CS 315-01

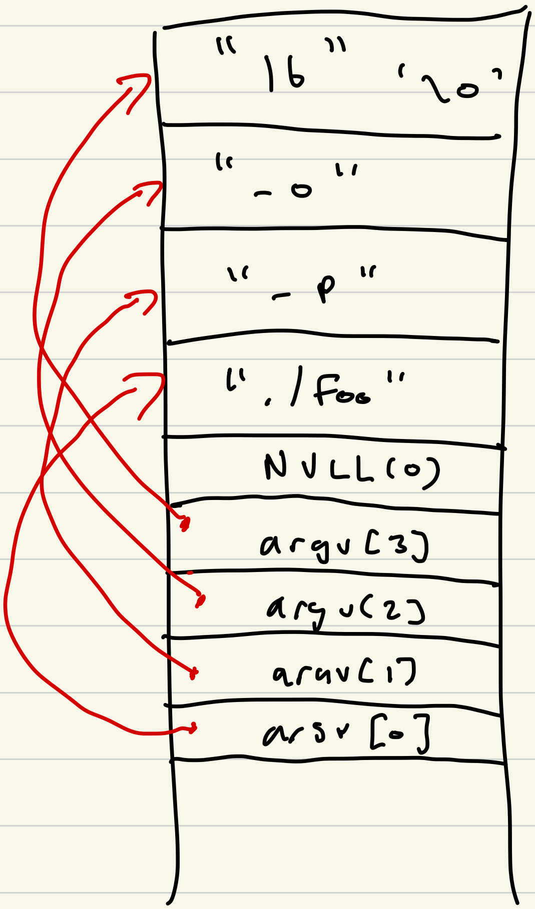
Args

Numbers

$\underbrace{\text{./foo}}\ \underbrace{\text{-p}}\ \underbrace{\text{-o}}\ \underbrace{\text{16}}$ 4 args

int argc)

char *argv[]



```
echorepeat -r <count> <str>
echorepeat -r 10 Foo
echorepeat Foo -r 5
```

Numbers

binary
↓

245 "245" $\boxed{245}$

quantity string int
machine

byte byte
'2' '4' '5'
ascii
↑
binary

Decimal (base 10)

base

245

pos

$2 \times (10^2) + 4 \times (10^1) + 5 \times (10^0)$

$2 \times 100 + 4 \times 10 + 5 \times 1$

$200 + 40 + 5 = 245$

Binary (base 2)

3 2 1 0

0b1101 $\rightarrow$ Dec 13

```
int x = 3;
int x = 0b11;
int x = 0x3;
```

$1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$8 + 4 + 0 + 1 = 13$

8 4 2 1

0b1101 $\longleftarrow$ 4 bit binary value

most significant bit

least significant bit

n-bit binary number

2^n possible values

0 to $2^n - 1$

Dec	2bit
0	00
1	01
2	10
→ 3	11

4

Dec	Bin
0	000
1	001
2	010
3	011
4	100
5	101
6	110
→ 7	111

Hexadecimal (base 16)

Dec(10)	Bin (2)	Hex(16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

0x1AF

↓

$$1 \times \underline{16}^2 + A \times \underline{16}^1 + F \times 16^0$$

$$1 \times \overline{16}^2 + 10 \times \overline{16}^1 + 15 \times \overline{16}^0$$

256 + 160 + 15 = 431

$$\begin{tikzcd}
0x & 1 & A & F \\
0b & 0001 \arrow[u] & 1010 \arrow[u] & 1111 \arrow[u]
\end{tikzcd}$$

Project01

numstr → int → numstr (base)

"245"

$\nearrow$ char ~~x~~s = "245";

s[0] = '2'
s[1] = '4'
s[2] = '5'

ASCII

'0'	= 48
'1'	= 49
'2'	= 50

int x = s[0];   printf("%d\n", x);
48

x = s[0] - 48;
x = s[0] - '0';

```
int num;

num = (s[0] - '0') * 100
      + (s[1] - '0') * 10
      + (s[2] - '0') * 1
                                  °
                                "245"
num = 245                        ↑
```

```
int    intstr_to_int (char *s) {
    int num = 0;
    int digit;
    int i = 0;


    while (s[i] != '\0')  {
        num *= 10;
        digit = s[i] - '0';
        num += digit;
        i += 1;

    }
    return num;
}
```

int to string

int x = 24$\textcircled{5}$;
int d;   ↑

d = x % 10   = $\boxed{5}$
x = x / 10   = 24

d = x % 10   = $\boxed{4}$
x = x / 10   = 2

d = x % 10   = $\boxed{2}$
x = x / 10   = $\boxed{0}$

5 + '0' = 5 + 48 = 53

int x = 245
int d   d = x % 16 =