3) (10 pts) DSN (Backtracking)

Consider printing out all strings of x A's and y B's, where x ≥ y-1 such that no two consecutive letters are Bs, in alphabetical order. For example, if x = 5 and y = 3, one of the valid strings printed would be AABABABA. One way to solve this problem would be to use backtracking, where a string is built up, letter by letter (first trying A, then trying B in the current slot), but skipping trying A, if doing so would leave 2 more Bs to place than As, and skipping trying the B if the previous letter is a B. **Complete the code below to implement this backtracking solution idea.** (Hint: it's always okay to place B as the first letter. But if not placing the first letter, multiple conditions must be checked.)

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

void printAll(char buffer[], int k, int a, int b);
void printWrapper(int x, int y);

void printWrapper(int x, int y) {
    char* buffer = malloc(sizeof(char)*(x+y+1));
    buffer[x+y] = '\0';
    printAll(buffer, 0, x, y);
    free(buffer);
}

void printAll(char buffer[], int k, int x, int y) {

    if (x == 0 && y == 0) {
        printf("%s\n", buffer);
        return;
    }

    if (x > y-1) {
        buffer[k] = 'A' ;

        printAll(buffer, k+1 , x-1 , y );
    }

    if ( y > 0  && ( k == 0  || ( k > 0  && buffer[k-1] == 'A') ) ){

        buffer[k] = 'B' ;

        printAll(buffer, k+1  , x  , y-1 );
    }
}
```

Grading: 1 pt per slot, needs to be perfectly correct to get the point.