UTF-8 Encoding Algorithm

Determine the character you want to encode, e.g., the 'REVERSED SEMICOLON'

1. determine its UTF-8 index, e.g., U+204F
2. lookup information about this character
    a. UTF-8 index: index= 0x 204F
    b. UTF-8 hex encoding: encoding= 0x e2818f

Algorithm to encode the UTF-8 index → UTF-8 encoding

1. Use gdb to obtain the binary representation of UTF-8 index
    (gdb) print /t 0x204F
    $3 = 10000001001111

2. If the number of significant digits <= 7, lookup the index in the ASCII table
    Otherwise: separate the binary bit pattern in groups of 6, from right to left
    10 000001 001111 (original)
    a. 10
    b. 000001
    c. 001111

3. Identify the number of bytes needed to encode the sequence, e.g., 3

4. Select the appropriate row within the 'Layout of UTF-8 byte sequence' table

Number of bytes	First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
3	U+0800	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	

5. Add the extra encoding bits (see Step #4) to each of your bit patterns from Step #2
    a. Byte 1: 10 → 1110 xx10
    b. Byte 2: 000001 → 10 000001
    c. Byte 3: 001111 → 10 001111

6. Concatenate the encoded bytes
    a. 1110 xx10 1000 0001 1000 1111

7. Replace any remaining x bits in byte 1 with zeros.
    a. 1110 0010 1000 0001 1000 1111

    This value is the binary encoding of the UTF-8 encoding: 0x e2818f

Double check your work, by obtinain the binary representation of the UTF-8 encoding
    (gdb) print /t 0xe2818f
    $4 = 11100010100000010001111

Compare the binary pattern with the value you obtained in Step 7.

Layout of UTF-8 byte sequences

Number of bytes	First code point	Last code point	Byte 1	Byte 2	Byte 3	Byte 4
1	U+0000	U+007F	0xxxxxxx			
2	U+0080	U+07FF	110xxxxx	10xxxxxx		
3	U+0800	U+FFFF	1110xxxx	10xxxxxx	10xxxxxx	
4	U+10000	[nb 3]U+10FFFF	11110xxx	10xxxxxx	10xxxxxx	10xxxxxx