第一題

結果：

```
E(Encryption) or D(Decryption): E
Key: 3
Plain Text: There is a zebra
Cipher Text: Wkhuh lv d cheud
```
加密

```
E(Encryption) or D(Decryption): D
Key: 5
Cipher Text: N qnpj fuuqjx
Plain_Text: I like apples
```
解密

```
E(Encryption) or D(Decryption): H
No such instructions. Please enter again.

E(Encryption) or D(Decryption):
```
錯誤指令，重新輸入

程式碼：

main：

```
int main()
{
    while (1)
    {
        cout << "E(Encryption) or D(Decryption): ";
        char operation;
        cin >> operation;

        if (operation == 'E')
        {
            encryption();
            break;
        }
        else if (operation == 'D')
        {
            decryption();
            break;
        }
        else
        {
            cout << "No such instructions. Please enter again." << endl
                 << endl;
        }
    }

    return 0;
}
```

讓使用者決定要加密 or 解密

若現在要加密⋯→進入加密演算法

若現在要解密⋯→進入解密演算法

encryption( )：加密演算法

```
void encryption()
{

    cout << "Key: ";
    cin >> key;

    cout << "Plain Text: ";
    cin.ignore(1024, '\n');
    cin.getline(P, 300);

    for (int i = 0; i < strlen(P); i++)
    {
        if (isalpha(P[i]))
        {
            /* c = E(p) = (p+k) mod 26 */
            int p;
            if (isupper(P[i]))
            {
                p = P[i] - 64; //turn ascii into 1-26
                C[i] = ((char)(p + key) % 26) + 64;
            }
            else //islower(P[i])
            {
                p = P[i] - 96; //turn ascii into 1-26
                C[i] = ((char)(p + key) % 26) + 96;
            }
        }
        else //is not alphabet
        {
            C[i] = P[i];
        }
    }
    cout << "Cipher Text: " << C << endl;
}
```

使用者輸入 key、Plain Text

首先判斷字元是不是字母

再判斷大小寫

右移動 k 後

將 Cipher Text 放入 C[ ]

decryption( )：解密演算法

```
void decryption()
{
    cout << "Key: ";
    cin >> key;

    cout << "Cipher Text: ";
    cin.ignore(1024, '\n');
    cin.getline(C, 300);

    for (int i = 0; i < strlen(C); i++)
    {
        if (isalpha(C[i]))
        {
            /* p = D(c) = (c-k) mod 26 */
            int c;
            if (isupper(C[i]))
            {
                c = C[i] - 64; //turn ascii into 1-26

                int tmp = c - key;
                if (tmp < 0)
                {
                    tmp += 26;
                }
                else if (tmp == 0)
                {
                    tmp += 27;
                }
                P[i] = ((char)tmp % 26) + 64;
            }
            else //islower(C[i])
            {
                c = C[i] - 96; //turn ascii into 1-26

                int tmp = c - key;
                if (tmp < 0)
                {
                    tmp += 26;
                }
                else if (tmp == 0)
                {
                    tmp += 27;
                }
                P[i] = ((char)tmp % 26) + 96;
            }
        }
        else //is not alphabet
        {
            P[i] = C[i];
        }
    }
    cout << "Plain_Text: " << P << endl;
}
```

使用者輸入 key、Cipher Text

首先判斷字元是不是字母

再判斷大小寫

左移動 k 後

將解出的 Plain Text 放入 P[ ]

第二題

結果：

```
Cipher Text: Rfc ucyrfcp gq fmr
Plain_Text:
k=1:Qeb tbxqebo fp elq
k=2:Pda sawpdan eo dkp
k=3:Oca ravocam dn cjo
k=4:Nby qyunbyl cm bin
k=5:Max pxtmaxk bl ahm
k=6:Law owslawj ak agl
k=7:Kyv nvrkyvi aj yfk
k=8:Jxu muqjxuh yi xej
k=9:Iwt ltpiwtg xh wdi
k=10:Hvs ksohvsf wg vch
k=11:Gur jrngure vf ubg
k=12:Ftq iqmftqd ue taf
k=13:Esp hplespc td sae
k=14:Dro gokdrob sc ryd
k=15:Cqn fnjcqna rb qxc
k=16:Bpm emibpma qa pwb
k=17:Aol dlhaoly pa ova
k=18:Ank ckgankx oy nua
k=19:Ymj bjfymjw nx mty
k=20:Xli aiexliv mw lsx
k=21:Wkh ahdwkhu lv krw
k=22:Vjg ygcvjgt ku jqv
k=23:Uif xfbuifs jt ipu
k=24:The weather is hot
k=25:Sgd vdasgdq hr gns
```

k = 24 is the key

程式碼：

main：

```
int main()
{
    cout << "Cipher Text: ";
    cin.getline(C, 300);

    cout << "Plain_Text: " << endl;

    for (int k = 1; k <= 25; k++)
    {
        decryption(k);
    }
    return 0;
}
```

使用者輸入 Cipher Text

接下來讓 key 從 1 到 25 暴力破解解密演算法

decryption( )：解密演算法

```
void decryption(int key)
{
    for (int i = 0; i < strlen(C); i++)
    {
        if (isalpha(C[i]))
        {
            /* p = D(c) = (c-k) mod 26 */
            int c;
            if (isupper(C[i]))
            {
                c = C[i] - 64; //turn ascii into 1-26

                int tmp = c - key;
                if (tmp < 0)
                {
                    tmp += 26;
                }
                else if (tmp == 0)
                {
                    tmp += 27;
                }
                P[i] = ((char)tmp % 26) + 64;
            }
            else //islower(C[i])
            {
                c = C[i] - 96; //turn ascii into 1-26

                int tmp = c - key;
                if (tmp < 0)
                {
                    tmp += 26;
                }
                else if (tmp == 0)
                {
                    tmp += 27;
                }
                P[i] = ((char)tmp % 26) + 96;
            }
        }
        else //is not alphabet
        {
            P[i] = C[i];
        }
    }
    cout << "k=" << key << ":" << P << endl;
}
```

首先判斷字元是不是字母

再判斷大小寫

左移動 k 後

將解出來的 Plain Text 放入 P[ ]