MATH 158
FINAL EXAM
20 DECEMBER 2016

Name : ______________________________________

Comment (2023): this exam is from an older version of this course, taught at Brown. There are some differences of style and emphasis compared to Math 252 here. Problems about topics we have not discussed are crossed out in this document.

- The exam is *double-sided.* Make sure to read both sides of each page.
- The time limit is three hours.
- No calculators are permitted.
- You are permitted one page of notes, front and back.
- The textbook's summary tables for the systems we have studied are provided at the back. There is also a multiplication table modulo 23. You may detach these sheets for easier reference.
- For any problem asking you to write a program, you may write in a language of your choice or in pseudocode, as long as your answer is sufficiently specific to tell the runtime of the program.

1	/12	**2**	/7
3	/7	**4**	/7
5	/7	**6**	/7
7	/7	**8**	/7
9	/7	**10**	/7
		$\sum$	/75

This page intentionally left blank.

(1) Briefly explain why each of the following choices is made in the cryptosystems we have studied (e.g. give a reason why it is necessary for the rest of the algorithm to work, why it makes a specific attack more difficult, or why it makes a computation more efficient).

(a) The element g in Diffie-Hellman (table 2.2) is chosen to have "large prime order."

(b) In Elgamal encryption (table 2.3) and digital signatures (table 4.2), the number k is chosen randomly each time a document is encrypted or signed.

(c) The decryption exponent e in RSA (table 3.1) satisfies $\gcd(e, (p-1)(q-1)) = 1$.

(d) The two primes p, q is DSA (table 4.3) satisfy $p \equiv 1 \pmod{q}$.

Parts (e-h) on reverse side.

(e) The prime p in ECDSA (table 6.7) can be chosen much smaller than the prime p in DSA (table 4.3).

(f) The primes p and q in ECDSA are roughly the same size (same number of bits in length).

(g) ~~In the congruential cryptosystem (table 7.1), the plaintext m is chosen less than $\sqrt{q/4}$, rather than less than $\sqrt{q/2}$ like the numbers f, g and r.~~

(h) ~~In NTRU (table 7.4), the element $f \in R$ is chosen from the set $\mathcal{T}(d+1, d)$ rather than from the set $\mathcal{T}(d, d)$ like the elements g and r. (Recall that the notation $\mathcal{T}(d_1, d_2)$ denotes the set of polynomials in R with d_1 coefficients equal to 1, d_2 coefficients equal to -1, and all other coefficients equal to 0.)~~

(12 points)

(2) Find the smallest positive integer n such that all three of the following congruences hold.

$$\begin{aligned} n &\equiv 3 \pmod{5} \\ n &\equiv 7 \pmod{8} \\ n &\equiv 0 \pmod{9} \end{aligned}$$

More space for work on reverse side. (7 points)

Additional space for problem 2.

(3) Let p be a prime number, and E be the elliptic curve over $\mathbb{F}_p$ described by $Y^2 \equiv X^3 + AX + B \pmod{p}$, where A and B are constants.

(a) Prove that given any integer x with $0 \leq x < p$, there are at most two integers y with $0 \leq y < p$ such that $(x, y) \in E(\mathbb{F}_p)$.

(3 points)

(b) Under what circumstances is there exactly *one* point on the elliptic curve with X-coordinate equal to x?

Part (c) on reverse side.

(1 point)

(c) Prove that if $P, Q \in E(\mathbb{F}_p)$ are two points on the elliptic curve with the same X-coordinate, and n is any integer, then either $n \cdot P$ and $n \cdot Q$ are both equal to the point $\mathcal{O}$ at infinity, or both have the same X-coordinate.

(3 points)

(4) Write a function `decipher(c,p,q,e)`, and any necessary helper functions, to decipher messages encrypted with RSA. The input consists of the ciphertext c, the secret primes p, q, and the encryption exponent e (notation as in table 3.1).

You should implement any helper functions you use that are not built into Python, or the standard programming language of your choice. You may assume that a fast modular exponentiation function `pow(a,b,m)` (returning $a^b\%m$) is built-in (as it is in Python).

More space for work on reverse side. (7 points)

Additional space for problem 4.

