CS29003 ALGORITHMS LABORATORY
Warm-up Assignment
Not for Submission

Part 1

You are given an array A of n integers. You are also given an integer s in the range $1 \leq s \leq n-1$. Your task is to right rotate (cyclically right shift) the array A by s cells. Write a program to solve this problem. Your program must not use any extra array (other than that needed to store A itself).

Of course, if you right shift A by one cell for a total of s times, you have achieved your goal. However, this strategy calls for a total effort of roughly proportional to ns operations, which may be large unless s is small. Your program should run in time independent of s, that is, the number of operations to be performed by your program should be roughly proportional to n only.

Hint: Let $n = 15$ and $s = 6$. Here are the 15 data-movement operations, where $i \rightarrow j$ indicates that $A[i]$ should go to $A[j]$.

$0 \rightarrow 6 \rightarrow 12 \rightarrow 3 \rightarrow 9 \rightarrow 0$
$1 \rightarrow 7 \rightarrow 13 \rightarrow 4 \rightarrow 10 \rightarrow 1$
$2 \rightarrow 8 \rightarrow 14 \rightarrow 5 \rightarrow 11 \rightarrow 2$

If $n = 15$ and $s = 7$, then the data-movement operations are as follows:

$0 \rightarrow 7 \rightarrow 14 \rightarrow 6 \rightarrow 13 \rightarrow 5 \rightarrow 12 \rightarrow 4 \rightarrow 11 \rightarrow 3 \rightarrow 10 \rightarrow 2 \rightarrow 9 \rightarrow 1 \rightarrow 8 \rightarrow 0$

Part 2

You are given an $n \times n$ board. Your task is to place m coins on the board such that no two of the coins go to the same cell or to two adjacent cells. Two cells are called adjacent if their boundaries share an edge. Two cells with boundaries sharing only a corner will not be called adjacent. Write a program that prints all possible arrangements of m coins in the $n \times n$ board. Your program must not use any global or static variables.

Here is a sample output for $n = 4$ and $m = 7$. The output is shown in a multi-column format to save space.

```
Arrangement 1:
 x . x .
 . x . x
 x . x .
 . x . .
Arrangement 2:
 x . x .
 . x . x
 x . x .
 . . . x
Arrangement 3:
 x . x .
 . x . x
 x . . .
 . x . x
Arrangement 4:
 x . x .
 . x . x
 . . x .
 x . . x
Arrangement 5:
 x . x .
 . x . x
 . . x .
 . x . x
Arrangement 6:
 x . x .
 . x . .
 x . x .
 . x . x
Arrangement 7:
 x . x .
 . . . x
 x . x .
 . x . x
Arrangement 8:
 x . . x
 . x . .
 x . x .
 . x . x
Arrangement 9:
 x . . x
 . . x .
 . x . x
 x . x .
Arrangement 10:
 x . . .
 . x . x
 x . x .
 . x . x
Arrangement 11:
 . x . x
 x . x .
 . x . x
 x . . .
Arrangement 12:
 . x . x
 x . x .
 . x . x
 . . x .
Arrangement 13:
 . x . x
 x . x .
 . x . .
 x . x .
Arrangement 14:
 . x . x
 x . x .
 . x . .
 x . . x
Arrangement 15:
 . x . x
 x . x .
 . . . x
 x . x .
Arrangement 16:
 . x . x
 x . . .
 . x . x
 x . x .
Arrangement 17:
 . x . x
 . . x .
 . x . x
 x . x .
Arrangement 18:
 . x . .
 x . x .
 . x . x
 x . x .
Arrangement 19:
 . . x .
 . x . x
 x . x .
 . x . x
Arrangement 20:
 . . . x
 x . x .
 . x . x
 x . x .
```

Hint: Write a recursive function. Achieve an output as shown above by suitable parameter passing only.
