**Algorithm explanation:**

We are supposed to find every answer to contain the each chapter in pages for only one side array.

First we should deal with the input, if the total number is 1, we should operate it in a different, and if the input is larger than 10, we can only input one digit each time and add it in the program.

Then we can use the total number as a flag to receive the input, we can identify the first input and the second by identifying the follow operation, space or enter, and for two digit, we use the same method behind.

To operate the data, we set the first input unchanged and call main(n-1) to end every cases in n-1 as total number, and then we change the data in first location, and call the main(n-1) the second time, after the second call, we swap the data again to recover it. In every call, if the number is reach zero, we call the check function to judge the array whether its elements are different from each other, if t is, we output this array.

After end all kinds of cases, we stop the program finally.

**Essential parts of your code with sufficient comments:**

*The major call part:*

opera ld r4,stonum ;r4 contain the total number

ld r5,stack1 ;load the start location of every used stack

ld r6,stack2

ld r0,stack3

add r3,r4, #0

jsr main ;call the recursion function

trap x25

*The recursion part:*

main add r3,r3, #0 ;r3 contain the current number, if the number equals to 0, make a output

brz test

add r2,r3, #-1 ;r2 equals to the n-1 in every circle

add r3,r2, #0 ;load the r2 to r3

jsr main ;call the main in size n-1

jsr swap ;swap the current data in stack1 and stack2

jsr main

jsr swap ;recover the original data structure

*The check function:*

and r1,r1, #0 ;use r1 to contain the state of a function

ld r5,stack1 ;judge the point is reach the end of the stack

add r6,r5,r4

not r5,r5

add r5,r5, #1 ;if reach the end, adjust the factor and continue

add r3,r6, #0 ;make a copy of the stack point

add r6,r6, #-1

loop3 add r2,r5,r6 ;ergodic the stack, to find if there exists a pair of same number

brz loop4

ldr r2,r6, #0

not r2,r2

add r2,r2, #1

ldr r4,r3, #0

add r2,r2,r4

brz fail ;if there exists, set r1 to 1

add r6,r6, #-1 ;the condition of inner circle

brnzp loop3

loop4 add r3,r3, #-1 ;the condition of outer circle

add r2,r3,r5

brz suces

add r6,r3, #-1

brnzp loop3

*The output function:*

ld r4,stonum

ld r5,stack1

loop5 add r5,r5, #1 ;load number from the stack

ldr r0,r5, #0 ;store the first data in the stack

ld r2,miten

add r2,r2,r0 ;judge the number is larger than 10 or not

brzp deal

ld r3,pldigit ;transfer the digit to ASCII code

add r0,r0,r3

next trap x21 ;reduce the line number

add r4,r4, #-1 ;if the number is larger than 0, output a space in each pair of digit

brz over1

ld r0,data4 ;during two digits, we output a space between them

trap x21

brnzp loop5

deal ld r2,miten ;if the digit is larger than 10

add r3,r0,r2 ;we output it in tens digits and single digit

ld r0,pldigit

add r0,r0, #1 ;first we output the first numebr 1

trap x21

ld r0,pldigit

add r0,r0,r3 ;then we calculate the second number

brnzp next

**Questions and Answers:**

Question 1:

Please describe your algorithm.

Answer 1:

Similar to algorithm explanation mode, and add some necessary comments of the machine code. See above for details.

Question 2:

When do you check your having been swaped data, and output it ?

Answer 2:

I check it when the recursion reach the end, which means the n is equals to 0, and then I check the stack one, if the element in it all are different from each other, I set the flag and output it.

Follow 2:

But in this case, we will lost a lot of time, because if we can check the data in the middle of the recursion, we can save lots of time to calculate the data when a mistake have been taken.