CS 240
Computer Systems

#2: Hexadecimal, UTF8, and C Programming

Aug. 26, 2021 · Wade Fagen-Ulmschneider

Representing Data: Hexadecimal

Binary data gets really long, really fast! The number of students enrolled at University of Illinois is **`0b1100 1100 0110 1011`**

- To represent binary data in a compact way, we often will use **hexadecimal** -- or "base-16" -- denoted by the prefix **`0x`**.

Hexadecimal Digits:

Place of Hexadecimal Numbers:

Hex Number:	`c`	`0`	`f`	`f`	`e`	`e`
Place Value:	16^5	16^4	16^3	16^2	16^1	16^0
Decimal Place Value:	1048576	65536	4096	256	16	1
SUM:						

Translation from Decimal to Hexadecimal:

11_{10} = 0x

34_{10} = 0x

87_{10} = 0x

255_{10} = 0x

Hexadecimal is particularly useful as it _____:

University of Illinois student population in Fall 2020 (52,331):				
`0b`	`1100`	`1100`	`0110`	`1011`
`0x`				

Number of people following Taylor Swift on Twitter (88,681,056):	
`0b`	`101 0100 1001 0010 1010 0110 0000`
`0x`	

Representing Letters: ASCII

Representing numbers is great -- but what about words? Can we make sentences with binary data?

- **Key Idea:** Every letter is _____ binary bits.*
  (This means that every letter is _____ hex digits.)
- Global standard called the **American Standard Code for Information Interchange (ASCII)** is a _____ _____ for translating numbers to characters.
- ASCII was not the first but was developed from a long history of other encodings. *Charles MacKenzie's "Coded Character Set, History, and Development" has an over 500-page history on character encodings! (Linked on the website!)*

ASCII Character Encoding Examples:					
Binary	**Hex**	**Char.**	**Binary**	**Hex**	**Char.**
`0b 0100 0001`	`0x41`	`A`	`0b 0110 0001`	`0x61`	`a`
`0b 0100 0010`	`0x42`	`B`	`0b 0110 0010`	`0x62`	`b`
		`C`			`c`
		`D`			`d`
`0b0010 0100`	`0x24`	`$`	`0b0111 1011`	`0x7b`	`{`

...and now we can form sentences!

Q: Are there going to be any issues with ASCII?

Representing Letters: Other Character Encodings

Since ASCII uses only 8 bits, we are limited to only 256 unique characters. There's far more than 256 characters -- and what about EMOJIs?? 🎉

- **Many** other character encodings exist other than ASCII.
- The most widely used character encoding is known as **Unicode Transformation Format (8-bit)** or _____.
- Standard is **ISO/IEC 10646** (Latest update is :2002, or v13).

UTF-8 uses a _____-bit design where each character by be any of the following:

Length	Byte #1	Byte #2	Byte #3	Byte #4
1-byte	`0___ ____`			
2-bytes:	`110_ ____`	`10__ ____`		
3-bytes:	`1110 ____`	`10__ ____`	`10__ ____`	
4-bytes:	`1111 0___`	`10__ ____`	`10__ ____`	`10__ ____`

Unicode characters are represented by **U+##** *(where ## is the hex value of the character encoding data)* and all 1-byte characters match the ASCII character encoding:

- `'a'` is ASCII `______`, or `________`.

Example: ε (epsilon) is defined as **`U+03b5`**. How do we encode this?

Example: I received the following binary message encoded in UTF-8:

`0100 1000 0110 1001 1111 0000 1001 1111 1000 1110 1000 1001`

1. What is the hexadecimal representation of this message?

2. What is the **byte length** of this message? _____
3. What is the **character length** of this message? _____

4. What does the message say?

Programming in C
One example of a plain-text file in a C source code file. Today, you'll see your very first Machine Problem in CS 240!

- You already know how to program in C++! 🎉
- Programming in C is a simplification of the C++ programming.

1. Program Starting Point of **ALL** C PROGRAMS:

2. Printing Using `printf()` (from `<stdio.h>`):

```
1  #include <stdio.h>
2
3  int main() {
4    int i = 42;
5    char *s = "Hello, world!";
6    float f = 3.14;
7
8    printf("%d %s %f\n", i, s, f);
9    printf("%d\n", s[0]);
10   printf("%f\n", s[0]);
11   printf("%d\n", s);
12   return 0;
13 }
```

`printf` has a variable number of arguments:

First argument

Additional arguments

3. Pointers:

4. Heap Memory Allocation:

```
1  int main() {
2    char *s = malloc(10);
3    int *num = malloc( sizeof(int) );
4
5    printf("%p %p\n", s, num);
6    return 0;
7  }
```

5. Strings -- `#include <string.h>`

Four Key Functions:

- `strcmp(char *s1, char *s2)` -- Compares two strings
- `strcat(char *dest, char *src)` -- Concatenate two strings
- `strcpy(char *dest, char *src)` -- Copies a string
- `strlen(char *s)` -- Returns the length of the string