COP3502 11/14/25

Bitwise Ops

& and

| or

^ xor

exclusive

```
    A = 10110010        ans
^   B = 10001001        student
-------------------
    00111011   -> 1's incorrect ans
```

a >> 3

chops off last 3 bits

int divides by 2^3.

a = 10100110 →

→ 10100

a << 2

left shift moves bits to left

multiplying by 2^2.

a = 10100110

⟶ 10100110<u>00</u>

How to test if bit k is on?

if (n & (1<<k))

k = 3

```
n = 01011010
&   00001000
    00001000
```

exactly 1 bits

```
int numOn = 0;
for (int i=0; i<20; i++)
  if (n & (1<<i))
    numOn++;
```

Knapsack - looks at each subset of a given set.

Recovery - about grading T/F exams
^ allows us to find differences

Contractor → - variant of knapsack

Baby Sitter → 011 → item 0, item 1
↑↑
10

Knapsack Idea

n = # items → mask represents a subset of items

2^n

n = 3

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

```
for (int mask=0; mask < (1<<n); mask) {
    int cWeight=0;
    int cValue=0;
    for(int i=0; i<n; i++){        // i is item were checking
        if (mask & (1<<i)) {
            cWeight += W[i];
            cValue += V[i];
        }
    }
}
```

⟹ Check if cWeight <= capacity

If yes, see if ~~this~~ value is better than your car max.

Recovery

#questions

try each ans key

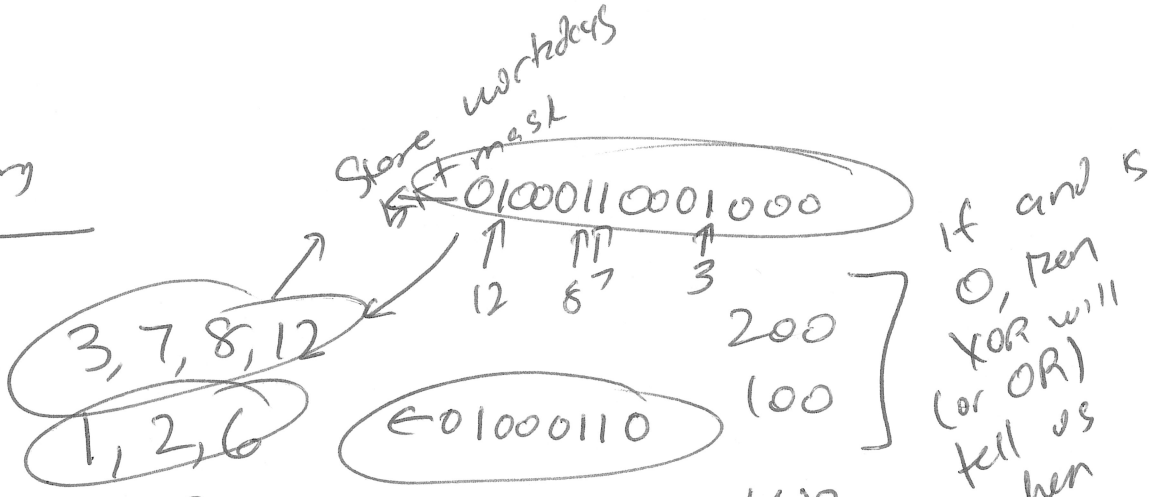
```
for (int key = 0; key < (1<<n); key++) {
    int* freq = calloc(n, sizeof(int));
    for (int i = 0; i < numStd; i++)
        freq[score(std[i], key)]++;
    if (equal(freq, corfreq))
        res++;
}
```

try each key

score exams

see if freq match

Baby sitting



✓ d = 4

✓ d = 3

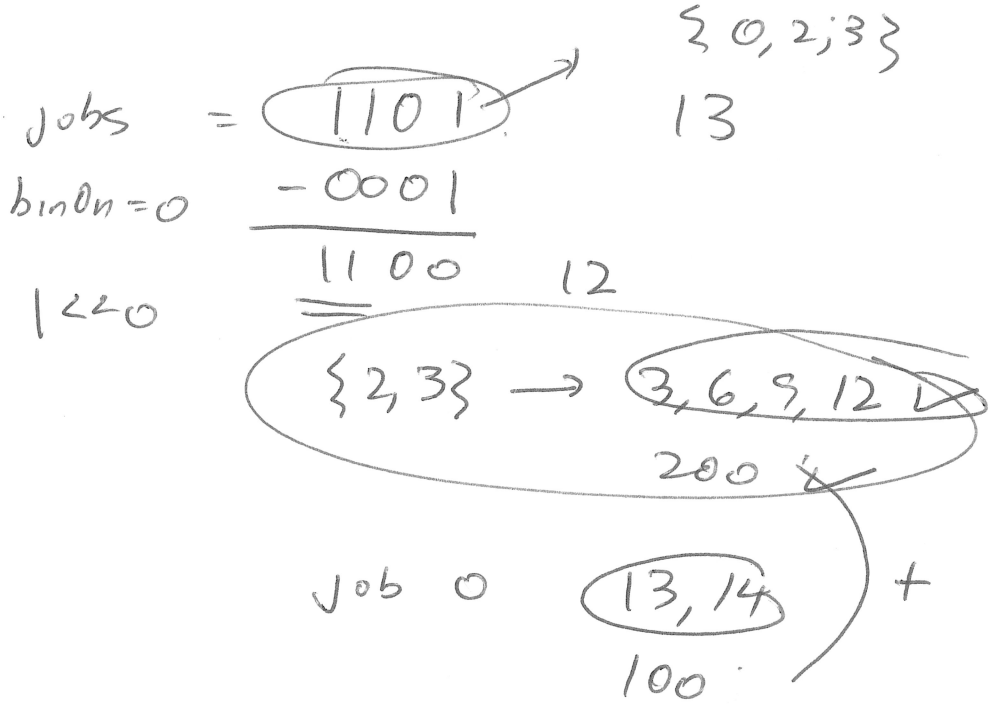
d = 2 — 2, 8

d = 5 — 13, 14, 15, 16 etc

HIGH LEVEL

BITMASK to go through all SUBSETS

* for each subset of jobs you check each pair for conflicts


{0,2,3}
jobs = 1101
13
bin0n=0
-0001
1100
12
1<<0
{2,3} → 3,6,9,12
200
Job 0
13,14
+
100