(5) ~~Suppose that Alice and Bob are using NTRU with parameters $(N, q, p, d) = (5, 23, 3, 1)$ (notation as in table 7.4). Alice's public key is~~

$$\mathbf{h} = 21 + 14x + 13x^2 + 4x^3 + 17x^4.$$

~~Bob wishes to encipher the message~~

$$\mathbf{m} = 1 + x + x^2 - x^4.$$

~~Find a valid ciphertext **c** that Bob might compute to send this message. (There are many possible answers; you only need to give one.)~~

~~Note that a multiplication table for **Z**/23 is provided at the back of the exam packet, which may be useful in your computations.~~

More space for work on reverse side. (7 points)

Additional space for problem 5.

(6) Let p be a prime number, and a an integer with $1 \leq a \leq p - 1$.

(a) Define the *order* of a modulo p.

(2 points)

(b) Define what it means for a to be a *primitive root* modulo p.

(2 points)

(c) Let $p = 7$. For each choice of a from 1 to 6 inclusive, determine the order of a, and identify whether or not it is a primitive root.

More space for work on reverse side. (3 points)

Additional space for problem 6.

(7) Each day, Alice and Bob perform Elliptic Curve Diffie-Hellman key exchange (notation as in table 6.5) to establish an encryption key for the day. Each day they use the same public parameters: the prime $p = 23$, curve $Y^2 \equiv X^3 + 2X + 6 \pmod{23}$, and the point $P = (1, 3)$.

On Monday, Alice and Bob exchange the values

$$Q_A = (18, 20) \qquad Q_B = (4, 3)$$

and establish the shared secret $S = (19, 7)$. Due to careless data management, Eve manages to learn *all three* of these values.

On Tuesday, Alice and Bob exchange the values

$$Q'_A = (5, 16) \qquad Q'_B = (18, 3)$$

and establish the shared secret S', which Eve is not able to intercept. However, Eve does notice that, due to poor random number generation by both Alice and Bob, these values are related to Monday's values by the equations

$$Q'_A = Q_A \oplus P \qquad Q'_B = 2 \cdot Q_B.$$

Use this information to determine the new shared secret S'. There is a multiplication table for $\mathbf{Z}/23$ at the back of the exam packet that may be useful in your computations. For partial credit you may express your answer in terms of the given points and elliptic curve operations; for full credit you should calculate the coordinates explicitly.

More space for work on reverse side. (7 points)

Additional space for problem 7.

(8) Comment (2023): I will not ask this type of estimation problem this year, since we put relatively less emphasis on the prime number theorem.

Estimate the number of 512-bit prime numbers (that is, prime numbers between 2^{511} and $2^{512} - 1$ inclusive). Your answer will be marked correct if it within a factor of 10 of the correct figure, and may be expressed in terms of standard mathematical functions (exponentials, logarithms, etc.).

(2 points)

(b) Assume that you have implemented a function `is_prime(n)` that efficiently determines whether or not n is prime, and returns either True or False. Write a function `safe_prime()` that returns a 512-bit prime number p such that the number $p - 1$ has at least one prime factor that is at least 256 bits long.

More space for work on reverse side. (5 points)

Additional space for problem 8.

(9) ~~Consider the following variation on the NTRU cryptosystem. In advance, Alice and Bob agree to the following public parameters.~~

$$N = 503, \quad q = 257, \quad p = 3$$

~~Privately, Alice chooses *three* polynomials at random, from the following sets. She keeps these polynomials secret; they constitute her private key.~~

$$\mathbf{f} \in \mathcal{T}(101, 100), \quad \mathbf{g}_1 \in \mathcal{T}(31, 30), \quad \mathbf{g}_2 \in \mathcal{T}(10, 10)$$

~~(Recall that $\mathcal{T}(d, e)$ denotes the subset of the ring $R = \mathbf{Z}[X]/(X^N - 1)$, where elements are represented as a list of N coefficients, consisting of polynomials with exactly d coefficients equal to 1, e coefficients equal to -1, and the rest of the coefficients equal to 0.) Alice ensures that $\mathbf{f}$ is invertible modulo q (otherwise she chooses a new value), with inverse $\mathbf{F}_q \in R_q$. She then computes the following two elements of R_q. She distributes these values; they constitute her public key.~~

$$\mathbf{h}_1 \equiv \mathbf{F}_q \star \mathbf{g}_1 \pmod{q}, \quad \mathbf{h}_2 \equiv \mathbf{F}_q \star \mathbf{g}_2 \pmod{q}$$

