COMP3121
Assignment 2
A17S1N2

Evangelos Kohilas
z5114986

By submitting this document you are confirming that all the answers are your work and are not take from any other sources unless clearly mentioned.

Question 1

a)

$$\text{let } A^n = \begin{pmatrix} F(n+1) & F(n) \\ F(n) & F(n-1) \end{pmatrix} \tag{1}$$

When $n = 1$

$$A^1 = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \tag{2}$$

Assume $n = k$

$$A^k = \begin{pmatrix} F(k+1) & F(k) \\ F(k) & F(k-1) \end{pmatrix} \tag{3}$$

let $n = k + 1$

$$A^{k+1} = AA^k \tag{4}$$

$$= \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} F(k+1) & F(k) \\ F(k) & F(k-1) \end{pmatrix} \tag{5}$$

$$= \begin{pmatrix} F(k+1) + F(k) & F(k) + F(k-1) \\ F(k+1) & F(k) \end{pmatrix} \tag{6}$$

$$= \begin{pmatrix} F(k+2) & F(k+1) \\ F(k+1) & F(k) \end{pmatrix} \tag{7}$$

Therefore the formula is true by induction for all $n > 0$.

b) $F(n)$ can be found in $\log n$ matrix multiplications using a recursive algorithm

```
matrix = ((1, 1), (1,0))
func(n):
    if n == 1:
        return matrix
    if n is even:
        return func(n/2)^2
    if n is odd:
        return func(n-1) * matrix
```

Question 2

a) First we calculate the Karastuba trick.

$$(a+b)(c+d) = ac + ad + bd + bc \tag{1}$$
$$ad + bd = (a+b)(c+d) - ac - bc \tag{2}$$

Then we substitute (2) into (4)

$$(a+ib)(c+id) = ac + adi + bdi + bc \tag{3}$$
$$= ac + i(ad+bd) + bc \tag{4}$$
$$= ac + i((a+b)(c+d) - ac - bc) + bc \tag{5}$$

Thus only requiring 3 real number multiplications.

b) First we calculate

$$a^2 - b^2 = (a+b)(a-b) \tag{1}$$

Then we substitute (1) into (2)

$$(a+ib)^2 = a^2 - b^2 + 2abi \tag{2}$$
$$= (a+b)(a-b) + 2abi \tag{3}$$

Thus only requiring 2 real number multiplications

c) By re-arranging by the laws of exponents:

$$(a+ib)^2(c+id)^2 = ((a+ib)(c+id))^2$$

Thus from above, we then calculate the middle multiplication using 3 real number multiplications, and then we find the square as above using 2 more real number multiplications.

Question 3

Expand $P(x)$ and $Q(x)$ as follows.

$$P(x) = a_0 + x^{17}(a_{17} + a_{19}x^2 + a_{21}x^4 + a_{23}x^6)$$
$$Q(x) = b_0 + x^{17}(b_{17} + b_{19}x^2 + b_{21}x^4 + b_{23}x^6)$$

let $y = x^2$ so that

$$R_a(y) = a_{17} + a_{19}y + a_{21}y^2 + a_{23}y^3$$
$$R_b(y) = b_{17} + b_{19}y + b_{21}y^2 + b_{23}y^3$$

then

$$P(x)Q(x) = a_0b_0 + x^{17}(a_0R_b(x^2) + b_0R_a(x^2)) + x^{34}R_a(x^2)R_b(x^2)$$

through brute force, we then calculate

a_0b_0 to require 1 multiplication

$a_0R_b(x^2)$ and $b_0R_a(x^2)$ to require 4 multiplications each

then to multiply $R_a(x^2)R_b(x^2)$ (of degree 3), we require $2(3)+1 = 7$ multiplications using the generalised Karatsuba method.
and so we get $1+4+4+7 = 16$ multiplications of large numbers.

Question 4

As $P(x)$ has all 15 roots of unity, and $x^{15} - 1$ and $P(x)$ are both of the same degree and are both monic then it follows that

$$P(x) = x^{15} - 1$$

Question 5

For any input $(a_0, a_1, a_2, ..., a_{2^n-1})$, we can describe a_i's new position by converting i to n binary places and finding the reversed sequence. e.g. $6 \rightarrow 110 \rightarrow 011 \rightarrow 3$

Question 6

let

$$f_m = \sum_{i+j=m} (j+1) q_j q_i \tag{1}$$

$$p_j = (j+1) q_j \tag{2}$$

then substitute (2) into (1)

$$\sum_{i+j=m} p_j q_i = \vec{p} * \vec{q} \tag{3}$$

as (3) is a linear convolution, f_m can be computed in $O(n \log n)$ time by transforming $\vec{p}$ and $\vec{q}$ to a point value representation using FFT, calculating the linear multiplication, and then transforming back to coefficient form using inverse FFT.

