6. RSA 암호

// RSA 암호에서 키 생성

① 두 개의 큰 소수 p, q(p≠q)를 임의로 선택한다. 소수인지 검사해야 한다.
→ p와 q는 적어도 512bit가 되도록 한다. 매우 큰 소수가 된다.

② 두 소수를 곱한 것을 n, 두 소수보다 1씩 작은 수를 곱한 수를 Phi(n)이라 한다.
`n = p × q` //p, q의 보안은 매우 중요, 키 생성 후 p, q는 삭제 가능
`Phi(n) = (p-1)(q-1)` //Φ(n) = (p-1)(q-1)

③ 1 < e < Φ(n)인 정수 중에서 Φ(n)과 **서로소** 관계가 되는 임의의 수 e를 선택한다.
`gcd(e, Φ(n)) = 1` //e는 공개키로 사용된다.

④ d와 e를 곱하여 Φ(n)으로 나눈 나머지가 1이 되는 d를 구한다.
`(d × e) mod Φ(n) = 1` //d × e ≡ 1 mod Φ(n) → d는 개인키가 된다.

↓

생성된 키 : 공개키는 (e, n)인 숫자쌍이고, 개인키는 (d, n)인 숫자쌍이다.

◉ RSA 알고리즘 분석

① 두 소수 p와 q를 모를 때,
공개키 (e, n)을 이용하여 개인키 (d, n)을 얼마나 빨리 구할 수 있을까?
→ p와 q에 의하여 최종적으로 e, d가 구해진다.
② 만약, n을 소인수분해하여 p와 q를 구할 수 있다면 d를 쉽게 계산할 수 있다.
③ 하지만, 소수 p와 q가 충분히 크다면
두 소수의 곱으로부터 원래의 두 소수를 찾는 것은 현재 컴퓨터 환경으로는 거의 불가능
④ RSA에 의해 만들어진 키를 풀려면 현재 가장 빠른 컴퓨터로 수년이 걸린다.
⑤ RSA 알고리즘 안전성은 소인수분해하는 것은 매우 어렵다는 문제 기반으로 설계되었다.
→ 서로 다른 큰 두 소수를 곱한 결과인 큰 수는 원래의 두 소수를 찾아내는 것이 어렵다.
⑥ RSA는 1977년에 미국 메사추세츠 공과대학 Rivest-Shamir-Adleman이 만들었다.

예제 1	RSA 암호시스템에 의한 키 생성 및 메시지 암복호화

① 먼저, 서로 다른 임의의 두 소수를 선택한다.
p = **7**, q = **11**

② n = p × q를 구한다.
n = p × q = 7 × 11 = **77**

③ n의 오일러 파이 함수 $\Phi(n) = (p-1)(q-1)$을 구한다.
$\Phi(n) = \Phi(77) = (p-1)(q-1) = (7-1)(11-1) =$ **60**

④ **1 〈 e 〈 60**인 정수 중에서 $\Phi(n)$과 **서로소** 관계가 되는 임의의 수 **e**를 선택한다.
→ 이를 어렵게 표현하면, Z_{60}^{*}에서 임의의 수 **e**를 선택한다.
→ **e = 13**이 선택된 것으로 가정한다.**(e는 공개키)**

⑤ d와 e를 곱하여 $\Phi(n)$으로 나눈 **나머지가 1**이 되는 **d**를 구한다.
$(d \times e) \bmod \Phi(n) =$ **1** → 즉, $(d \times e) \bmod 60 = 1$로 d와 e는 서로 **역원** 관계이다.
$(d \times$ **13**$) \bmod 60 =$ **1**
d = **37**

⑥ 최종적으로 생성된 키 관계는 다음과 같다.
공개키 **(e, n) = (13, 77)**
개인키 **(d, n) = (37, 77)**

⑦ 평문 **p=2**에 대한 **암복호화** 과정은 다음과 같다.

