Foster Home

Objective

Give practice with reading and writing to standard input in C.
Give practice with loops and conditionals in C.
Give practice with functions in C.
Give practice with strings in C.

Story

You have recently been hired to help out with Kaylee and Tran's (KaT) Pet Shop. There are 3 foster homes that help take care of cats. Every week they take in a cat. When the week is over, they bring the cat back, but the home leaves their mark by adding a signature suffix to the name of the cat.

The cats to adopt are arranged in a row of ***N***, which is at least 3, cozy cages. You may assume there is exactly 1 cat per cozy cage. The first family adopted the cat in cage 1. The second family adopted the cat in cage 2. The third family adopted the cat in cage 3. After they bring their cat back they will adopt a new cat. The foster families typically want to adopt a different cat for the week so they will move over some number of positions in the list of cages and adopt the cat in that location. If there is no cat in that cage that week, the family will go catless for the week (🙁), remain at that position, and then resume counting from that position in the following week. If they reach the last cage when counting, they move back to the first cage and resume counting from there.

We know the following about the three foster homes,

- The First Family
    - Adopts the cat in position 1 in the first week
    - Adopts on Monday
    - Adds "Lee" to the end of the name
    - Moves 2 cages down
- The Second Family
    - Adopts the cat in position 2 in the first week
    - Adopts on Tuesday
    - Adds "Lyn" to the end of the name
    - Moves 3 cages down
- The Third Family
    - Adopts the cat in position 3 in the first week
    - Adopts on Wednesday
    - Adds "Eve" to the end of the name
    - Moves 5 cages down

You are pretty sure that none of the original cats have been adopted. These families have been adopting cats for some time now, and you want to know what are the current cats available for adoption.

Problem

Given the initial cages and the number of weeks the adoption has occurred determine the names of the cats present on the Friday of that week.

Input

The first line of input will contain 2 integers, ***N*** and ***K***, (3 ≤ ***N*** ≤ 500, 1 ≤ ***K*** ≤ 500) representing the number of cozy cages and the number of weeks of which the adoption process has occurred, respectively.

The following ***N*** lines will each contain a single string of up to 19 characters representing the original name of the cat.

Output

Output ***N*** lines each containing the current name of the cat in the cozy cage at the end of the ***K*** weeks. If no cat is present print “No cat found.” instead.

Sample Input	**Sample Output**

`6 4`
`Steve`
`CrunchyCatLuna`
`PrincessHoneybelle`
`Bruce`
`TippyToe`
`Ili`
	`SteveLeeEve`
	`CrunchyCatLunaLynEve`
	`PrincessHoneybelleEve`
	`Bruce`
	`No cat found.`
	`No cat found.`

`5 1`
`Midna`
`Chloe`
`KhaleesiCat`
`Megan`
`RubyTheBabyCat`
	`No cat found.`
	`No cat found.`
	`No cat found.`
	`Megan`
	`RubyTheBabyCat`

`8 6`
`Moosh`
`Liz`
`Alligator`
`Preppy`
`LAPOCHKA`
`jimothy`
`toebeans`
`arson`
	`No cat found.`
	`LizLynEve`
	`No cat found.`
	`No cat found.`
	`LAPOCHKALynEve`
	`jimothyLyn`
	`toebeansLeeEve`
	`arsonEve`

Sample Explanation

FOR CASE 1
In the first case there are 6 cats, and we will let the cats be adopted for 4 weeks total. Below is an table representing the situation at the end of the first week,

Cat	Steve	CrunchyCatLuna	PrincessHoneyBelle	Bruce	TippyToe	Ili
Family 1	Here (with cat)					
Family 2		Here (with cat)				
Family 3			Here (with cat)			

If we printed the list now we would have

```
No cat found.
No cat found.
No cat found.
Bruce
TippyToe
Ili
```

But we need to reach week 4! Let the cats come back at the beginning of the week. The 1st foster family arrives and drops off Steve (now called “SteveLee”). They move over 2 places and try to adopt PrincessHoneyBelle, BUT that cat is not at that location, so the first foster family goes catless for this week.

