

main.c



Run

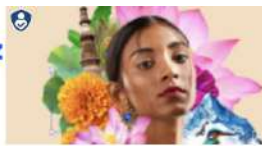
Output

Clear

```
1 #include <stdio.h>
2 #include <wchar.h>
3 #include <stdlib.h>
4 #include <locale.h>
5 #define MAX_LEN 1000
6 #define CHARSET_SIZE 128
7 void frequency_analysis(const wchar_t *text) {
8     int freq[CHARSET_SIZE] = {0};
9
10    for (int i = 0; text[i] != L'\0'; i++) {
11        freq[(unsigned char)text[i]]++;
12    }
13
14    wprintf(L"Character Frequency:\n");
15    for (int i = 0; i < CHARSET_SIZE; i++) {
16        if (freq[i] > 0) {
17            wprintf(L"'%lc' (ASCII %d): %d\n", i
                , i, freq[i]);
```

```
'.' (ASCII 46): 1
'0' (ASCII 48): 6
'1' (ASCII 49): 7
'2' (ASCII 50): 5
'3' (ASCII 51): 4
'4' (ASCII 52): 19
'5' (ASCII 53): 12
'6' (ASCII 54): 11
'8' (ASCII 56): 34
'9' (ASCII 57): 5
':' (ASCII 58): 4
';' (ASCII 59): 27
'?' (ASCII 63): 3
']' (ASCII 93): 1
Segmentation fault
```

=== Code Exited With Errors ===



main.c



Run

Output

Clear

```
19     }
20 }
21 void apply_mapping(const wchar_t *cipher, const
    wchar_t map[CHARSET_SIZE]) {
22     wprintf(L"\nPartially Decrypted Message:\n"
        );
23     for (int i = 0; cipher[i] != L'\0'; i++) {
24         wchar_t c = cipher[i];
25         if (map[(unsigned char)c] != 0) {
26             wprintf(L"%lc", map[(unsigned char
                )c]);
27         } else {
28             wprintf(L"_");
29         }
30     }
31     wprintf(L"\n");
32 }
33 int main() {
```

```
'.' (ASCII 46): 1
'0' (ASCII 48): 6
'1' (ASCII 49): 7
'2' (ASCII 50): 5
'3' (ASCII 51): 4
'4' (ASCII 52): 19
'5' (ASCII 53): 12
'6' (ASCII 54): 11
'8' (ASCII 56): 34
'9' (ASCII 57): 5
':' (ASCII 58): 4
';' (ASCII 59): 27
'?' (ASCII 63): 3
']' (ASCII 93): 1
Segmentation fault
```

```
=== Code Exited With Errors ===
```

main.c

```
31 wprintf(L"\n");
32 }
33 int main() {
34     setlocale(LC_ALL, "");
35     const wchar_t *ciphertext =
36         L"53†††305)6*;4826)4†.)4†);806*;48†8¶60
37         )85;;]8*;:‡*8†83"
38         L"(88)5*†;46(;88*96*?;8)*‡(;485);5*†2:*‡
39         (;4956*2(5*-4)8¶8*"
40         L";4069285);)6†8)4†‡;1(†9;48081;8:8†1
41         ;48†85;4)485†528806*81"
42         L"(†9;48;(88;4(†?34;48)4†;161;;:188;†?;"
43     frequency_analysis(ciphertext);
44     wchar_t map[CHARSET_SIZE] = {0};
45     map[L'8'] = L'E';
46     map[L':'] = L'T';
47     map[L'‡'] = L'H';
48     map[L'5'] = L'O';
49     ... ..
```

Run

Output

Clear

```
'.' (ASCII 46): 1
'0' (ASCII 48): 6
'1' (ASCII 49): 7
'2' (ASCII 50): 5
'3' (ASCII 51): 4
'4' (ASCII 52): 19
'5' (ASCII 53): 12
'6' (ASCII 54): 11
'8' (ASCII 56): 34
'9' (ASCII 57): 5
': ' (ASCII 58): 4
'; ' (ASCII 59): 27
'? ' (ASCII 63): 3
']' (ASCII 93): 1
Segmentation fault
```

=== Code Exited With Errors ===

main.c		Run	Output	Clear
48	map[L'+'] = L'S';		'.' (ASCII 46): 1	
49	map[L'0'] = L'A';		'0' (ASCII 48): 6	
50	map[L'6'] = L'D';		'1' (ASCII 49): 7	
51	map[L'] = L'L';		'2' (ASCII 50): 5	
52	map[L'*'] = L'I';		'3' (ASCII 51): 4	
53	map[L';'] = L' ';		'4' (ASCII 52): 19	
54	map[L'2'] = L'C';		'5' (ASCII 53): 12	
55	map[L'('] = L'M';		'6' (ASCII 54): 11	
56	map[L'9'] = L'Y';		'8' (ASCII 56): 34	
57	map[L'1'] = L'B';		'9' (ASCII 57): 5	
58	map[L'?'] = L'U';		':' (ASCII 58): 4	
59	map[L'.'] = L'G';		',' (ASCII 59): 27	
60	map[L'-'] = L'P';		'?' (ASCII 63): 3	
61	map[L'] = L'W';		']' (ASCII 93): 1	
62	map[L'@'] = L'F';		Segmentation fault	
63	apply_mapping(ciphertext, map);			
64				
65	return 0;			
--			=== Code Exited With Errors ===	