평 문	2
암호화	$c = p^e \bmod n = 2^{13} \bmod 77 =$ **30** $\bmod 77$
암호문	30
복호화	$p = c^d \bmod n = 30^{37} \bmod 77 =$ **2** $\bmod 77$
평 문	2

• RSA 암호시스템의 메시지 암복호화는 서로 역관계에 있다.

예제 2	RSA 암호시스템에 의한 키 생성 및 메시지 암복호화

① 먼저, 서로 다른 임의의 두 소수를 선택한다.
p = **13**, q = **17**

② n = p × q를 구한다.
n = p × q = 13 × 17 = **221**

③ n의 오일러 파이 함수 $\Phi(n)$ = (p−1)(q−1)을 구한다.
$\Phi(n) = \Phi(221) = (p-1)(q-1) = (13-1)(17-1) =$ **192**

④ **1 〈 e 〈 192**인 정수 중에서 $\Phi(n)$과 **서로소** 관계가 되는 임의의 수 **e**를 선택한다.
→ 이를 어렵게 표현하면, Z_{192}^{*} 에서 임의의 수 **e**를 선택한다.
→ **e = 7**이 선택된 것으로 가정한다.**(e는 공개키)**

⑤ d와 e를 곱하여 $\Phi(n)$으로 나눈 **나머지가 1**이 되는 **d**를 구한다.
(d × e) mod $\Phi(n)$ = **1** → 즉, (d × e) mod 192 = 1로 d와 e는 서로 **역원** 관계이다.
(d × **7**) mod 192 = **1**
d = **55**

⑥ 최종적으로 생성된 키 관계는 다음과 같다.
공개키 **(e, n) = (7, 221)**
개인키 **(d, n) = (55, 221)**

⑦ 평문 **p=5**에 대한 **암복호화** 과정은 다음과 같다.

평 문	5
암호화	c = p^e mod n = 5^7 mod 221 = **112** mod 221
암호문	112
복호화	p = c^d mod n = 112^{55} mod 221 = **5** mod 221
평 문	5

• RSA 암호시스템의 메시지 암복호화는 서로 역관계에 있다.

◈ RSA 암호를 이용한 메시지 암호화 및 복호화 / 전자서명

// 놀부가 흥부에게 메시지 m을 보낼 때

	진행 과정
메시지 암호화 복호화	• 놀부는 흥부의 **공개키** (e, n)을 이용하여 메시지를 다음처럼 **암호화**한다. • c = m^e mod n → 메시지 암호화, c는 암호화된 메시지 • 암호화된 메시지 c를 흥부에게 보낸다. • 흥부는 **개인키** (d, n)을 이용하여 원래의 메시지 m을 **복호화**한다. • m = c^d mod n → 메시지 복호화,
전자서명	• 놀부는 자신의 **개인키** (d, n)을 이용하여 메시지에 **전자서명**을 한다. • c = m^d mod n → 전자서명, c는 전자서명을 수행한 결과 값 • 전자서명된 메시지 c를 흥부에게 보낸다. • 흥부는 놀부의 **공개키** (e, n)을 이용하여 **전자서명을 검증**한다. • m = c^e mod n → 전자서명검증, m은 문서(메시지)

- 놀부는 메시지 m을 n보다 작은 숫자로 바꾼다.
- 만약, 메시지가 크면 토막 내어 하나의 메시지가 일정 비트를 넘지 않게 한다.
- 실제 이용에서는 이중 보안을 위해 매우 복잡한 변환법이 사용될 수 있다.