Cat	SteveLee	CrunchyCatLuna	PrincessHoneyBelle	Bruce	TippyToe	Ili
Family 1			Here (with no cat)			
Family 2		Here (with cat)				
Family 3			Here (with cat)			

The 2nd family arrives and drops off CrunchyCatLuna (now called “CrunchyCatLunaLyn”). They move over 3 places and try to adopt TippyToe, AND they succeed!!!

Cat	SteveLee	CrunchyCatLunaLyn	PrincessHoneyBelle	Bruce	TippyToe	Ili
Family 1			Here (with no cat)			
Family 2					Here (with cat)	
Family 3			Here (with cat)			

The 3rd family arrives and drops off PrincessHoneyBelle (now called “PrincessHoneyBelleEve”). They move over 5 places, but they move past Ili and have to wrap back around. They end up at the cozy cage with CruchCatLunaLyn. They are able to successfully adopt. The following table is the result of the end of week 3,

Cat	SteveLee	CrunchyCatLunaLyn	PrincessHoneyBelle	Bruce	TippyToe	Ili
Family 1			Here (with no cat)			
Family 2					Here (with cat)	
Family 3		Here (with cat)				

At the end of week 3 we have the following,

Cat	SteveLee	CrunchyCatLunaLynEve	PrincessHoneyBelleEve	Bruce	TippyToeLyn	Ili
Family 1					Here (with no cat)	
Family 2		Here (with no cat)				
Family 3	Here (with cat)					

At the end of week 4 we have the following,

Cat	SteveLeeEve	CrunchyCatLunaLynEve	PrincessHoneyBelleEve	Bruce	TippyToeLyn	Ili
Family 1	Here (with no cat)					
Family 2					Here (with cat)	
Family 3						Here (with cat)

This is why the resulting answer is

```
SteveLeeEve
CrunchyCatLunaLynEve
PrincessHoneybelleEve
Bruce
No cat found.
No cat found.
```

FOR CASE 2

We have 5 cats:

- Minda
- Chloe
- KhaleesiCat
- Megan
- RubyTheBabyCat

At the end of the first week is the following

Cat	Minda	Chloe	KahleesiCat	Megan	RubyTheBabyCat
Family 1	Here (with cat)				
Family 2		Here (with cat)			
Family 3			Here (with cat)		

This is what we want, so the resulting output would be the following,

```
No cat found.
No cat found.
No cat found.
Megan
RubyTheBabyCat
```

FOR CASE 3

We have 8 cats and have to wait 6 weeks. Below are the names of the cats as the weeks progress. The colors denote which cats are adopted by which family. The numbers denote which cozy cage the family is considering.

Week	Moosh	Liz	Alligator	Preppy	LAPOCHKA	jimothy	toebeans	arson
1	NONE 1	NONE 2	NONE 3	Preppy	LAPOCHKA	jimothy	toebeans	arson
2	MooshLee	LizLyn	AlligatorEve 1	Preppy	NONE 2	jimothy	toebeans	NONE 3
3	MooshLee	LizLyn	AlligatorEve	Preppy	NONE 1 3	jimothy	toebeans	arsonEve 2
4	MooshLee	NONE 3	NONE 2	Preppy	LAPOCHKALynEve	jimothy	NONE 1	arsonEve
5	NONE 1	LizLynEve	AlligatorEveLyn	Preppy	LAPOCHKALynEve	NONE 2	NONE 3	arsonEve
6	NONE 2	LizLynEve	NONE 1	NONE 3	LAPOCHKALynEve	jimothyLyn	toebeansLeeEve	arsonEve

The cats names are the following based on the last row (week).

```
No cat found.
LizLynEve
No cat found.
No cat found.
LAPOCHKALynEve
jimothyLyn
toebeansLeeEve
arsonEve
```

Hints

Cat Name Array: I made an array to hold the cat names. Since each cat name is a string. I made an array of array of characters (a 2D array of characters) for the cat names.