~~To send messages, Bob chooses a plaintext $\mathbf{m} \in R_p$, chooses a random ephemeral key $\mathbf{r} \in \mathcal{T}(10, 10)$, and computes a ciphertext $\mathbf{e} \in R_q$ as follows:~~

$$\mathbf{e} \equiv \mathbf{h}_1 \star \mathrm{cl}_p(\mathbf{m}) + p\mathbf{h}_2 \star \mathbf{r} \pmod{q}.$$

~~(Here cl_p denotes centerlifting from R_p to R; in the case $p = 3$ this gives a polynomials with all coefficients equal to $-1, 0,$ or 1.)~~

(a) ~~Describe a procedure that Alice can use to recover the plaintext $\mathbf{m}$ from the ciphertext $\mathbf{e}$. You may need to make an additional assumption about an element being invertible in a ring.~~

Part (b) on reverse side. (3 points)

(b) ~~Prove that the method you describe in part (a) will succeed, given the specific parameters specified above.~~

(4 points)

(10) Suppose that p and q are prime numbers, E is an elliptic curve over $\mathbb{F}_p$, and $G \in E(\mathbb{F}_p)$ is a point of order q.

Samantha and Victor are making use of the following signature scheme, similar to ECDSA. Samantha has a secret signing key s ($1 < s < q-1$), and a verification key $V = s \cdot G$, which is public information. A signature consists of a pair (s_1, s_2) of integers, both between 0 and $q-1$ inclusive, and a document consists of an integer d from 1 to $q-1$ inclusive. Victor will consider a signature (s_1, s_2) valid for the document d if the following equation holds.

$$x((d^{-1}s_1) \cdot V \oplus (d^{-1}s_2) \cdot G)\%q = s_1$$

Here d^{-1} denotes the inverse modulo q, and $x(P)$ denotes the x-coordinate of a point P on $E(\mathbb{F}_p)$.

(a) Suppose that Samantha wishes to sign a document d, and she begins by choosing a random ephemeral key e, and computing $s_1 = x(e \cdot G)\%q$. Explain a method Alice can use to compute a value s_2 such that (s_1, s_2) will be a valid signature for d.

Parts (b) and (c) on reverse side. (3 points)

(b) Suppose that Eve wishes to forge a valid signature for this system. As in the "blind forgery" methods we've discussed in class, she will not be able to choose the document d in advance. Instead, she begins by choosing two integers i and j at random from 1 to $q-1$ inclusive, and computes $s_1 = x(i \cdot G \oplus j \cdot V)\%q$. Explain a method Eve can use to compute a value of s_2 *and a value of* d, so that (s_1, s_2) will be a valid signature for the document d (even though d will likely appear to be gibberish).

