Q1: What is the decimal value of the 8-bit binary number 0000 1101 ? 8+4+1 = 13

8 4 2 1

Q2: What is 37 as an 8-bit binary number? 128 64 32 16 8 4 2 1

0 0 1 0 0 1 0 1

Q3: printf(" %c %d ", 70, 70);

Fill in the blanks to make this print F 70

Exams

Weeks 4, 8, 10

Testing Center

Week 10 Lab

Make-ups

Finals Week

prairietest.com

Practice Exam

printf("<format string>", val1, val2, ...)

Prints the format string with any format specifiers replaced by the corresponding formatted value

%c - characters
%d - decimal values (signed)
%u - decimal values (unsigned)
%s - strings

```
printf("%s has %d characters", "Hello", 5);
```

Hello has 5 characters.

To put '%' in the printed string, use \%

A C string is a sequence of char (1 byte each) followed by a 0 byte called a null terminator

Often created as an array

```
char msg[] = "Hello!";
char msg2[] = {'H', 'e', 'l', 'l', 'o', '!', 0};
```

Stored as

'H'	'e'	'l'	'l'	'o'	'!'	'\0'	] convenient human-readable views
72	101	108	108	111	33	0	
			... bits on/off ...				] truth

msg[0] evaluate to 'H'

msg[3] evaluate to 'l'

msg[1] = 'a' changes the 'e' to 'a'

changes index 1 to store 'a' instead of 'e'

There is no length field for C arrays.

C strings define strlen which gives the length of a C string.

msg = "abc" (assuming msg defined as above)

X an error!

```c
#include <stdio.h>

void uppercase(char str[]) {









}

void inspect(char str[]) {
    for (int i = 0; str[i] != '\0'; i++) {
        char c = str[i];
        printf("(%c %d) ", c, c);
    }
    printf("\n");
}

int main() {
    char message[] = "Hello!";
    printf("%s\n", message);
    printf("(%c %d) (%c %d) (%c %d) (%c %d) (%c %d) (%c %d) (%c %d)\n",
           message[0], message[0],
           message[1], message[1],
           message[2], message[2],
           message[3], message[3],
           message[4], message[4],
           message[5], message[5],
           message[6], message[6]);
    inspect(message);

    // After this call, message should contain "HELLO!"
    uppercase(message);
    printf("After uppercase: %s\n", message);
}
```

inspect doesn't print the terminator

'a' 'z' 'A' 'Z'
97 122 65 90

```
$ gcc hello.c -o hello
$ ./hello
Hello!
(H 72) (e 101) (l 108) (l 108) (o 111) (! 33) ( 0)
(H 72) (e 101) (l 108) (l 108) (o 111) (! 33)
After uppercase: HELLO!
```