String Size: Make sure to create character arrays that are long enough for the full cat name at the end of the time.

String Function: I recommend using the strcat function.

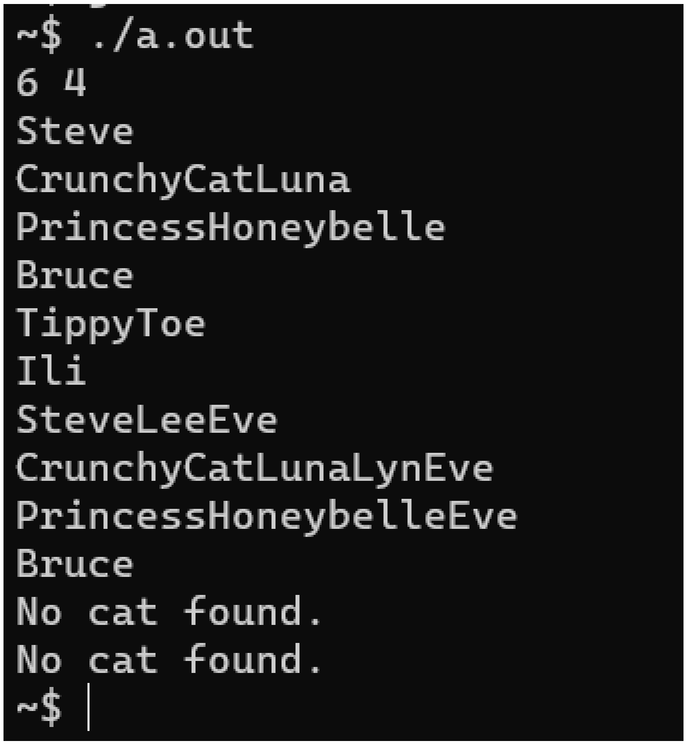
Foster Struct: I personally made a struct for the foster family with an array of 3 families.

Cat Ownership: Make sure to store which family has a cat. It can be frustrating when multiple families all want the same cat.

NO PROMPTS: Do NOT prompt for input. Do not print messages like, “Please enter the number of cats:” or the like.

NO LABELS: Do NOT label the output. Do not print messages like, “The cat in cozy cage 1 is…“.

Input Output Syncing: You DO NOT need to sync the input and output. Below is an example of what the execution of the program could look like,



```
~$ ./a.out
6 4
Steve
CrunchyCatLuna
PrincessHoneybelle
Bruce
TippyToe
Ili
SteveLeeEve
CrunchyCatLunaLynEve
PrincessHoneybelleEve
Bruce
No cat found.
No cat found.
~$
```

The first 7 lines are all from the user, and the last 7 lines of output are from the program.

Grading Criteria

- Read/Write from/to **standard** input/output (e.g. scanf/printf and no FILE *)
    - 10 points
- Good comments, whitespace, and variable names
    - 15 points
- No extra input output (e.g. input prompts, “Please enter the number of cats:”)
    - 10 points
- Read in all the input
    - 5 points
- Loop over all characters in both messages
    - 10 points
- Programs will be tested on 10 cases
    - 5 points each

No points will be awarded to programs that do not compile using “gcc -std=gnu11 -lm”.

Sometimes a requested technique will be given, and solutions without the requested technique will have their maximum points total reduced. For this problem use a loop and conditional. ***Without this programs will earn at most 50 points!***

Any case that causes a program to return a non-zero return code will be treated as wrong. Additionally, any case that takes longer than the maximum allowed time (the max of {5 times my solutions time, 10 seconds}) will also be treated as wrong.

No partial credit will be awarded for an incorrect case.