FTP storage covert channel tutorial-ish

suppose that each file on the FTP server represents a single ASCII character
    by this we mean the permissions of the file: `drwxrwxrwx`
    this works nicely because a permission "bit" can either be on (d, l, rwx) or off (-)
let's further restrict that to the basic ASCII character set (i.e., 0-127), which requires 7 bits
    the permissions of a file is made up of 10 characters
    we can effectively throw away 3 of them: `---xrwxrwx`
to add noise, we can add files that have one or more of the first three permission "bits" set
    but we ignore them on the receiving side
    problem is, this wastes space (i.e., we could use these files, plus we could use the extra 3 bits)
        we'll deal with this later
let's try the message: **Tarot**
    first, we break it down into its ASCII representation:

```
T=84
a=97
r=114
o=111
t=116
```

    next, we convert to 7-bit binary:

```
T=1010100
a=1100001
r=1110010
o=1101111
t=1110100
```

    file permissions are made up of three categories (user, group, other), each made up of 3 bits
        we need to prepend the bits with two 0s (to end up with 9 total bits)
        then we split them into 3 groups (or octet)

```
T=001 010 100
a=001 100 001
r=001 110 010
o=001 101 111
t=001 110 100
```

        next, we need to convert each octet to decimal to obtain the permission values

```
T=124
a=141
r=162
o=157
t=164
```

    we can now create random files, sort them, and apply the permissions in sorted order
    e.g., (in order):

```
touch file1
chmod 124 file1
touch file2
chmod 141 file2
touch file3
chmod 162 file3
touch file4
chmod 157 file4
touch file5
chmod 164 file5
```

the result is something like this:

```
---x-w-r-- 1 at at 0 Mar 29 9:00 file1*
---xr----x 1 at at 0 Mar 29 9:00 file2*
---xrw--w- 1 at at 0 Mar 29 9:00 file3*
---xr-xrwx 1 at at 0 Mar 29 9:00 file4*
---xrw-r-- 1 at at 0 Mar 29 9:00 file5*
```

adding noise means adding files with some of the first three bits set; e.g.,:

```
---x-w-r-- 1 at at 0 Mar 29 9:00 file1*
d--xrw-r-- 1 at at 0 Mar 29 9:00 file1.5*
---xr----x 1 at at 0 Mar 29 9:00 file2*
---xrw--w- 1 at at 0 Mar 29 9:00 file3*
-r-xrwxrwx 1 at at 0 Mar 29 9:00 file3.5*
---xr-xrwx 1 at at 0 Mar 29 9:00 file4*
---xrw-r-- 1 at at 0 Mar 29 9:00 file5*
-rwx--xr-x 1 at at 0 Mar 29 9:00 file5.5*
```

receiving is just the reverse

```
---x-w-r-- 1 at at 0 Mar 29 9:00 file1*
0001010100=84=T
d--x---r-- 1 at at 0 Mar 29 9:00 file1.5*
1001000100=ignored
---xr----x 1 at at 0 Mar 29 9:00 file2*
0001100001=97=a
```

...and so on...

what about using all permission “bits” and not wasting space?

- no more noise files (i.e., all files are meaningful)
- let's use them all in the same manner (on or off)
- 10 bits per file/directory
- order alphabetically, decode, and concatenate all the bits

to create the message, its bits must first be divisible by 10

- if not, either add extra “fluff” characters to the message to ensure this
- or append the bits with 0s and ignore those when decoding

when decoding, bits must be split up in groups of 7 (since we are using basic ASCII)

- extended ASCII is not really workable at the command line
  - many characters are not printable
  - although so are characters with ASCII values 0-31...

try to decode the following:

```
d---r--rwx 2 at at 4K Mar 29 9:15 0fd1b45f22e18b3
-r-xrw--w- 1 at at  0 Mar 29 9:15 17c455d90e49
-rw--w-r-x 1 at at  0 Mar 29 9:15 302289542768697c
-rw---x--- 1 at at  0 Mar 29 9:15 4bdf419390d83b860cec
--wxr-xrwx 1 at at  0 Mar 29 9:15 51451ddb647ff3566601f232
d-w---xr-- 2 at at 4K Mar 29 9:15 6e8dd5f0924ce30b35aeaed9
d-wxrw--w- 2 at at 4K Mar 29 9:15 70a8cbb30
dr--r-x-w- 2 at at 4K Mar 29 9:15 79bf30d265cbd436079e
-rwxrwx--x 1 at at  0 Mar 29 9:15 81052541de641ff1ed7ca40
d-w-----wx 2 at at 4K Mar 29 9:15 a8b18ffb171e161c753ab8d
-rw-rwxrw- 1 at at  0 Mar 29 9:15 c52eda933ff95be8f914eaf62
-r-x-----x 1 at at  0 Mar 29 9:15 daf9509999adb4f6e6b49c7e91
```

```
d---rwxr-- 2 at at 4K Mar 29 9:15 f35c8e8ed0fb8a609
--wxrw--w- 1 at at  0 Mar 29 9:15 f4ed4ab4e61c850de968
-rwxrwx-w- 1 at at  0 Mar 29 9:15 f59a77545fe6d10
---x------ 1 at at  0 Mar 29 9:15 fce47615d2
```

solution on the next page (don't look yet!)

first file:

```
d---r--rwx 2 at at 4K Mar 29 9:15 0fd1b45f22e18b3
```

decodes to:

```
1000100111
```

second file:

```
-r-xrw--w- 1 at at  0 Mar 29 9:15 17c455d90e49
```

decodes to:

```
0101110010
```

and so on...we keep decoding

```
1000100111010111001001100101010110001000001110111110100011001011
1110010110010101001111110011010000011011011111001010000011000111
1100001111001001111110100001000000
```

and now to get the message (first, split into groups of 7 bits)

1000100	1110101	1100100	1100101	0101100	0100000	1110111	1101000
D	u	d	e	,	**space**	w	h
1100101	1110010	1100101	0100111	1110011	0100000	1101101	1111001
e	r	e	'	s	**space**	m	y
0100000	1100011	1100001	1110010	0111111	0100001	000000	
space	c	a	r	?	!	**ignored**	

message: Dude, where's my car?!