// RSA 암호 알고리즘 권장사항 – Behrouz A. Forouzan 참조

- 법(modular, 모듈로)으로 사용하는 n의 비트수는 최소한 1024비트가 되도록 한다.
- 2^{1024}은 10진수로 309자리 정도이다.
- 두 소수 p와 q는 최소한 512비트가 되도록 한다.
- 2^{512}은 10진수로 154자리 정도이다.
- 모듈로로 사용하는 n은 공동으로 사용하지 않도록 한다.
- p와 q는 서로 매우 가까이 있는 수가 되지 않도록 한다.
- p-1, q-1은 각각 최소한 하나의 큰 소인수를 가지도록 한다.
- p와 q의 비율 p/q는 작은 분자나 분모를 갖는 유리수와 가까이 있지 않도록 한다.
- 공개키 e 값은 2^{16}+1이거나, 이 값에 가까운 정수를 사용하도록 한다.
- 만약, 개인키 d가 유출되면 즉시 n, e, d를 모두 변경해야 한다.

// 우리나라 법에서 권장하는 비트 수(2018년 7월 5일부터 시행)

무결성과 전자서명을 위한 암호 기술은 각각 최소 SHA-256과 **RSA-2048비트** 이상
또는 이에 준하는 안전성이 입증된 해시함수와 비 대칭키 암호화 알고리즘을 이용해야 한다.

Java로 구현한 RSA 알고리즘 - 암호화, 복호화 까지

```
import java.io.*;
import java.math.BigInteger;              //빅데이터 처리
import java.security.SecureRandom;        //난수 처리
public class TestRSA{
 public static void main(String[] args) throws Exception{
   int bitLen = 128;    //키 길이 결정(0 ~ 2^128 - 1)
   int prime = 32;      //소수 난수 생성 확률 상대 지수 결정, 소수일 확률은 1-(0.5)^32
   SecureRandom srnd = new SecureRandom();                  //난수 생성 준비
   BufferedReader in = new BufferedReader(new InputStreamReader(System.in));
   System.out.print("평문 입력(주민번호) : "); String input = in.readLine();
   byte[] byteArray = input.getBytes();                     //평문 입력을 byte 배열로
   BigInteger m = new BigInteger(1, byteArray);             //byte 배열을 BigInteger형으로
   BigInteger p = new BigInteger(bitLen, prime, srnd);      //난수 이용 큰 소수 생성
   BigInteger q = new BigInteger(bitLen, prime, srnd);      //난수 이용 큰 소수 생성
   BigInteger n = p.multiply(q);                            //n = p * q;
   BigInteger one = new BigInteger("1");                    //빅데이터 처리 방식
   BigInteger pi = p.subtract(one).multiply(q.subtract(one)); //pi = (p-1) * (q-1);
   BigInteger e;                                            //공개키 생성 준비
   do{
     e = new BigInteger(p.bitLength(), srnd);               //난수에 의한 공개키 생성
   }while(!e.gcd(pi).equals(one));                          //e와 pi의 관계가 서로소인지 검사
   BigInteger d = e.modInverse(pi);                         //개인키 생성
   System.out.println("p : " + p);       System.out.println("q : " + q);
   System.out.println("n : " + n);       System.out.println("pi : " + pi);
   System.out.println("공개키(e) : " + e);  System.out.println("개인키(d) : " + d);
   BigInteger encText = m.modPow(e, n);                     //공개키를 이용한 암호화
   System.out.println("암호문(주민번호) : " + encText);          //암호화된 암호문 출력
   BigInteger dec = encText.modPow(d, n);                   //개인키를 이용한 복호화
   byte[] decBytes = dec.toByteArray();                     //BigInteger형을 바이트 배열로
   String decText = new String(decBytes);                   //바이트 배열을 스트링으로
   System.out.println("평문(주민번호) : " + decText);            //복호화된 평문 출력
 }
}
```

//실행 결과는 다음 쪽에 있다.

평문 입력(주민번호) : 881225-1234567

p : 250107907813426253157306528728909612641

q : 250785524872924859948968981558510791803

n : 62723442935859207533576529137577772771678718040144921043199163713942927981723

pi : 627234429358592075335765291375777727711778246074585699300928882036555075772 80

공개키(e) : 291662137868829296759136418144303406 23

개인키(d) : 53873171732208217737038650266726908696486683390921866383994146296596329091 67

암호문(주민번호) : 5908701514144550551762767769863897509743580565357146164765206200014566038818 1

평문(주민번호) : 881225-1234567 → 복호화 결과

• 키 길이가 매우 긴 것을 알 수 있다.