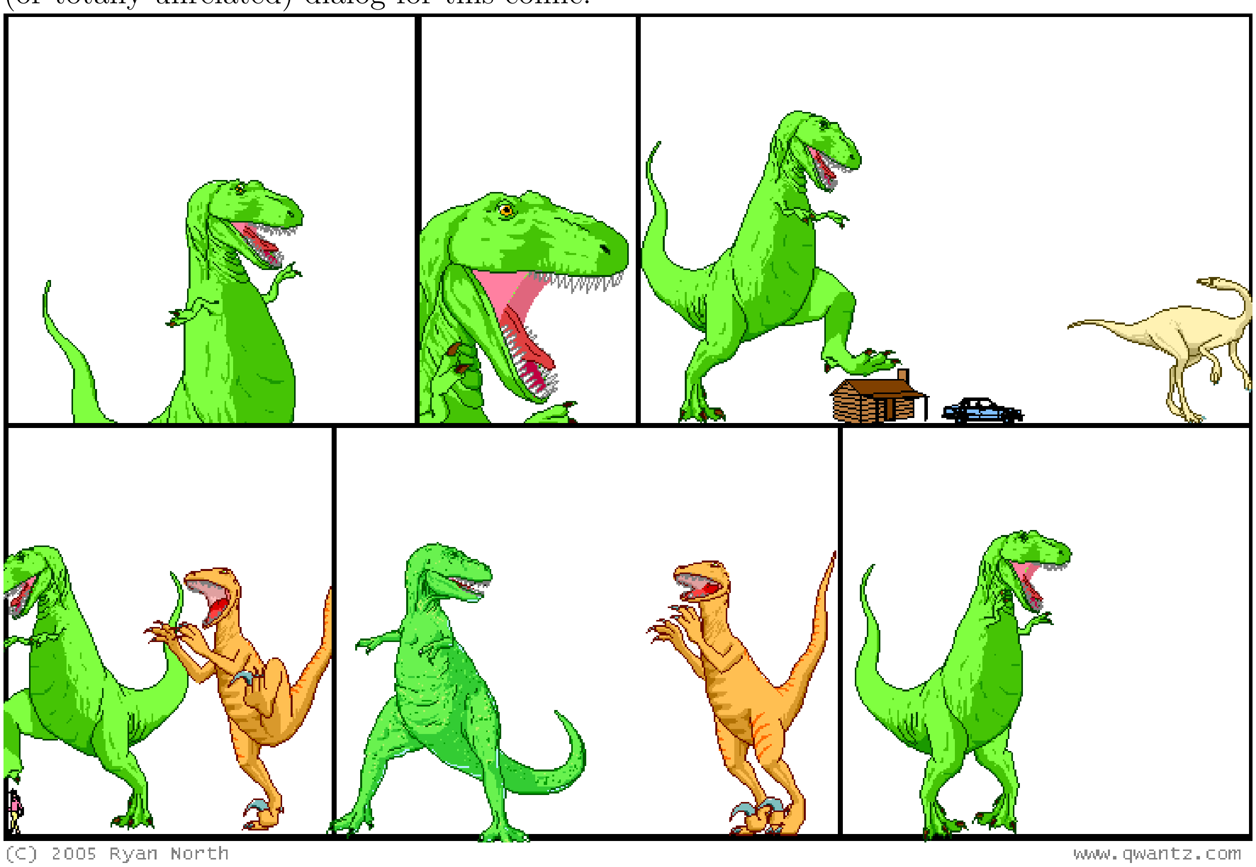
(3 points)

(c) Comment (2023): We are not discussing hash functions this semester.
~~Explain briefly how Samantha and Victor could modify this signature scheme using a hash function, in order to make Eve's method in (b) infeasible.~~

(1 point)

Additional space for work.

"Bonus" (to keep me happy during grading, not for real points): fill in cryptography-related (or totally unrelated) dialog for this comic.



Additional space for work.

Public parameter creation	
A trusted party chooses and publishes a (large) prime p and an integer g having large prime order in $\mathbb{F}_p^*$.	
Private computations	
Alice	**Bob**
Choose a secret integer a. Compute $A \equiv g^a \pmod p$.	Choose a secret integer b. Compute $B \equiv g^b \pmod p$.
Public exchange of values	
Alice sends A to Bob $\longrightarrow$	A
B $\longleftarrow$	Bob sends B to Alice
Further private computations	
Alice	**Bob**
Compute the number $B^a \pmod p$.	Compute the number $A^b \pmod p$.
The shared secret value is $B^a \equiv (g^b)^a \equiv g^{ab} \equiv (g^a)^b \equiv A^b \pmod p$.	

Table 2.2: Diffie–Hellman key exchange

Public parameter creation	
A trusted party chooses and publishes a large prime p and an element g modulo p of large (prime) order.	
Alice	**Bob**
Key creation	
Choose private key $1 \le a \le p-1$. Compute $A = g^a \pmod p$. Publish the public key A.	
Encryption	
	Choose plaintext m. Choose random element k. Use Alice's public key A to compute $c_1 = g^k \pmod p$ and $c_2 = mA^k \pmod p$. Send ciphertext (c_1, c_2) to Alice.
Decryption	
Compute $(c_1^a)^{-1} \cdot c_2 \pmod p$. This quantity is equal to m.	

Table 2.3: Elgamal key creation, encryption, and decryption

Bob	**Alice**
Key creation	
Choose secret primes p and q. Choose encryption exponent e with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and e.	
Encryption	
	Choose plaintext m. Use Bob's public key (N, e) to compute $c \equiv m^e \pmod N$. Send ciphertext c to Bob.
Decryption	
Compute d satisfying $ed \equiv 1 \pmod{(p-1)(q-1)}$. Compute $m' \equiv c^d \pmod N$. Then m' equals the plaintext m.	

