```
#include <iostream>
#include <string>
#include <cctype> // Include for isalpha()

class Cipher {
public:
   virtual std::string encrypt(const std::string& data) = 0;
   virtual std::string decrypt(const std::string& data) = 0;
   virtual ~Cipher() {}
};

class CaesarCipher : public Cipher {
private:
   int key;
public:
   CaesarCipher(int k);
   std::string encrypt(const std::string& data) override;
   std::string decrypt(const std::string& data) override;
};

CaesarCipher::CaesarCipher(int k) : key(k) {}

std::string CaesarCipher::encrypt(const std::string& data) {
   std::string result;
   for (char c : data) {
      if (isalpha(c)) {
         char base = isupper(c) ? 'A' : 'a';
         result += (c - base + key) % 26 + base;
```

```
        } else {
            result += c;
        }
    }
    return result;
}

std::string CaesarCipher::decrypt(const std::string& data) {
    std::string result;
    for (char c : data) {
        if (isalpha(c)) {
            char base = isupper(c) ? 'A' : 'a';
            result += (c - base - key + 26) % 26 + base;
        } else {
            result += c;
        }
    }
    return result;
}

// XORCipher definition (example implementation)
class XORCipher : public Cipher {
private:
    char key;
public:
    XORCipher(char k) : key(k) {}

    std::string encrypt(const std::string& data) override {
```

```
        std::string result = data;
        for (char& c : result) {
            c ^= key;
        }
        return result;
    }

    std::string decrypt(const std::string& data) override {
        return encrypt(data); // XOR encryption and decryption are
symmetric
    }
};

int main() {
    std::string message = "Hello Inheritance World!";
    CaesarCipher caesar(3);
    XORCipher xorCipher('K');  // XOR key

    std::string encryptedCaesar = caesar.encrypt(message);
    std::string decryptedCaesar = caesar.decrypt(encryptedCaesar);

    std::string encryptedXOR = xorCipher.encrypt(message);
    std::string decryptedXOR = xorCipher.decrypt(encryptedXOR);

    std::cout << "Original:    " << message << std::endl;
    std::cout << "Encrypted (Caesar): " << encryptedCaesar <<
std::endl;
    std::cout << "Decrypted (Caesar): " << decryptedCaesar <<
```

```
std::endl;
    std::cout << "Encrypted (XOR): " << encryptedXOR << std::endl;
    std::cout << "Decrypted (XOR): " << decryptedXOR << std::endl;

    return 0;
}
```