CS 315-01 Optional Args & Base Conversion

ls -a

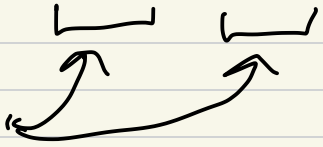
ls -a -l ls -l -a

gcc -o <Filename>

loop:

* check for each option and consume

ls -a -l



echorepeat -h -f -n <count> <string>

echorepeat -h -f foo

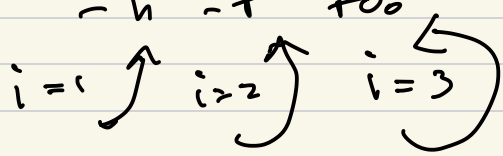
- - - - -
foo
- - - - -

echorepeat -h foo

0 1 2

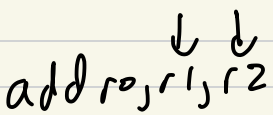
argc == 3

echorepeat -h -f foo



Numbers

245	"245"	binary 245
quantity	String	machine integer
	byte byte byte '2' '4' '5' 50 52 53	11110101



Bases

10 decimal

2 binary

16 hexadecimal

Decimal Base 10

2 1 0
2 4 5

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

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

$200 + 40 + 1$

$= 245$

Binary Base 2

3 2 1 0
0 b 1 1 0 1
8 4 2 1

```
int x = 13;
int x = 0b1101;
int x = 0xD;
```

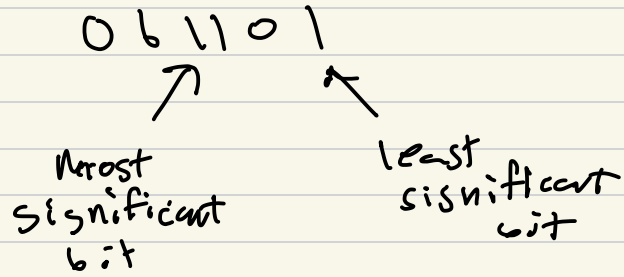
$(1 \times 2^3) + (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0)$

$1 \times 8 + 1 \times 4 + 0 \times 2 + 1 \times 1$

$8 + 4 + 0 + 1$

$= 13$

4 bit binary number



n - bit binary numbers

2^n possible values

range 0 to $(2^n)-1$

$2^4 = 16$

0 to 15

0000 1111

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	a
11	1011	B	b
12	1100	C	c
13	1101	D	d
14	1110	E	e
15	1111	F	f

2 1 0

0x2AF

$(2 \times 16^2) + (\text{'A'} \times 16^1) + (\text{'F'} \times 16^0)$

$2 \times 256 + 10 \times 16 + 15 \times 1$

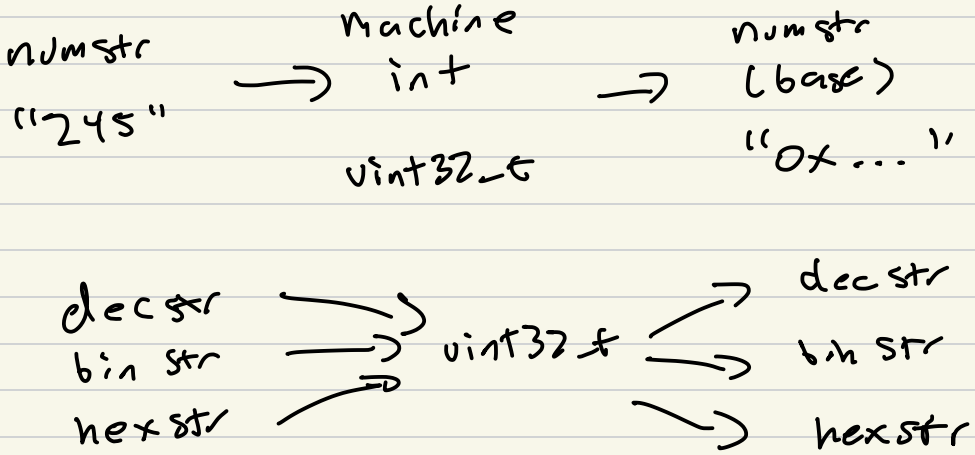
$512 + 160 + 15$

$= 687$

Project01

```
$ numconv 687 -o 16
0x2AF

$ numconv 0x2AF -o 10
687
```



input base

```
if (prefix(str) == "0b") == bin
if (prefix(str) == "0x") == hex
else dec
```

char *s = "245";

s[0] = '2'
s[1] = '4'
s[2] = '5'
s[3] = '\0'

ASCII

'0' = 48
'1' = 49
'2' = 50
'3' = 51
'4' = 52
'5' = 53

int v0 = s[0] - '0' ('2')
= 50 - 48
= 2

int v1 = s[1] - '0' ('4')
= 52 - 48
= 4

int v2 = s[2] - '0' ('5')
= 53 - 48
= 5

int V = (v0 * 100) + (v1 * 10) + (v0 * 1)
= 245

```
uint32_t decstr_to_uint32(char *s) {
    int v = 0;
    int d;
    int i = 0;

    while (s[i] != '\0') {
        v = v * 10;
        d = s[i] - '0';
        v = v + d;
        i = i + 1;
    }
```

uint32_t to numstr.

int v = 245;

int d0, d1, d2

base

245 / 10 = 24

remainder = 5

245 % 10 = 5

d0 = v % 10

= 245 % 10

= 5 ⟶ char c0 = '0' + d0

ascii

v = v / 10 24

d1 = v % 10

= 24 % 10

= 4 ⟶ c1 = '0' + d1

ascii

v = v / 10 2

d2 = v % 10

= 2 % 10

= 2 ⟶ c2 = '0' + d2

ascii