◉ Java에서 빅데이터 처리

• $c^d = 112^{55}$은 빅데이터이다. 다음 자바코드를 이용하여 확인할 수 있다.

```
import java.math.BigInteger;                  //빅데이터 처리 클래스
public class Test{
  public static void main(String[] args){
    BigInteger c = new BigInteger("112");     //빅데이터 처리 방식
    BigInteger n = new BigInteger("221");
    int d = 55;                               //지수(지수값은 빅데이터가 아님)
    BigInteger r = c.pow(d);                  //112^55 구하기
    System.out.println(r);                    //빅데이터, 아주 큰 수가 출력
    System.out.println(r.mod(n));             //원래의 메시지 5 출력
  }
}
```

RSA 알고리즘 추가 정리

① RSA 알고리즘에서 **공개키, 개인키**를 구한 후에는 두 소수 p, q는 더 이상 필요 없다.

② RSA 알고리즘으로 암호화할 때, **큰 수를 직접 소인수분해** 하는가?
→ 아니다. 서로 다른 임의의 두 소수를 선택하여 이용할 뿐이다.
→ RSA 알고리즘은 암호화, 복호화, 키 생성에서 수를 소인수분해하지 않는다.

③ RSA 알고리즘을 해독하는 어려움은 큰 수를 소인수분해하는 것과 같은가?
→ 같은지? 아닌지? 아직 모른다.
→ 소인수분해 문제와 같다고 추측하지만 아직 증명되지 않았다.

④ 큰 수 소인수분해를 고속으로 할 수 있다면, RSA 알고리즘은 해독되는가?
→ RSA 알고리즘 해독은 반드시 소인수분해해야 한다는 것으로 증명된 것은 아니다.
→ 소인수분해를 하지 않아도 해독할 수 있는 방법이 있을 수도 있다.

⑤ 큰 수 N이 소인수분해 되지 않기 위해서는 몇 비트의 길이가 필요한가?
→ 비트 길이 매우 길어도 기술이 발전되면, 언젠가는 소인수분해 될 수 있다.

⑥ 서로 다른 두 사람이 우연하게 같은 소수를 선택하지 않을까?
→ 소수가 무수히 많으므로 같은 소수를 선택할 확률은 극히 낮다.
→ 512bit 크기 범위에서 대략 10^{151}개의 소수가 있다.

⑦ DES와 RSA의 속도는
소프트웨어 부분에서는 DES가 RSA보다 100배 정도 더 빠르고,
하드웨어 부분에서는 DES가 RSA보다 1,000~10,000배 정도 더 빠르다.

기출문제 분석

1. 공개키 암호인 RSA의 특징에 대한 설명으로 옳지 않은 것은? [2015년 국회 9급]

① 매우 큰 소수를 사용하여 키를 만든다.
② 암•복호화 과정에 계산량이 많다.
③ 개인 인증서에도 사용한다.
④ 키를 교환해야 하는 불편함이 있다.
⑤ 디지털 서명에도 사용한다.

♤ RSA의 특징

- 키를 교환해야 하는 불편함이 있다.(×)
- 공개키 암호는 송수신자가 키를 미리 서로 교환할 필요가 없다.
- 공개키 암호는 암호화와 복호화에 서로 **다른 키**를 사용한다.(비대칭키)
- 공개키 암호는 암복호화를 위해서 반드시 한 쌍의 키가 필요하다.(공개키, 개인키)

정답 : ④

2. RSA에 대한 설명으로 가장 옳지 않은 것은? [2018년 서울 9급]

① AES에 비하여 암, 복호화 속도가 느리다.
② 키 길이가 길어지면 암호화 및 복호화 속도도 느려진다.
③ 키 생성에 사용되는 서로 다른 두 소수(p, q)의 길이가 길어질수록 개인키의 안전성은 향상된다.
④ 중간자(man-in-the-middle) 공격으로부터 안전하기 위해서는 2,048비트 이상의 공개키를 사용하면 된다.

