**18. Write a high-level code for Bob using the RSA cryptosystem with a very large modulus n for which the factorization cannot be found in a reasonable amount of time. Suppose Alice sends a message to Bob by representing each alphabetic character as an integer between 0 and 25 (A S 0, c, Z S 25) and then encrypting each number separately using RSA with large e and large n. Is this method secure? If not, describe the most efficient attack against this encryption method.**

**CODE :**

**import random**

**# function to check if a number is prime**

**def is\_prime(num):**

**if num == 2:**

**return True**

**if num < 2 or num % 2 == 0:**

**return False**

**for n in range(3, int(num \*\* 0.5) + 2, 2):**

**if num % n == 0:**

**return False**

**return True**

**# function to compute the gcd of two numbers**

**def gcd(a, b):**

**while b != 0:**

**a, b = b, a % b**

**return a**

**# function to generate the public and private keys**

**def generate\_keys():**

**# generate two large prime numbers**

**p = random.randint(1000, 10000)**

**while not is\_prime(p):**

**p += 1**

**q = random.randint(1000, 10000)**

**while not is\_prime(q) or q == p:**

**q += 1**

**# compute n and phi(n)**

**n = p \* q**

**phi\_n = (p-1) \* (q-1)**

**# choose a large random exponent e that is relatively prime to phi(n)**

**e = random.randint(10000, 100000)**

**while gcd(e, phi\_n) != 1:**

**e += 1**

**# compute the modular multiplicative inverse of e modulo phi(n)**

**d = pow(e, -1, phi\_n)**

**return (e, n), (d, n)**

**# function to encrypt a message**

**def encrypt(public\_key, message):**

**e, n = public\_key**

**cipher = []**

**for m in message:**

**c = pow(ord(m)-65, e, n)**

**cipher.append(c)**

**return cipher**

**# function to decrypt a message**

**def decrypt(private\_key, cipher):**

**d, n = private\_key**

**message = ''**

**for c in cipher:**

**m = pow(c, d, n)**

**message += chr(m+65)**

**return message**

**# example usage**

**public\_key, private\_key = generate\_keys()**

**print("Public key:", public\_key)**

**print("Private key:", private\_key)**

**message = "HELLO WORLD"**

**cipher = encrypt(public\_key, message)**

**print("Encrypted message:", cipher)**

**decrypted\_message = decrypt(private\_key, cipher)**

**print("Decrypted message:", decrypted\_message)**

**OUTPUT :**

