비트마스크(BitMask)

비트마스크(Bitmask)는 컴퓨터에서 정수를 이진수(binary)로 표현할 때, 각 비트(bit)의 상태를 이용하여 집합(set)을 표현하는 기법입니다. 예를 들어, 4개의 원소를 가지는 집합 `{0, 1, 2, 3}`을 비트마스크를 이용하여 표현하면, 0부터 15까지의 정수를 사용하여 표현할 수 있습니다.

0번 비트는 1번 원소가 집합에 속하는지, 1번 비트는 2번 원소가 집합에 속하는지, 2번 비트는 3번 원소가 집합에 속하는지, 3번 비트는 4번 원소가 집합에 속하는지를 나타냅니다. 예를 들어, `{0, 2}`를 비트마스크로 나타내면 1010(2진수) = 10(10진수)가 됩니다. 이때, 1번과 3번 비트가 1로 설정되어 있으므로 0번과 2번 원소가 집합에 포함됩니다.

자바에서는 비트마스크를 이용하여 연산을 수행할 수 있습니다. 예를 들어, 비트마스크를 이용하여 다음과 같은 연산을 수행할 수 있습니다.

1. 원소 추가: `set |= (1 << i)` (i번째 원소 추가)
2. 원소 제거: `set &= ~(1 << i)` (i번째 원소 제거)
3. 원소 포함 여부 확인: `(set & (1 << i)) > 0` (i번째 원소가 포함되어 있는지 확인)
4. 모든 부분집합 탐색: `for (int subset = set; subset > 0; subset = (subset - 1) & set)` (set의 모든 부분집합을 탐색)

비트마스크를 이용하여 부분집합의 합 문제를 해결하는 자바 코드

```
import java.util.*;

public class SubsetSum {
    public static void main(String[] args) {
        int[] arr = {1, 3, 5, 7};
        int n = arr.length;
        int target = 8;

        boolean found = false;
        for (int i = 0; i < (1 << n); i++) {
            int sum = 0;
            for (int j = 0; j < n; j++) {
                if ((i & (1 << j)) > 0) {
                    sum += arr[j];
                }
            }
            if (sum == target) {
                found = true;
                break;
            }
        }
        System.out.println(found ? "부분집합의 합이 존재합니다." : "부분집합의 합이 존재하지 않습니다.");
    }
}
```

위 코드에서 `arr` 배열은 부분집합을 구할 원소들을 담고 있습니다. `n`은 `arr`의 길이, `target`은 찾고자 하는 부분집합의 합입니다.

`for`문의 조건식인 `(1 << n)`은 부분집합의 개수인 `2^n`과 같습니다. `i` 변수는 0부터 `2^n-1`까지 증가하며, `i`의 2진수 표현에서 `j`번째 비트가 1인 경우 `arr[j]`를 더해 `sum`에 누적합니다.

마지막으로 `sum`이 `target`과 같은 경우 `found`를 `true`로 바꾸고 `break`를 수행합니다. `found`가 `true`이면 "부분집합의 합이 존재합니다."를 출력하고, 그렇지 않은 경우 "부분집합의 합이 존재하지 않습니다."를 출력합니다.