♤ RSA 암호 알고리즘

- 중간자(man-in-the-middle) 공격으로부터 안전하기 위해서는 2,048비트 이상의 공개키를 사용하면 된다.(×)
  → 현재, RSA에 대한 치명적인 공격 방법은 얼려진 것이 없다.
  → 소인수분해, 선택 암호문, 복호화 지수, 암호화 지수 등에 근거한 공격이 예상될 뿐이다.
  → 그리고, 공개키는 누구나 알 수 있도록 공개하는 것이므로 중간자 공격과 무관하다.

정답 : ④

3. RSA 암호 알고리즘에 대한 설명으로 옳지 않은 것은? [2018년 국가 7급]

① 대표적인 비대칭 암호 알고리즘으로, 널리 사용되고 있다.
② 공개키 {e, n}이 주어지면 지수 및 모듈러 연산을 통해 n과 무관한 임의 크기의 평문 블록을 하나의 암호문 블록으로 암호화할 수 있다.
③ 공개키 {e, n}의 n을 소인수분해할 수 있으면 개인키 {d, n}의 d를 알아낼 수 있다.
④ 일반적으로 키의 길이가 길수록 안전성은 높아지지만 알고리즘 수행시간은 길어진다.

♤ RSA 암호 알고리즘

- 공개키 {e, n}이 주어지면 지수 및 모듈러 연산을 통해 n과 무관한 임의 크기의 평문 블록을 하나의 암호문 블록으로 암호화할 수 있다.(×)
  → 암호화 : c = m^e mod n, 암호화는 n과 무관한 것이 아니다. n과 연관을 가진다.

정답 : ②

4. 다음의 지문은 RSA 알고리즘의 키생성 적용 순서를 설명한 것이다. ( )를 바르게 채운 것은? [2017년 서울 9급]

ㄱ	두 개의 큰 소수 p와 q를 생성한다. (p≠q)
ㄴ	두 소수를 곱하여, n=p · q를 계산한다.
ㄷ	(㉮)을 계산한다.
ㄹ	1 < A < Φ(n)이면서 A, Φ(n)이 서로소가 되는 A를 선택한다. A · B를 Φ(n)으로 나눈 나머지가 1임을 만족하는 B를 계산한다.
ㅁ	공개키로 (㉯), 개인키로 (㉰)를 각각 이용한다.

	㉮	㉯	㉰
①	Φ(n)=(p−1)(q−1)	(n, A)	(n, B)
②	Φ(n)=(p+1)(q+1)	(n, B)	(n, A)
③	Φ(n)=(p−1)(q−1)	(n, B)	(n, A)
④	Φ(n)=(p+1)(q+1)	(n, A)	(n, B)

♤ RSA 암호 알고리즘

ㄷ. (㉮)을 계산한다. → 예 : Φ(n) = Φ(221) = (p−1)(q−1) = (13−1)(17−1) = 192
ㅁ. 공개키로 (㉯), 개인키로 (㉰)를 각각 이용한다.
→ 예 : 공개키 (n, A) = (221, 7) / 개인키 (n, B) = (221, 55)

정답 : ①

5. RSA 암호시스템에서 어떤 사용자의 공개키를 {e, n}이라 할 때, 평문 블록 M과 암호문 블록 C는 수식, $C = M^e \bmod n$을 만족한다. n을 두 소수 11과 13의 곱이라 할 때, e로 선택할 수 있는 것만을 모두 고른 것은? [2017년 법무부 9급]