Question 7

a) Starting at $(0, 0)$ any spiral arrangement such as the one below would require n^2 queries to find the middle element as the local minimum.

16	15	14	13	12
17	18	19	20	11
2	1	0	21	10
3	24	23	22	9
4	5	6	7	8

b) Using the fact that a surface attains its minimal height either along its edges or in the interior, we can develop a divide-and-conquer algorithm by first finding the minimal height along a grid's boundary rows & columns, and center rows & columns as follows for an odd and even n.

25	75	63	34	9	45	49	57	77
3	58	6	51	4	27	39	18	76
62	33	46	59	36	11	50	5	42
10	0	15	60	55	35	74	28	14
30	68	78	21	31	29	54	73	65
40	48	80	43	44	38	37	23	72
19	7	22	32	2	1	67	70	71
64	61	56	8	79	16	52	17	69
13	47	12	20	66	41	26	24	53

16	8	25	24	22	38	37	53
54	3	27	17	57	10	35	62
4	28	5	52	34	56	42	18
20	12	6	11	21	49	26	61
46	33	2	60	40	41	58	14
48	47	51	9	43	13	23	19
45	7	30	63	36	50	0	29
39	44	15	59	55	31	1	32

We then check to see if the minimal height is a local minimum.
If it is, we return it, otherwise we recurse into the quadrant of its smallest neighbour, including our original boundary, since there must be a minimum in that direction as we would be following the slope of the curve. We do this as follows.

25	75	63	34	9	45	49	57	77
3	58	6	51	4	27	39	18	76
62	33	46	59	36	11	50	5	42
10	0	15	60	55	35	74	28	14
30	68	78	21	31	29	54	73	65
40	48	80	43	44	38	37	23	72
19	7	22	32	2	1	67	70	71
64	61	56	8	79	16	52	17	69
13	47	12	20	66	41	26	24	53

16	8	25	24	22	38	37	53
54	3	27	17	57	10	35	62
4	28	5	52	34	56	42	18
20	12	6	11	21	49	26	61
46	33	2	60	40	41	58	14
48	47	51	9	43	13	23	19
45	7	30	63	36	50	0	29
39	44	15	59	55	31	1	32

Using the master theorem, we can reduce the $n \times n$ matrix to a $\sim \frac{n}{2} \times \frac{n}{2}$ matrix with $O(n)$ queries.

$$\begin{aligned}
T(n) &= T(n/2) + cn \\
T(n) &= T(n/4) + c\frac{n}{2} + cn \\
T(n) &= T(n/8) + c\frac{n}{4} + c\frac{n}{2} + cn \\
&\vdots \\
T(n) &= T(1) + cn(1 + \frac{1}{2} + \frac{1}{4} + \ldots) \\
T(n) &= T(1) + 2cn
\end{aligned}$$

Thus, our algorithm is of $\Theta(n)$ time.

Question 8

let $A(k) = k^{th}$smallest element of A

let $B(k) = k^{th}$smallest element of B

let i be the current iteration (starting from 1)

let $s_f(i) = \lfloor \frac{n}{2^i} \rfloor, s_c(i) = \lceil \frac{n}{2^i} \rceil$

let $a = s_f(1), b = s_c(1)$

let $m_a = A(a+1), m_b = B(b+1)$

let N be the set of n elements such that $\forall x \in N \cap A, x < m_a$ and $\forall x \in N \cap B, x < m_b$

When all elements in N are smaller than m_a and m_b, the median will be the smallest of both m_a and m_b.

So we iterate.
if $B(b) > m_a$ then there exist elements in N that are not smaller than m_a:
  Therefore, we change m_a and m_b and increment i.
  We increase a by $s_c(i)$ to consider elements that may be smaller than the median.
  We decrease b by $s_f(i)$ to ignore elements that may be larger than the median.
else if $A(a) > m_b$ then there exist elements in N that are not smaller than m_b:
  Therefore, we change m_a and m_b and increment i.
  We decrease a by $s_f(i)$ to ignore elements that may be larger than the median.
  We increase b by $s_c(i)$ to consider elements that may be smaller than the median.
else:
  We stop iterating, as all elements of N are smaller than both m_a and m_b, and we take the median to be smaller of the two

This algorithm will take at most $O(\log n)$ queries, as every iteration we jump by a factor of 2, similar to that of a binary search.

Note: if $a + 1 > n$ assume m_b to be the median.