3) (10 pts) ALG (Backtracking)

A D-digit divisible number is a positive integer of D digits (with no leading digits zero) such that each of its prefixes of k digits is a number divisible by k. For example, 52240 is a 5-digit divisble number because 5 is divisible by 1, 52 is divisible by 2, 522 is divisible by 3, 5224 is divisible by 4 and 52240 is divisible by 5. Assume that there exists a function as specified below:

```
int kDigitPrefixValue(char* number, int k);
```

such that if number is storing the string version of an integer that is at least k digits long, then the function will return the integer value of the first k digits represented in number. For example, `kDigitPrefixValue("52240", 4)` will return the integer 5224.

Complete the recursive function below so that it will print out all 6-digit divisible numbers. (A complete wrapper function is provided for you, so just fill out the blanks in the recursive function.)

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int kDigitPrefixValue(char* number, int k);
void printkDivisibleRec(char* number, int k);
void wrapper(int numdigits);

int main() {
    wrapper(6);
    return 0;
}

void wrapper(int numdigits) {
    char* tmp = malloc(sizeof(char)*(numdigits+1));
    int i;
    for (i=0; i<numdigits; i++) tmp[i] = '0';
    tmp[numdigits] = '\0';
    printkDivisibleRec(tmp, 0);
    free(tmp);
}

void printkDivisibleRec(char* number, int k) {
    if (k == strlen(number)) {
        printf("%s\n", number);
        return;
    }
    int i = k == 0 ? 1 : 0;
    for (; i < 10 ; i++) {                Grading: 1 pt per blank

        number[ k ] = (char)( i +'0');

        int prefix = kDigitPrefixValue(number, k+1 );

        if ( prefix %( k+1 ) == 0 )

            printDivisibleRec(number, k+1  );
    }
}
```