ㄱ. 9 ㄴ. 17 ㄷ. 19 ㄹ. 127

---

① ㄴ, ㄷ ② ㄱ, ㄴ, ㄷ ③ ㄴ, ㄷ, ㄹ ④ ㄱ, ㄴ, ㄷ, ㄹ

♤ **RSA 암호시스템**

① 먼저, 서로 다른 임의의 두 소수를 선택한다.
$p = 11, q = 13$

② $n = p \times q$를 구한다.
$n = p \times q = 11 \times 13 = 143$

③ n의 오일러 파이 함수 $\Phi(n) = (p-1)(q-1)$을 구한다.
$\Phi(n) = \Phi(143) = (p-1)(q-1) = (11-1)(13-1) =$ **120**

④ $1 < e < 120$인 정수 중에서 $\Phi(n)$과 **서로소** 관계가 되는 임의의 수 e를 선택한다.
→ 이를 어렵게 표현하면, Z_{192}^* 내에서 임의의 수 e를 선택한다.

// e로 선택할 수 있는 것(120과 서로소 관계인 것)

(9, 120)	• 9와 120은 **서로소 관계가 아니다.** • 최대공약수 gcd가 **3**이므로	**선택 불가**
(17, 120)	• 17과 120은 서로소 관계이다. • 최대공약수 gcd가 **1**이므로	선택 가능
(19, 120)	• 19와 120은 서로소 관계이다. • 최대공약수 gcd가 **1**이므로	선택 가능
127	• $1 < e < 120$ 조건을 만족하지 않는다.	**선택 불가**

◉ **서로소(relatively prime / disjoint)**
• 최대공약수 gcd가 1인 두 정수를 서로소라 한다.
• 다르게 정의하면, 서로소는 1과 -1 이외에는 공약수를 갖지 않는 두 정수이다.
• 두 정수의 쌍 (4, 9), (7, 13), (3, 4) 등은 각각 서로소 관계이다.
4의 약수 = { -4, -2, -1, 1, 2, 4 }이고,
9의 약수 = { -9, -3, -1, 1, 3, 9 }이다.
• 4와 9의 공약수는 1과 -1 뿐이다. 4와 9는 서로소이고, (4, 9) = 1이다.

정답 : ①

6. 두 소수, p=13, q=11을 사용하는 RSA 시스템에서 키값 (e, d)로 사용할 수 있는 쌍은? [2019년 국가 7급]

① (7, 11)
② (7, 23)
③ (13, 37)
④ (13, 47)

♤ **RSA 암호시스템**

① 먼저, 서로 다른 임의의 두 소수를 선택한다.
p = 13, q = 11

② n = p × q를 구한다.
n = p × q = 13 × 11 = 143 (143은 소수가 아니고 합성수이다)

③ n의 오일러 파이 함수 $\Phi(n)$ = (p−1)(q−1)을 구한다.
$\Phi(n) = \Phi(143) = (p-1)(q-1) = (13-1)(11-1) =$ **120**

④ 1 < e < 120인 정수 중에서 $\Phi(n)$과 **서로소** 관계가 되는 임의의 수 e를 선택한다.
→ 이를 어렵게 표현하면, Z_{192}^{*} 에서 임의의 수 e를 선택한다.(e는 공개키)
→ 최대공약수 gcd가 1인 두 정수를 서로소라 한다.

⑤ e와 d를 곱하여 $\Phi(n)$으로 나눈 나머지가 1이 되는 d를 구한다.
(e × d) % $\Phi(n)$ = 1 → 즉, (d × e) mod **120** = 1로 d와 e는 서로 역원 관계이다.
↓
↓ 이를 만족하는 쌍은 (13, 37)이다.
↓
(13 × 37) % 120 = 1
↓
481 % 120 = 1

• (e, d)로 사용할 수 있는 쌍은 **(13, 37)**이다.

