Комбинации C(K, N)

Дадени са ви първите N главни букви от английската азбука. Генерирайте всички възможни уникални техни подгрупи (подмножества) от K букви, като реда на буквите в подгрупата не е от значение (т.е 'AB' и 'BA' се смятат за една и съща подгрупа и са неуникални, а не са 2 различни подгрупи - от тях трябва да изпечатате 'AB', а не 'BA').
Жокер: За да генерираме всички комбинации C(K, N) от първите букви на англ. азбука трябва да направим K-вложени цикъла. Ki-тия цикъл ще ни генерира Ki-тата буква в поредната дума, която генерираме. След като генерираме и последната K-та буква на текущата дума я печатаме в края на най-вътрешния K-ти вложен цикъл и оставяме циклите да се извъртят до край. От колко до колко трябва да върти Ki-тия цикъл? Първият трабва да въртим от първата ('A') до N-K+1-вата буква, за да останат поне K-1 (= N-(N-K+1)) букви за останалите K-1 цикъла. Ако въртим до повече следващите вложени цикли няма как да вземат K-1 букви от по-малко от K-1 останали. Вторият вложен цикъл ще върти от буквата след избраната от първия цикъл до N-K+2-та буква по същите съображения, че трябва да остави поне K-2 букви за следващите K-2 вложени цикъла. Третият вложен цикъл ще върти от буквата след избраната от втория цикъл до N-K+3-та буква и т.н. Последният K-ти най-вложен цикъл по същата логика ще върти от буквата след избарната от K-1-ия вложен цикъл до последната N-K+K=N-та буква.
ПРОБЛЕМ: В нормален код на програма няма как да напишем променлив брой цикли.
РЕШЕНИЕ: Ще използуваме рекурсия (GenLetterCombinationsRecursively(int Ki, char StartLetter)) с два параметъра: Ki - номера на вложения цикъл, който ще въртим в текущото Ki-то ниво на вложеност на рекурсията; StartLetter - буквата веднага след тази избрана от горното (предния вложен цикъл, който обхваща текущия) ниво на рекурсията. Освен това ще ни трябва и помощен глобален масив Word в който да генерираме думата. На всяко ниво Ki на рекурсията записваме текущо генерираната буква на позиция Ki-1 (нула базирана) в масива Word (Word[Ki-1] = CurrentlyIteratedChar или Word.push_back(CurrentlyIteratedChar), ако Word е std::string). След това викаме следващото ниво на рекурсията (следващия вложен цикъл) така: GenLetterCombinationsRecursively(Ki+1, CurrentlyIteratedChar + 1)
Спокойно към променлива от тип char можем да прибавим едно, тъй като тя е 8-битово цяло число със знак (char('A' + 1) дава 'B'). Ако използуваме std::string за Word, то след като се върнем от долното ниво на рекурсия трябва да направим Word.pop_back().
Условието за край на рекурсията (проверяваме го в началото на рекурсивната функция) е Ki > K (K и N ще са глобални променлива). Когато то се изпълни нашата текуща дума Word е вече запълнена с текущите K букви. Печатаме текущо генерираната дума Word (ако Word е от тип char[], го затапваме преди печата с 0: Word[K] = 0) и излизаме веднага от текущото K+1-во ниво на рекурсията. Първото извикване на рекурсивната функция ще изглежда така:
GenLetterCombinationsRecursively(1, 'A').

Input Format

На първият ред е зададена стойноста на N, а на втория на K.

Constraints

1 <= N <= 20
1 <= K <= N

Output Format

Изпечатайте подредени по азбучен ред всички възможни уникални думи (по една на ред) с K различни букви избрани от първите N букви от английската азбука. Буквите във всяка дума също трябва да са подредени по азбучен ред.

Sample Input 0

```
4
2
```

Sample Output 0

```
AB
AC
AD
BC
BD
CD
```