Table 3.1: RSA key creation, encryption, and decryption

Samantha	**Victor**
Key creation	
Choose secret primes p and q. Choose verification exponent e with $\gcd(e, (p-1)(q-1)) = 1$. Publish $N = pq$ and e.	
Signing	
Compute d satisfying $de \equiv 1 \pmod{(p-1)(q-1)}$. Sign document D by computing $S \equiv D^d \pmod N$.	
Verification	
	Compute $S^e \bmod N$ and verify that it is equal to D.

Table 4.1: RSA digital signatures

Public parameter creation	
A trusted party chooses and publishes a large prime p and primitive root g modulo p.	
Samantha	**Victor**
Key creation	
Choose secret signing key $1 \le a \le p-1$. Compute $A = g^a \pmod p$. Publish the verification key A.	
Signing	
Choose document $D \bmod p$. Choose random element $1 < k < p$ satisfying $\gcd(k, p-1) = 1$. Compute signature $S_1 \equiv g^k \pmod p$ and $S_2 \equiv (D - aS_1)k^{-1} \pmod{p-1}$.	
Verification	
	Compute $A^{S_1} S_1^{S_2} \bmod p$. Verify that it is equal to $g^D \bmod p$.

Table 4.2: The Elgamal digital signature algorithm

Public parameter creation	
A trusted party chooses and publishes large primes p and q satisfying $p \equiv 1 \pmod q$ and an element g of order q modulo p.	
Samantha	**Victor**
Key creation	
Choose secret signing key $1 \le a \le q-1$. Compute $A = g^a \pmod p$. Publish the verification key A.	
Signing	
Choose document $D \bmod q$. Choose random element $1 < k < q$. Compute signature $S_1 \equiv (g^k \bmod p) \bmod q$ and $S_2 \equiv (D + aS_1)k^{-1} \pmod q$.	
Verification	
	Compute $V_1 \equiv DS_2^{-1} \pmod q$ and $V_2 \equiv S_1 S_2^{-1} \pmod q$. Verify that $(g^{V_1} A^{V_2} \bmod p) \bmod q = S_1$.

Table 4.3: The digital signature algorithm (DSA)

Public parameter creation	
A trusted party chooses and publishes a (large) prime p, an elliptic curve E over $\mathbb{F}_p$, and a point P in $E(\mathbb{F}_p)$.	
Private computations	
Alice	**Bob**
Chooses a secret integer n_A. Computes the point $Q_A = n_A P$.	Chooses a secret integer n_B. Computes the point $Q_B = n_B P$.
Public exchange of values	
Alice sends Q_A to Bob $\longrightarrow$	Q_A
Q_B $\longleftarrow$	Bob sends Q_B to Alice
Further private computations	
Alice	**Bob**
Computes the point $n_A Q_B$.	Computes the point $n_B Q_A$.
The shared secret value is $n_A Q_B = n_A(n_B P) = n_B(n_A P) = n_B Q_A$.	

Table 6.5: Diffie–Hellman key exchange using elliptic curves

Public parameter creation	
A trusted party chooses a finite field $\mathbb{F}_p$, an elliptic curve $E/\mathbb{F}_p$, and a point $G \in E(\mathbb{F}_p)$ of large prime order q.	
Samantha	**Victor**
Key creation	
Choose secret signing key $1 < s < q-1$. Compute $V = sG \in E(\mathbb{F}_p)$. Publish the verification key V.	
Signing	
Choose document $d \bmod q$. Choose random element $e \bmod q$. Compute $eG \in E(\mathbb{F}_p)$ and then, $s_1 = x(eG) \bmod q$ and $s_2 \equiv (d + s s_1)e^{-1} \pmod q$. Publish the signature (s_1, s_2).	
Verification	
	Compute $v_1 \equiv d s_2^{-1} \pmod q$ and $v_2 \equiv s_1 s_2^{-1} \pmod q$. Compute $v_1 G + v_2 V \in E(\mathbb{F}_p)$ and verify that $x(v_1 G + v_2 V) \bmod q = s_1$.

