

```
[jpolitz@ieng6-203]:10-03-w1f-inspect-utf8:498$ gcc inspect.c -o inspect
[jpolitz@ieng6-203]:10-03-w1f-inspect-utf8:499$ ./inspect
Hello! (length 6): H 72 e 101 l 108 l 108 o 111 ! 33
[jpolitz@ieng6-203]:10-03-w1f-inspect-utf8:500$
```

%s  
strings

%ld  
prints  
"long decimal" size\_t

```
<.edu:/home/linux/ieng6/CSE29_FA25_A00/public/lecture/10-03-w1f-inspect-utf8] 1
```

```
#include <stdio.h>
#include <string.h>
```

```
// Write a function called inspect that prints a lot of information about a C
// string
```

```
void inspect(char str[]) {
    printf("%s (length %ld): ", str, strlen(str));
    int i = 0;
    while(str[i] != 0) {
        printf("%c %hhu ", str[i], str[i]);
        i += 1;
    }
    printf("\n");
}
```

```
int main() {
    inspect("Hello!");
}
```

```
[jpolitz@ieng6-203]:10-03-w1f-inspect-utf8:505$ gcc inspect.c -o inspect
[jpolitz@ieng6-203]:10-03-w1f-inspect-utf8:506$ ./inspect
Hello! (length 6): H 72 e 101 l 108 l 108 o 111 ! 33
Kiên (length 5): K 75 i 105 195 170 n 110
헌트릭스 (length 12): 237 151 140 237 138 184 235 166 173 236 138
🚀 (length 4): 240 159 154 128
```

2 bytes for `ë`

3 bytes for `리`

4 bytes for `🚀`

```
<.edu:/home/linux/ieng6/CSE29_FA25_A00/public/lecture/10-03-w1f-inspect-utf8] 1,1
#include <stdio.h>
#include <string.h>
```

```
// Write a function called inspect that prints a lot of information about a C
// string
void inspect(char str[]) {
    printf("%s (length %ld): ", str, strlen(str));
    int i = 0;
    while(str[i] != 0) {
        printf("%c %hhu ", str[i], str[i]);
        i += 1;
    }
    printf("\n");
}

int main() {
    inspect("Hello!");
    inspect("Kiên");
    inspect("헌트릭스");
    inspect("🚀");
}
```

Kiên (length 5):

K 75 1001011  
i 105 1101001  
195 11000011  
170 10101010  
n 110 1101110

2 byte symbol

starts w 110

starts 10

헌트릭스 (length 12):

237 11101101  
151 10010111  
140 10001100  
237 11101101  
138 10001010  
184 10111000  
235 11101011  
166 10100110  
173 10101101  
236 11101100  
138 10001010  
164 10100100

start 10

10

3 byte symbols

starts w 1110

🚀 (length 4):

240 11110000  
159 10011111  
154 10011010  
128 10000000

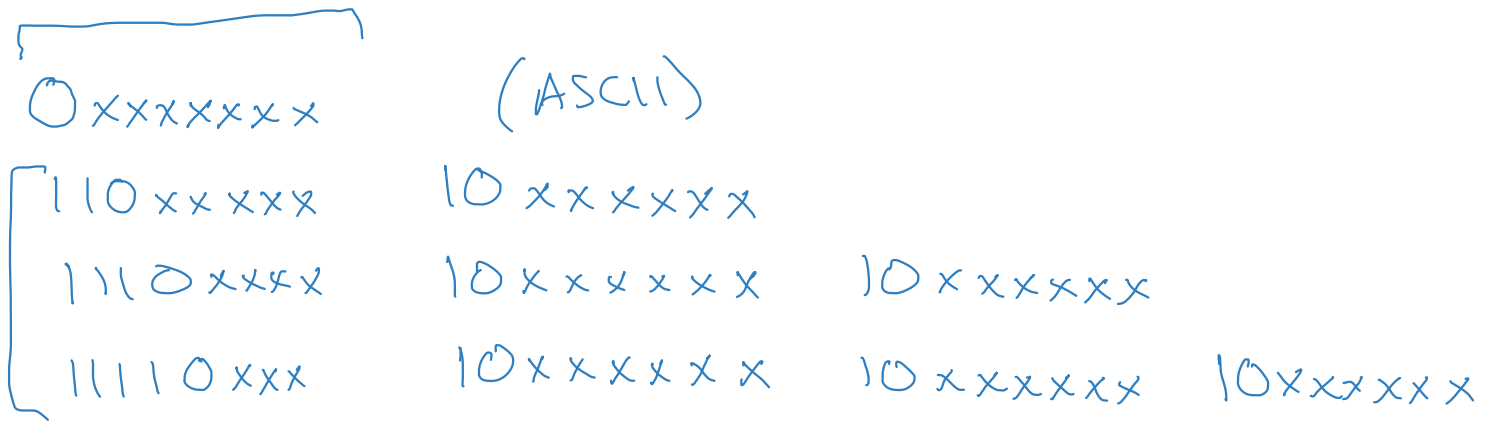
4 byte symbol

starts w 11110

starts 10

# UTF8 encoding of strings

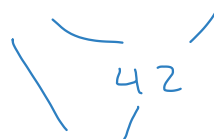
— backwards-compatible w/ASCII



<sup>^</sup>  
e    11000011    10101010

00011101010

128 64 32    8    2



106

<sup>1</sup>  
234 = "code point"