```bash
···
Returns e2 / e1, fails if scalar is zero.
```bash
****
```bash
```bash
···
```bash

```
```bash
***
```bash
\label{lem:compressed} Uncompressed G1 \ group \ operations \ {\it ///} \ Create \ a \ Element < \underline{G1} > from \ its \ uncompressed \ form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```bash
```

```bash	
···	
```bash	
····	
Returns e2/e1, fails if a is zero.	
```bash	
···	
```bash	
····	
```bash	
···	
```bash	
```bash	
····	
```bash	
```bash	

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```

```
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
,,,
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```bash
```

```bash	
···	
```bash	
····	
Returns e2/e1, fails if a is zero.	
```bash	
···	
```bash	
····	
```bash	
···	
```bash	
```bash	
···	
```bash	
```bash	

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```

```
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
,,,
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```bash
```

Returns e2/e1, fails if a is zero.	
```bash	
WY	
```bash	
····	
```bash	
```bash	
```bash	
···	
```bash	
```bash	
WY	
```bash	
```bash	
```bash	
```bash	

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element < \underline{G1} > to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
```

with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
···
```bash
···
```bash
****
```bash
···
```bash
···
```bash
···
```bash
```bash
```bash
···
```bash
···
```bash
···
Returns e2 / e1, fails if scalar is zero.
```bash
****
```bash
```bash
···
```bash

```
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
```bash
***
```bash
```bash
```bash
***
```bash
Returns e2/e1, fails if a is zero.
```bash
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```

```bash

```
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq \underline{G1} \geq to uncompressed form.
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
```bash
Returns e2/e1, fails if a is zero.
```bash
```bash

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
```

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).

```
```bash
***
```bash
***
Convert an Element < \underline{G1} > to uncompressed form.
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```

Returns e2 / e1, fails if scalar is zero.

```bash
```bash
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector $[s1, s2,, sn]$ and 'elements' be the vector $[e1, e2,, en]$ . Returns $s1$ $e1 + s2$ $e2 + + sn*en$ . Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash

```
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
```bash
Returns e2/e1, fails if a is zero.
```

```bash		
···		
```bash		
***		
```bash		
****		
```bash		
W.		
```bash		
W.		
```bash		
W.		
```bash		
···		
```bash		
WY .		
```bash		
···		
```bash		
***		
```bash		
```bash		
```bash		
```bash		
```bash		
```bash		
Dasii		
```bash		

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```

```bash

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```

```
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
Returns e2/e1, fails if a is zero.
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
,,,
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```

```
```bash
***
Convert an Element < \underline{G1} > to uncompressed form.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```

```
```bash
***
```bash
```bash
,,,
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
,,,
```bash
```bash
***
```bash
```bash
```

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
***
```bash
***
```bash
```bash
```

```bash	
```bash	
***	
```bash	
····	
```bash	
····	
```bash	
***	
```bash	
***	
```bash	
***	
```bash	
····	
Returns e2 / e1, fails if scalar is zero.	
```bash	
····	
```bash	
···	
```bash	
****	
```bash	

```
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
,,,
```bash
```bash
***
```bash
```bash
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
,,,
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
```bash
```bash
***
```bash
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
```

٠,,

## **Function**

```bash		
w		
```bash		
···		
```bash		
***		
```bash		
***		
```bash		
***		
```bash		
w		
```bash		
w		
```bash		
w		
```bash		
w		
```bash		
w		
```bash		
w		
```bash		
***		
```bash		
***		
```bash		
***		
Returns e2 / e1, fails if scalar is zero.		
```bash		
***		
```bash		

```
```bash
***
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq \underline{G1} \geq to uncompressed form.
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements. ```bash \*\*\* ```bash \*\*\*

## **Function**

```
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```

Returns e2 / e1, fails if scalar is zero.

```bash

```
```bash
***
```bash
```bash
,,,
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
```bash
,,,
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
```

```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash

```
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
Convert an Element \leq \underline{G1} \geq to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
,,,
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
```bash
```bash
***
```bash
,,,
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element< G1 > from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.

```
```bash
   ***
   ```bash
 ***
Function
 ```bash
 ```bash
 ***
 ```bash
 ***
 ```bash
 ```bash
 ***
 ```bash
 ```bash
 ***
   ```bash
Returns e2 / e1, fails if scalar is zero.
 ```bash
 ***
 ```bash
 ***
 ```bash
 ***
   ```bash
Hash\ using\ DST = BLS\_SIG\_BLS12381G1\_XMD:SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL\_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU\_RO\_NUL_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-256\_SSWU_SHA-
 ```bash
 ```bash
```

٠.,

\*\*\*

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).

```
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
,,,
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
```

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element < \underline{G1} > to uncompressed form.
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
```

