Assignment 4 Q5

watIAM: sh2yap
student ID: 21111395

(a) 15, 18, 3 (Refer to appendix for full codes)

```
i:  0 t:  15
i:  1 t:  18
i:  2 t:  3
```

(b)

Q5b)

$f(s_{i-1}) = (s_i, t_i)$ $\quad E: y^2 = x^3 + Ax + B$

$r_i = x(s_{i-1} P)$ $\quad s_i = x(r_i P)$ $\quad t_i = x(r_i Q)$

$p \equiv 3 \bmod 4$

Given p, A, B, Q, P & c s.t. $P = cQ$

Note:

$t_2 = x(r_2 Q)$ $\quad t_1 = x(r_1 Q)$

$r_2 = x(s_1 P)$

Consider $p^* = (x_{p^*}, y_{p^*})$

& $x_{p^*} = t_1$

since $s_1 = x(r_1 P)$ and $P = cQ$,

$\therefore s_1 = x(r_1 cQ) = x(cr_1 Q)$ (Since commutative)

so $cr_1 Q$ requires point addition of $c \cdot (r_1 Q)$

We can now utilise publicly known t_1 and calculate the corresponding y-coordinate, y_{p^*}, by solving E.

Note that y_{p^*} corresponds to the y-coordinate of $r_1 Q$.

$\therefore$ we have the full coordinates of $r_1 Q$,

get s_1, r_2 and then t_2 by point addition accordingly, and we're done.

(c)

(d) s1 = 102, t2 =37 https://andrea.corbellini.name/ecc/interactive/modk-add.html

(e) Use well-established elliptic curves.

Appendix:

```
class EllipticCurve:
    def __init__(self, a, b, p):
```

Assignment 4 Q5

```
        self.a = a
        self.b = b
        self.p = p
    def is_point_on_curve(self, point):
        x, y = point
        return (y**2) % self.p == (x**3 + self.a * x + self.b) % self.p

    def point_addition(self, p, q):
        if p == (float('inf'), float('inf')):
            return q
        if q == (float('inf'), float('inf')):
            return p

        x_p, y_p = p
        x_q, y_q = q
        if p != q:
            # Calculate the slope
            s = ((y_q - y_p) * pow(x_q - x_p, -1, self.p)) % self.p
        else:
            # Point doubling
            s = ((3 * x_p**2 + self.a) * pow(2 * y_p, -1, self.p)) % self.p
        # Calculate the new x-coordinate
        x_r = (s**2 - x_p - x_q) % self.p
        # Calculate the new y-coordinate
        y_r = (s * (x_p - x_r) - y_p) % self.p
        return (x_r, y_r)

    def scalar_multiply(self, k, point):
        result = (float('inf'), float('inf'))
        for _ in range(k.bit_length()):
            if k & 1:
                result = self.point_addition(result, point)
            point = self.point_addition(point, point)
            k >>= 1
        return result


# Example usage
a = 2
b = 3
p = 19  # Prime modulus

# Create an elliptic curve instance
curve = EllipticCurve(a, b, p)

# Define points P and Q on the curve
P = (1, 14)
Q = (3, 13)

# Check if points are on the curve
print(f"Is P on the curve? {curve.is_point_on_curve(P)}")
```

Assignment 4 Q5

```
print(f"Is Q on the curve? {curve.is_point_on_curve(Q)}")

# Initializations
s = 2

# Do iteration
for i in range(3):
    r = curve.scalar_multiply(s, P)[0]
    s = curve.scalar_multiply(r, P)[0]
    t = curve.scalar_multiply(r, Q)[0]

    print("i: ", i, "t: ", t)
```