# Laboratory work 11

Author: Dávid Kromka

**Algorithm Description - Task 1 and 2: Hash Functions**

Function **h(n, m)**:

1. **Inputs:**
   * **n**: Integer input.
   * **m**: Integer modulus.
2. **Algorithm:**
   * Calculate the hash function **h(n) = n mod m**.

Function **g(n1, n2, n3, n4, n5, n6, m)**:

1. **Inputs:**
   * **n1, n2, n3, n4, n5, n6**: Integer inputs.
   * **m**: Integer modulus.
2. **Algorithm:**
   * Calculate the hash function **h(n)** for each **n** separately.
   * Sum the results of **h(n1) + h(n2) + h(n3) + h(n4) + h(n5) + h(n6)**.
   * Calculate the hash function **g(n\_1, n\_2, n\_3, n\_4, n\_5, n\_6) = sum mod m**.

Function **string\_to\_hash(string, m)**:

1. **Inputs:**
   * **string**: Input string of arbitrary length.
   * **m**: Integer modulus.
2. **Algorithm:**
   * Initialize **hash\_value** to 0.
   * Iterate through each character in the input string.
   * For each character, update **hash\_value** using the formula **hash\_value = (hash\_value \* 10) + ord(char)**.
   * Calculate the hash code using **hash\_value mod m**.
   * Return the hash code.

Obrázok, na ktorom je snímka obrazovky, text, dizajn

Automaticky generovaný popis

Obrázok, na ktorom je snímka obrazovky, štvorec, dizajn

Automaticky generovaný popis

**Algorithm Description - Task 3: Custom Hash Function**

Function **my\_hash\_function(string, m)**:

1. **Inputs:**
   * **string**: Input string of arbitrary length.
   * **m**: Integer modulus.
2. **Algorithm:**
   * Initialize **hash\_value** to 0.
   * Iterate through each character in the input string.
   * For each character at index **i**, update **hash\_value** using the formula **hash\_value = (hash\_value + (ord(char) \* pow(2, i))) % m**.
   * Return the final hash value.

Obrázok, na ktorom je text, písmo, snímka obrazovky

Automaticky generovaný popis

**Algorithm Description - Task 4: Schnorr's Digital Signature**

Function **generate\_key\_pair()**:

1. **Algorithm:**
   * Generate a private key and corresponding public key using the SECP256R1 elliptic curve.

Function **sign\_message(private\_key, message)**:

1. **Inputs:**
   * **private\_key**: Private key for signing.
   * **message**: Message to be signed.
2. **Algorithm:**
   * Sign the message using the private key and the ECDSA algorithm with the SHA256 hash function.
   * Return the signature.

Function **verify\_signature(public\_key, message, signature)**:

1. **Inputs:**
   * **public\_key**: Public key for signature verification.
   * **message**: Original message.
   * **signature**: Signature to be verified.
2. **Algorithm:**
   * Verify the signature using the public key, original message, and the ECDSA algorithm with the SHA256 hash function.
   * Return **True** if the verification is successful, otherwise catch the exception and return **False**.