```
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
```bash
***
```bash
***
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
\label{lem:compressed} \mbox{Uncompressed G1 group operations} \mbox{\ensuremath{/\!/}{//} Create a Element} < \mbox{\ensuremath{\underline{G1}} > from its uncompressed form.}
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```

```
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
,,,
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.	
```bash	
```bash	
Function	
```bash	
```bash	
Returns e2 / e1, fails if scalar is zero.	
```bash	
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_	
```bash	
```bash	
Let 'scalars' be the vector $[s1, s2,, sn]$ and 'elements' be the vector $[e1, e2,, en]$ . Returns $s1 e1 + s2 e2 + + sn*en$ . Abort with ElnputTooLong if the vectors are larger than 32 (may increase in the future).	ts
```bash	
```bash	
Convert an Element $< \underline{G1} >$ to uncompressed form.	
```bash	
```hash	

```bash	
```bash	
***	
```bash	
····	
```bash	
···	
```bash	
****	
```bash	
····	
```bash	
***	
```bash	
***	
```bash	
***	
Returns e2 / e1, fails if scalar is zero.	
```bash	
****	
```bash	
····	
```bash	
····	
```bash	

\*\*\*

\*\*\*

```
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
```bash
```bash
***
```bash
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```

Hash using DST = BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_

```
```bash
***
```bash
```bash
***
```bash
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
```

٠.,

\*\*\*

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).

```
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
```

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
```

\*\*\*

```
```bash
***
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
Convert an Element \leq G1 \geq to uncompressed form.
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
```bash
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
,,,
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```

```bash

```
```bash
,,,
Function
Hash using DST = BLS_SIG_BLS12381G1_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
Convert an Element \leq G1 \geq to uncompressed form.
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
• • • •
```

```
```bash
***
```bash
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
,,,
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
Let 'scalars' be the vector $[s1, s2,, sn]$ and 'elements' be the vector $[e1, e2,, en]$ . Returns $s1 e1 + s2 e2 + + sn*en$ . Aborts with ElnputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
Convert an Element $< \underline{G1} >$ to uncompressed form.
```bash
```bash

```
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 \ e1 + s2 \ e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```

```
```bash
```

## **Function**

Convert an Element  $\leq G1 \geq$  to uncompressed form.

```
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
```

Returns e2 / e1, fails if scalar is zero.

```bash

```
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 \ e1 + s2 \ e2 + ... + sn*en. Aborts with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
,,,
```bash
***
```bash
```bash
```

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
```

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts

with EInputTooLong if the vectors are larger than 32 (may increase in the future).

····	
```bash	
Returns e2 / e1, fails if scalar is zero.	
```bash	
```bash	

```bash

```
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```

```
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash
***
```bash
***
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
```

```
```bash
***
```bash
```bash
***
```bash
```bash
,,,
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
***
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
***
```

```
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
```

```
```bash
***
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
```bash
```bash
***
```bash
```bash
***
```bash
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
```

with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
```bash
···
```bash
```bash
```bash
···
```bash
****
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash

```bash

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts

```
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
***
Function
```bash
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
```

 $Hash\ using\ DST = BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_$ 

```
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
```bash
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```

```
```bash
***
```bash
```bash
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
***
Function
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
```

Hash using DST = BLS\_SIG\_BLS12381G2\_XMD:SHA-256\_SSWU\_RO\_NUL\_

```
```bash
***
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
```bash
```bash
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```

```
```bash
***
```bash
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
***
Function
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
***
```bash
```

with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
```bash
···
```bash
```bash
```bash
···
```bash
****
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```bash

```bash

Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn\*en. Aborts

```
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
***
Function
```bash
```bash
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
```

```
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
Hash using DST = BLS_SIG_BLS12381G2_XMD:SHA-256_SSWU_RO_NUL_
```bash
```bash
***
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
```bash
***
```bash
```bash
• • • •
```

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
Let 'scalars' be the vector [s1, s2, ..., sn] and 'elements' be the vector [e1, e2, ..., en]. Returns s1 e1 + s2 e2 + ... + sn*en. Aborts
with EInputTooLong if the vectors are larger than 32 (may increase in the future).
```bash
```bash
***
```

```
```bash
***
```bash
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```bash
***
```

UncompressedG1 group operations /// Create a Element  $\leq$  G1  $\geq$  from its uncompressed form.

```
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
***
```bash
***
```bash
***
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```

```
```bash
***
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
***
Function
```bash
***
```bash
***
```bash
***
```bash
***
```bash
```bash
```bash
```

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
```bash
```bash
```

```
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
***
```bash
```bash
```bash
***
```bash
***
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
```bash
***
```bash
***
```

```
```bash
***
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
```bash
***
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
```

```
```bash
***
```bash
```bash
```bash
***
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
***
Function
Returns e2 / e1, fails if scalar is zero.
```bash
```bash
***
```bash
```bash
***
```bash
```bash
```

 $\label{lem:compressedG1} \mbox{ Uncompressed G1 group operations /// Create a Element $<$ $\underline{G1}$ > from its uncompressed form.}$ 

```
```bash
***
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
***
```bash
Function
```bash
***
```bash
***
```bash
```bash
UncompressedG1 group operations /// Create a Element \leq G1 \geq from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
```bash
```bash
•••
```

UncompressedG1 group operations /// Create a Element $\leq G1 >$ from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
UncompressedG1 group operations /// Create a Element $\leq$ G1 $\geq$ from its uncompressed form.
```bash
```bash
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash
Function
Compute the sum of a list of uncompressed elements. This is significantly faster and cheaper than summing the elements.
```bash
```bash