***SAVEETHA SCHOOL OF ENGINEERING***

***SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCE***

**EXP NO 12: To implement the signature scheme named digital signature standard**

**(Euclidean algorithm)**

**AIM**

To implement the signature scheme named digital signature standard (Euclidean algorithm)

**PROCEDURE**

* Let “A” and “B” be the fictional actors
* “A” is the sender and calculates the hash of the message and The other side “B” hashes the message and then decrypts.
* Finally verifying the signature using public key.
* Verify\_Digital\_Signature() method is used to check

**PROGRAM**

#include <stdio.h>

#include <stdint.h>

// Function to compute the modular inverse using the extended Euclidean algorithm

uint64\_t mod\_inverse(uint64\_t a, uint64\_t m) {

uint64\_t m0 = m, t, q;

uint64\_t x0 = 0, x1 = 1;

while (a > 1) {

q = a / m;

t = m;

m = a % m;

a = t;

t = x0;

x0 = x1 - q \* x0;

x1 = t;

}

if (x1 < 0)

x1 += m0;

return x1;

}

int main() {

// Parameters

uint64\_t p = 47, q = 23, alpha = 5, x = 6, y = 8, message = 10, k = 15;

// Step 1: Signature Generation

uint64\_t r = ((uint64\_t)alpha \* k) % q;

uint64\_t s = ((mod\_inverse(k, q)) \* (message + x \* r)) % q;

// Step 2: Signature Verification

uint64\_t w = mod\_inverse(s, q);

uint64\_t u1 = (message \* w) % q;

uint64\_t u2 = (r \* w) % q;

uint64\_t v = ((uint64\_t)alpha \* u1 \* (uint64\_t)y \* u2) % q;

if (v == r) {

printf("Signature is valid\n");

} else {

printf("Signature is invalid\n");

}

return 0;

}

**OUTPUT**