정답 : ③

7. RSA 암호 알고리즘에서 두 소수, p=17, q=23과 키 값 e=3을 선택한 경우, 평문 m=8에 대한 암호문 c로 옳은 것은? [2020년 지방 9급]

① 121 ② 160
③ 391 ④ 512

♤ **RSA 암호 알고리즘**

① 먼저, 서로 다른 임의의 두 소수를 선택한다.
p = 17, q = 23

② n = p × q를 구한다.
n = p × q = 17 × 23 = **391**

③ n의 오일러 파이 함수 $\Phi(n)$ = (p−1)(q−1)을 구한다.
Φ(n) = $\Phi(391)$ = (p−1)(q−1) = (17−1)(23−1) = **352**

④ 1 < e < 352인 정수 중에서 $\Phi(n)$과 **서로소** 관계가 되는 임의의 수 e를 선택한다.
→ 이를 어렵게 표현하면, Z_{352}^{*} 에서 임의의 수 e를 선택한다.
→ 주어진 문제에서 **e=3**이 선택되었다.(e는 공개키)

⑤ d와 e를 곱하여 $\Phi(n)$으로 나눈 나머지가 1이 되는 d를 구한다.
(d × e) % $\Phi(n)$ = 1 → 즉, (d × e) mod 352 = 1로 d와 e는 서로 역원 관계이다.
(d × 3) % 352 = 1
d = 235 → **주어진 문제에서는 개인키 d를 구할 필요가 없음**

⑥ 최종적으로 생성된 키 관계는 다음과 같다.
공개키 (e, n) = (3, 391)
개인키 (d, n) = (235, 391)

⑦ 평문(메시지) m=8일 때, 공개키 (e, n) = (3, 391)을 이용하여 암호화 한다.

c = m^e mod n = 8^3 mod 391 = 512 mod 391 = **121** → c는 암호화된 메시지

정답 : ①

8. 정수의 소인수분해를 기반으로 한 RSA 암호에서 공개키 (e, n) = (7, 33)을 이용하여 생성된 암호문 C 값이 7일 때, 이를 다시 복호화한다면 원문 메시지 값은? [2017년 국가 7급]

① 11 ② 13 ③ 17 ④ 19

♤ RSA 암호 알고리즘 - 해킹 문제

// 문제 분석

- 먼저, 이 문제는 해킹 문제이다. 해서, 풀이 방법은 여러 가지가 될 수 있다.
- RSA 암호 알고리즘에서 개인키를 모르는데 어떻게 암호문을 해독할 수 있는가?
- c = m^e mod n 이므로
- 7 = m^7 mod 33 → 여기서, 원문 메시지 m을 찾는 문제이다.(m = 13)

// 답을 찾는 방법(풀이 방법)

- m = 13이므로
- 7 = 13^7 mod 33을 증명하면 된다.

// 풀이 1 : 단순한 지수 계산

- 13^7 = 62748517 = 7 mod 33 → 원문 메시지 m이 13이라는 것을 증명
  ↳ 이 방식은 13^7을 구하는데 시간이 많이 소모되므로 짧은 시간에는 불가능

// 풀이 2 : 모듈러 연산에서 지수 연산은 곱셈의 반복으로 간단하게 계산할 수 있다.

- c = 13^7 mod 33 → 계산 결과가 **c = 7이라는 것을 증명**하면 된다.
  ↓**곱셈 반복 원리 이용**
- 13^7 mod 33 = **?**
- 13^2 = (13 × 13) mod 33 = 169 mod 33 = **4**
- 13^4 = (13^2 × 13^2) mod 33 = (4 × 4) mod 33 = 16 mod 33 = **16**
- 13^7 = (13 × 13^2 × 13^4) mod 33 = (13 × 4 × 16) mod 33 = 832 mod 33 = **7**
  → 계산 결과에서 **c = 7**이라는 것은 원문 메시지 **m이 13이라는 것을 증명**

