3) (10 pts) ALG (Backtracking)

Consider an arbitrary permutation of the integers 0, 1, 2, …, *n*-1. We define the "jumps" in a permutation array *perm* to be the set of values of the form *perm[i] - perm[i-1]*, with $1 \le i \le n-1$. For this problem you will write a backtracking solution count the number of permutations that can be created given a limited set of jumps. The function will take in arrays *perm*, representing the current permutation array, *used*, storing which items were used in the current permutation, *k*, the number of fixed items in the current permutation, *jumps*, an array storing the valid jumps allowed, and *len*, representing the length of the *jumps* array. The length of the *perm* and *used* arrays will be the constant N. Note that the jumps array contains both positive and negative values. For example, the permutation 3, 0, 2, 1 has the following jumps: -3, 2 and -1. **Complete the framework that has been given below to solve the problem.**

```
#include <stdio.h>
#define N 10

int numperms(int perm[], int used[], int k, int* jumps, int len) {
    int i, j, res = 0;

    if (k == N) return ___;
    for (i=0; i<N; i++) {

        if (used[i]) _______________;

        int flag = 0;
        if (k == 0)
            flag = ___;
        else {
            for (j=0; j < ____; j++)

                if ( _____________ == jumps[j])

                    flag = ___;
        }

        if (flag) {
            used[i] = ___;
            perm[k] = ___;
            res += numperms(perm, used, _____, jumps, len);
            used[i] = ___;
        }
    }

    return res;
}
```