main.c

Run

Output

Clear

```c
#include <stdio.h>
#include <wchar.h>
#include <stdlib.h>
#include <locale.h>
#define MAX_LEN 1000
#define CHARSET_SIZE 128
void frequency_analysis(const wchar_t *text) {
    int freq[CHARSET_SIZE] = {0};

    for (int i = 0; text[i] != L'\0'; i++) {
        freq[(unsigned char)text[i]]++;
    }

    wprintf(L"Character Frequency:\n");
    for (int i = 0; i < CHARSET_SIZE; i++) {
        if (freq[i] > 0) {
            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

```
    }
}
void apply_mapping(const wchar_t *cipher, const
    wchar_t map[CHARSET_SIZE]) {
    wprintf(L"\nPartially Decrypted Message:\n"
        );
    for (int i = 0; cipher[i] != L'\0'; i++) {
        wchar_t c = cipher[i];
        if (map[(unsigned char)c] != 0) {
            wprintf(L"%lc", map[(unsigned char
                )c]);
        } else {
            wprintf(L"_");
        }
    }
    wprintf(L"\n");
}
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

Run

```
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
                ))85;;]8*;:‡*8†83"
37          L"(88)5*†;46(;88*96*?;8)*‡(;485);5*†2:*‡
                (;4956*2(5*—4)8¶8*"
38          L";4069285);)6†8)4‡‡;1(‡9;48081;8:8‡1
                ;48†85;4)485†528806*81"
39          L"(‡9;48;(88;4(‡?34;48)4‡;161;:188;‡?;";
40      frequency_analysis(ciphertext);
41      wchar_t map[CHARSET_SIZE] = {0};
42      map[L'8'] = L'E';
43      map[L':'] = L'T';
44      map[L'‡'] = L'H';
45      map[L'5'] = L'O';
```

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

```
48      map[L'†'] = L'S';
49      map[L'0'] = L'A';
50      map[L'6'] = L'D';
51      map[L')'] = L'L';
52      map[L'*'] = L'I';
53      map[L';'] = L' ';
54      map[L'2'] = L'C';
55      map[L'('] = L'M';
56      map[L'9'] = L'Y';
57      map[L'1'] = L'B';
58      map[L'?'] = L'U';
59      map[L'.'] = L'G';
60      map[L'—'] = L'P';
61      map[L']'] = L'W';
62      map[L'¶'] = L'F';
63      apply_mapping(ciphertext, map);
64
65      return 0;
```

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 ===
```