Table 6.7: The elliptic curve digital signature algorithm (ECDSA)

Public Parameter Creation	
A trusted party chooses and publishes a (large) prime p, an elliptic curve E over $\mathbb{F}_p$, and a point P in $E(\mathbb{F}_p)$.	
Alice	**Bob**
Key Creation	
Chooses a secret multiplier n_A. Computes $Q_A = n_A P$. Publishes the public key Q_A.	
Encryption	
	Chooses plaintext values m_1 and m_2 modulo p. Chooses a random number k. Computes $R = kP$. Computes $S = kQ_A$ and writes it as $S = (x_S, y_S)$. Sets $c_1 \equiv x_S m_1 \pmod p$ and $c_2 \equiv y_S m_2 \pmod p$. Sends ciphertext (R, c_1, c_2) to Alice.
Decryption	
Computes $T = n_A R$ and writes it as $T = (x_T, y_T)$. Sets $m_1' \equiv x_T^{-1} c_1 \pmod p$ and $m_2' \equiv y_T^{-1} c_2 \pmod p$. Then $m_1' = m_1$ and $m_2' = m_2$.	

Table 6.13: Menezes–Vanstone variant of Elgamal (Exercises 6.17, 6.18)

Alice	**Bob**
Key Creation	
Choose a large integer modulus q. Choose secret integers f and g with $f < \sqrt{q/2}$, $\sqrt{q/4} < g < \sqrt{q/2}$, and $\gcd(f, qg) = 1$. Compute $h \equiv f^{-1} g \pmod q$. Publish the public key (q, h).	
Encryption	
Choose a random $r < \sqrt{q/2}$	Choose plaintext m with $m < \sqrt{q/4}$. Use Alice's public key (q, h) to compute $e \equiv rh + m \pmod q$. Send ciphertext e to Alice.
Decryption	
Compute $a \equiv fe \pmod q$ with $0 < a < q$. Compute $b \equiv f^{-1} a \pmod g$ with $0 < b < g$. Then b is the plaintext m.	

Table 7.1: A congruential public key cryptosystem

Public parameter creation	
A trusted party chooses public parameters (N, p, q, d) with N and p prime, $\gcd(p, q) = \gcd(N, q) = 1$, and $q > (6d+1)p$.	
Alice	**Bob**
Key creation	
Choose private $\boldsymbol{f} \in \mathcal{T}(d+1, d)$ that is invertible in R_q and R_p. Choose private $\boldsymbol{g} \in \mathcal{T}(d, d)$. Compute $\boldsymbol{F}_q$, the inverse of $\boldsymbol{f}$ in R_q. Compute $\boldsymbol{F}_p$, the inverse of $\boldsymbol{f}$ in R_p. Publish the public key $\boldsymbol{h} = \boldsymbol{F}_q \star \boldsymbol{g}$.	
Encryption	
Should be $cl_p(m)$ (centerlift from R_p to R) →	Choose plaintext $\boldsymbol{m} \in R_p$. Choose a random $\boldsymbol{r} \in \mathcal{T}(d, d)$. Use Alice's public key $\boldsymbol{h}$ to compute $\boldsymbol{e} \equiv p\boldsymbol{r} \star \boldsymbol{h} + \boldsymbol{m} \pmod q$. Send ciphertext $\boldsymbol{e}$ to Alice.
Decryption	
Compute $\boldsymbol{f} \star \boldsymbol{e} \equiv p\boldsymbol{g} \star \boldsymbol{r} + \boldsymbol{f} \star \boldsymbol{m} \pmod q$. Center-lift to $\boldsymbol{a} \in R$ and compute $\boldsymbol{m} \equiv \boldsymbol{F}_p \star \boldsymbol{a} \pmod p$.	

Table 7.4: NTRUEncryt: the NTRU public key cryptosystem

Relevant definitions: (in NTRU)

$R = \mathbb{Z}[x]/(x^N - 1)$; elements represented by N coefficients.

$\mathcal{T}(d_1, d_2)$ = elements of R with exactly d_1 coefficients equal to 1, d_2 coefficients equal to -1 & the rest equal to 0.

$R_q = (\mathbb{Z}/q)[x]/(x^N - 1)$

Reference information. You may detach this sheet for easier use.

Multiplication table modulo 23

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

This page intentionally left blank.