// 풀이 3 : RSA 암호시스템 응용 - 만약, d 값이 크면 풀이 2를 알아야 함

- p=3, q=11이면, n = p × q = 3 × 11 = **33**
- Phi(n) = (p−1)(q−1) = (3−1)(11−1) = 2 × 10 = **20**
- (e × d) mod 20 = (7 × d) mod 20 = 1
  ↓**d=3**이면
- (e × d) mod 20 = (7 × 3) mod 20 = 21 mod 20 = **1**
- **m** = c^d mod n = 7^3 mod 33 = 343 mod 33 = **13** → 메시지 복호화,

---

정답 : ②

9. RSA 암호시스템에서 밥이 앨리스의 공개키 (e, N)= (11, 143)을 취득하여 앨리스에게 평문 4를 암호화하여 보내고자 한다. 이때 전송되는 암호문은? [2019년 국회 9급]

① $11^4 \bmod 143$ ② $4^{11} \bmod 143$
③ $4^{143} \bmod 11$ ④ $11^{143} \bmod 7$
⑤ $4^{143} \bmod 7$

♤ RSA 암호시스템

• 암호화(수신자의 공개키 11을 이용) : $C = m^e \bmod n = 4^{11} \bmod 143$

정답 : ②

10. RSA 공개키 암호에서 2개의 소수 p=3와 q=7가 주어지고 복호화키(d) 5로 고정했을 때, 암호화 키(e) 값과 메시지 2에 대한 암호문 값(C)은? [2022년 군무원 9급]

① e = 3, C = 5 ② e = 4, C = 7
③ e = 5, C =11 ④ e = 6, C =15

♤ RSA 암호

---

① 먼저, 서로 다른 임의의 두 소수를 선택한다. p = **3**, q = **7**
② n = p × q를 구한다. n = p × q = 3 × 7 = **21**
③ n의 오일러 파이 함수 $\Phi(n)$ = (p−1)(q−1)을 구한다.
$\Phi(n) = \Phi(21)$ = (p−1)(q−1) = (3−1)(7−1) = **12**
④ d와 e를 곱하여 $\Phi(n)$으로 나눈 **나머지가 1**이 되는 **e**를 구한다.
(d × e) mod $\Phi(n)$ = **1**에서 **d** = **5**이므로
(**5** × e) mod 12 = **1**
e = **5**
⑤ 최종적으로 생성된 키 관계는 다음과 같다.**(공개키와 개인키가 같음)**
공개키 **(e, n) = (5, 21)**
개인키 **(d, n) = (5, 21)**
⑥ 메시지 **m=2**에 대한 **암호문 c = ?**
c = m^e mod n = 2^5 mod 21 = **32** mod 21 = **11**

정답 : ③

11. 공개키 암호 알고리즘인 RSA에 대한 설명으로 가장 옳지 않은 것은? (단, N은 서로 다른 두 개의 소수를 곱한 값이다) [2022년 서울 7급]

① 공개키 암호 알고리즘은 암호화 키와 복호화 키를 사용하며, RSA는 디지털 서명에 사용할 수 있다.

② RSA의 암호문은 평문을 E제곱해서 mod N을 취하여 만들 수 있으며 여기서 E와 N으로 이뤄진 쌍이 공개키다.

③ RSA의 평문은 암호문을 D제곱해서 mod N을 취하여 만들 수 있으며, 여기서 D와 N으로 이뤄진 쌍이 개인키다.

④ 큰 수의 소인수분해를 고속으로 할 수 있는 방법이 발견되면 RSA를 해독할 수 있으며, 중간자 공격을 사용해도 RSA를 해독할 수 있다.

♤ RSA

• 큰 수의 소인수분해를 고속으로 할 수 있는 방법이 발견되면 RSA를 해독할 수 있으며, **중간자 공격**을 사용해도 RSA를 **해독**할 수 있다.**(×)**

→ RSA 암호에서 **중간자 공격**은 **평문**을 직접 **해독**하는 것이 **아니고**

→ 송수신자 사이에 끼어들어 송신자에게는 수신자인 것처럼

→ 수신자에게는 송신자인 것처럼 **위장**하는 것이다.

정답 : ④