

바킹독의 실전 알고리즘

기초코드 작성요령 1

0x00 시간, 공간복잡도

시간복잡도(Time Complexity)

입력의 크기와 문제를 해결하는데 걸리는 시간의 상관관계

빅오표기법(Big-O Notation)

주어진 식을 값이 가장 큰 대표항만 남겨서 나타내는 방법.

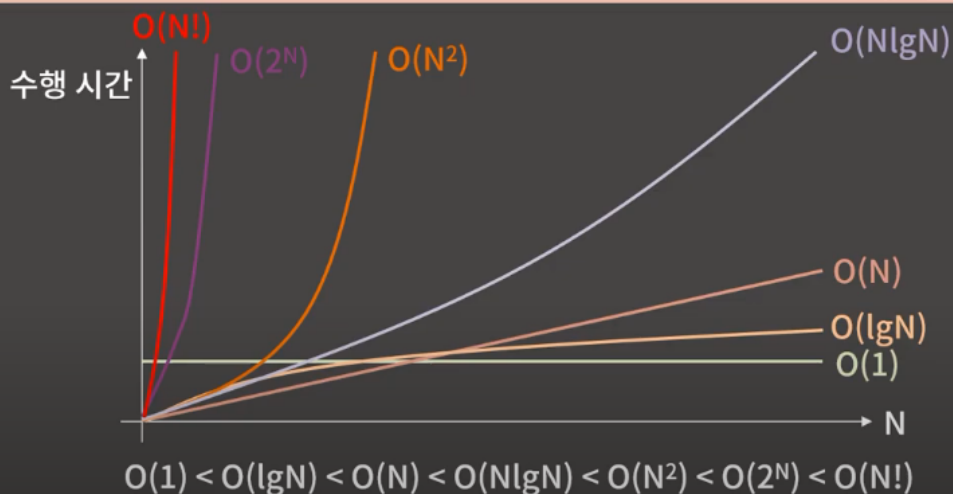
$O(N)$: $5N + 3$, $2N + 10\lg N$, $10N$

$O(N^2)$: $N^2 + 2N + 4$, $6N^2 + 20N + 10\lg N$

$O(N\lg N)$: $N\lg N + 30N + 10$, $5N\lg N + 6$

$O(1)$: 5, 16, 36

0x00 시간, 공간복잡도



N 의 크기	허용 시간복잡도
$N \leq 11$	$O(N!)$
$N \leq 25$	$O(2^N)$
$N \leq 100$	$O(N^4)$
$N \leq 500$	$O(N^3)$
$N \leq 3,000$	$O(N^2 \lg N)$
$N \leq 5,000$	$O(N^2)$
$N \leq 1,000,000$	$O(N \lg N)$
$N \leq 10,000,000$	$O(N)$
그 이상	$O(\lg N), O(1)$

unsigned char	0 0 0 0 1 0 0 1	→ $2^3 + 2^0 = 9$
	1 0 0 0 0 0 1 1	→ $2^7 + 2^1 + 2^0 = 131$
char	0 0 0 0 1 0 0 1	→ $2^3 + 2^0 = 9$
	1 0 0 0 0 0 1 1	→ $-2^7 + 2^1 + 2^0 = -125$

short (2 byte)	$2^{15} - 1 (=32767)$
int (4 byte)	$2^{31} - 1 (\doteq 2.1 \times 10^9)$
long long(8 byte)	$2^{63} - 1 (\doteq 9.2 \times 10^{18})$

Integer Overflow

0	0	0	0	1	0	0	1
---	---	---	---	---	---	---	---

$$2^3 + 2^0 = 9$$

+

0	0	0	1	0	1	0	1
---	---	---	---	---	---	---	---

$$2^4 + 2^2 + 2^0 = 21$$

=

0	0	0	1	1	1	1	0
---	---	---	---	---	---	---	---

$$2^4 + 2^3 + 2^2 + 2^1 = 30$$

0x02 실수 자료형

float (4 byte)

double (8 byte)

$$3 = 2^1 + 2^0 = 11_{(2)}$$

$$3.75 = 2 + 1 + 0.5 + 0.25 = 2^1 + 2^0 + 2^{-1} + 2^{-2} = 11.11_{(2)}$$

$$3561.234 = 3.561234 \times 10^3$$

$$11101.001_{(2)} = 1.1101001_{(2)} \times 2^4$$

float	sign(1)	exponent(8)	fraction(23)
double	sign(1)	exponent(11)	fraction(52)

1. 실수의 저장/연산 과정에서 반드시 오차가 발생할 수 밖에 없다.

```

01 int main(void) {
02     if(0.1+0.1+0.1 == 0.3) cout << "true";
03     else cout << "no no...";
04 }
05 /**result**
06 no no...
07 *****/

```

float : 유효숫자 6자리
double : 유효숫자 15자리

출력

첫째 줄에 놀이에 성공할 확률을 출력한다. 절대/상대 오차는 10^{-6} 까지 허용한다.

열에 아홉은 실수를 아홉고 모든 연산을 정수에서 해결할 수 있는 문제일거예요

2. double에 long long 범위의 정수를 함부로 담으면 안된다.

```

01 int main(void) {
02     double a = 10000000000000000001;
03     double b = 10000000000000000000;
04     if(a == b) cout << "wow..";
05     else cout << "a != b";
06 }
07 /**result**
08 wow..
09 *****/

```

3. 실수를 비교할 때는 등호를 사용하면 안된다.

```
01  int main(void) {  
02      double a = 0.1+0.1+0.1;  
03      double b = 0.3;  
04      if(a==b) cout << "same 1\n";  
05      if(abs(a-b) < 1e-12) cout << "same 2\n";  
06  }  
07  /**result**/  
08  same 2  
09  *****/
```