1.) Input: A, B, C → (n×n) matrix

Output: $A \cdot B = C$, if this is the case then yes, otherwise no

Run time $O(n^3)$ $\boxed{p \times q} \times \boxed{q \times r} = p \times q \times r$

Use randomized Algorithms to find runtime $O(n^2)$

A randomized Algorithm

1. Select a n×1 matrix whose entries are 0's or 1's

Ex) $A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 1 \\ 1 & 1 & 4 \end{bmatrix}$ $B = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 1 & 4 \\ 1 & 5 & 10 \end{bmatrix}$ $C = \begin{bmatrix} 2 & 1 & 10 \\ 1 & 5 & 11 \\ 2 & 1 & 1 \end{bmatrix}$

$A \cdot B = C$?

n×1 matrix
0 or 1 $r = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

$A \cdot B \cdot r = Cr =$ n×1 matrix

Ex) $A \cdot B \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \stackrel{?}{=} C \cdot \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$

$A \cdot \underbrace{B \cdot r}_{O(n^2)} = C \cdot r$

$\boxed{n \times n} \times \boxed{n \times 1} \rightarrow O(n^2)$ $\boxed{n \times n} \cdot \boxed{n \times 1} \rightarrow O(n^2)$

3. cases.

1. if $A \times B = C$ 100% correct

$AB = C$ if $A \cdot B \cdot r - Cr = 0$ then yes
$AB - C = 0$
$ABr - Cr = 0$
$(AB - C)\, r = 0$

2. if $A \times B \neq C$, $prob[error] \leq \frac{1}{2}$

$AB \neq C$
$AB - C \neq 0$
$ABr - Cr \neq 0$

$ABr = 0$, $Cr = 0$ probability

$(AB - C) r \neq 0$

n개 $\left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix} \right.$ 일 확률 $\leq \frac{1}{2}$.

if $A \cdot B \cdot r - Cr = 0$
and $ABr \neq 0$, $Cr \neq 0$
then yes.

Out of $n \times 1$ matrix, the probability of all n would be 0 is $\frac{1}{2^n}$ which is $\leq \frac{1}{2}$.

3. if $A \times B \neq C$, $prob[error] \leq \frac{1}{2}$

$AB \neq C$
$AB - C \neq 0$
$ABr - Cr \neq 0$

$(AB - C) \cdot r \neq 0$
$AB - C \neq 0$

$AB \neq C$ OR이면 $ABr = Cr$, $ABr \neq 0$, $Cr \neq 0$ 일경우. ?

2) input : $A = \{ \quad \}$, x

A: A set of n distinct integers, $n \geq 2$

Output: if there is Elements $\exists\, u, v \in A$ $u + v = x$ then yes, else no.

EX) $A = [-10, 6, 4, 2, 1, 11, 15]$, $x = 10$

$6 + 4 = 10$

Runtime $O(n \log n)$

```
def printpairs ( arr, size, n ):
```

```
New array = { }

for i in range (0, size):
    temp = n - arr[i]
      if temp in New arry:
          return true

      New arry [arr[i]] = i
  print(NO)
```

make a newarray for the range 0 ~ our size subtract each element from n and if it's not the answer add the element in to the new array and check for the next element.

3) compare the following 2 problems.

① 0/1 knapsack problem

② fractional knacksacp problem.

input : n distinct items, $, W (weights)

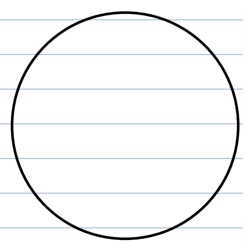
output : Optimal load ≤ W

Ex)

≤ 250 lb

5 items

	$ Value	(lb) weight
a_1	50	100
a_2	25	75
a_3	100	100
a_4	40	200
a_5	30	60



100 75 60

a_1, a_2, a_5

50 25 30 = 105

235 ≤ 250

knapsack

2) 50% a_4 20 / 20

Greedy Algorithm

Fractional knapsack
- items canbe broken into smaller pieces.

n items, the fraction rate : x ∴ $0 \le x_i \le 1$

weight $x_i \cdot W_i$
profit $x_i \cdot p_i$ (value)

maximize $\sum_{n=1}^{n} p_i \cdot x_i$, $\sum_{n=1}^{n} (x_i \cdot w_i) \le W$

optimal solution $\sum_{n=1}^{n} (x_i \cdot w_i) = W$

so find the ratio $\frac{p_i}{w_i}$ while largest ration select from the beginning

Ex)

	\$ Value	(lb) weight	$\frac{p}{w}$
a_1	50	100	$\frac{1}{2}$
a_2	25	75	$\frac{1}{3}$
a_3	100	100	1
a_4	40	200	$\frac{1}{5}$
a_5	30	60	$\frac{1}{2}$

$\rightarrow$ rearrange

	p	w	$\frac{p}{w}$
a_3	100	100	1
a_5	30	60	$\frac{1}{2}$
a_1	50	100	$\frac{1}{2}$
a_2	25	75	$\frac{1}{3}$
a_4	40	200	$\frac{1}{5}$

Ex 250lb is max so $a_3 + a_5 + \frac{9}{10} a_1 =$

$100 + 30 + 45 = \$175$ best profit