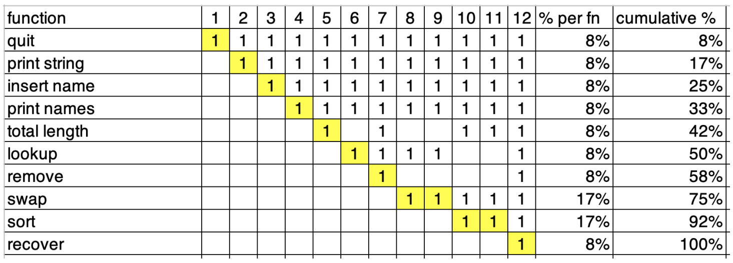
Computation I 5EIA0
Homework 4: Sorting and Pointers (v0.5, September 22, 2022)
Deadline Tuesday 4 October 13:30

In this homework you will a program that sorts strings. It is an interactive program that you can give commands. These are the commands that your program will support.

command	operation
q	quit program
p	print string
n	print names
i	inst name
e	total length of all names
l	look up name
r	remove name
s	swap names
o	sort names
v	recover removed names



function	1	2	3	4	5	6	7	8	9	10	11	12	% per fn	cumulative %
quit	1	1	1	1	1	1	1	1	1	1	1	1	8%	8%
print string		1	1	1	1	1	1	1	1	1	1	1	8%	17%
insert name			1	1	1	1	1	1	1	1	1	1	8%	25%
print names				1	1	1	1	1	1	1	1	1	8%	33%
total length					1		1			1	1	1	8%	42%
lookup						1	1	1	1			1	8%	50%
remove							1					1	8%	58%
swap								1	1	1	1	1	17%	75%
sort										1	1	1	17%	92%
recover												1	8%	100%

Figure 1: Test cases.

Task 1. In this task we'll implement the `void printString(char string[])` function that prints all characters in the array. Print `\0` if the character is the null character. All names are stored in a single string `char names[LENGTH]`, where `LENGTH` is defined as 100 (with a `#define`). Declare it in your main function, and initialise it with a 100 null characters. Call `printString(names)` to print it on the screen. Then copy the string `"Asterix"` into the array, and call `PrintString` again. This should be the output:

```
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Asterix\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
```

(The line is too long to fit on the page and not all 100 characters are shown. But those not shown are `\0`.)

Hint: Remember how to initialise arrays. The longest (and not recommended!) way would be: `char names[100] = { '\0', '\0', ... in total a 100 times ..., '\0' };`
Since the compiler initalises remaining entries to zero (i.e. the null character), this can be done with much less typing. Note that `char names[100];` does *not* initialise the array, and it will contain random rubbish.

Hint: If you get the output

```
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Asterix\0
```

Then something went wrong! Note that `names = "Asterix";` is not correct since now `names` points to an array of 8 characters and no longer to the array of 100 characters. You need to copy the string Asterix into the array of 100 characters. For this use the `strcpy` function from the `string.h` library. Add `#include <string.h>` to your program to include the library. See Kernighan & Ritchie Appendix B3 for documentation on all string functions. It is good to get to know them and use them because they save a lot of work and you can also use them in the exam.

Important

Some of you may know that it is also possible to declare variables *outside* functions.
While in some cases this may make your program easier, all homework up to week 7 requires that you declare all variables inside functions. After all, the purpose of the homework is to practice with functions and function arguments.
In exams, declaring variables outside functions when it is asked to declare them inside, will lead to a zero grade for each test that uses that variable. (This will be checked after the exam, and your grade will be lower than indicated by the Oncourse evaluation.)

Task 2. Since `printString` is working you can remove the copying of `"Asterix"` and calls to `printString` from your main function. Write a loop to ask for a command. A command is a single character. To read a character use the `scanf(" %c", &cmd);` function call, where `cmd` is a variable of type `char`. Notice that there is a space in front of the `%c`, which instructs `scanf` to skip white space such as space, tab, and newline.

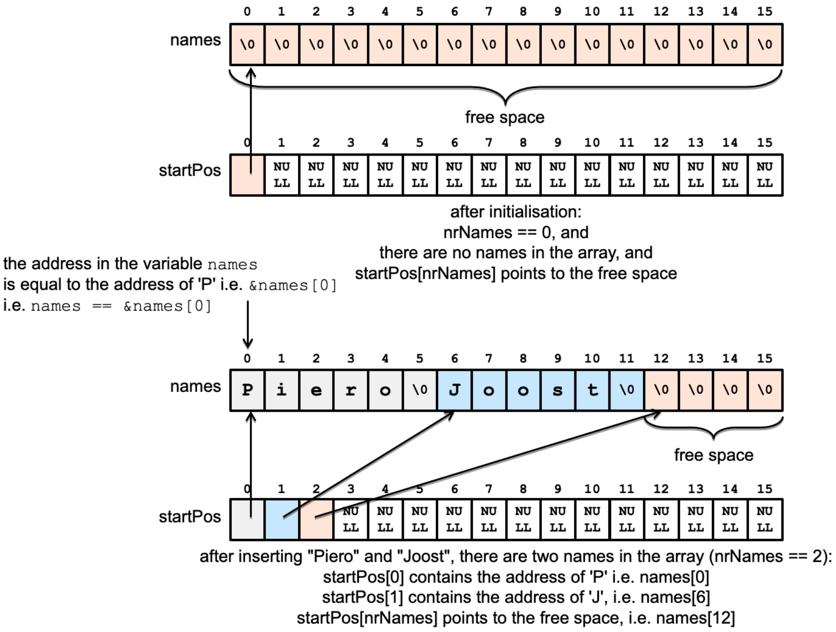
On receiving the 'p' (print) command call the `printString` function on the `names` array. On receiving the 'q' (quit) command print `Bye!` (with a newline of course) and exit the program. For all other (invalid) commands print the error message shown below and ask for another command. The output of your program should now be:

```
Command? p
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? X
Unknown command 'X'
Command? q
Bye!
```

Hint: The statement `scanf(" %c",&cmd);` is one that you will use most in all homeworks and exams. It is good to get to know it and remember it.

Hint: The sample output makes clear whether something that is printed should have a newline or not. Almost all of the time you'll need a newline, but printing a prompt to ask for input will not have a newline. The only possible confusion is with the final `Bye!` but since it's not a prompt it should have a newline.

Task 3. As mentioned before, all names are stored in a single string `char names[LENGTH]` declared in the main function. We use the `int nrNames` to keep track of the total number of names stored in the array. Declare `nrNames` inside the main function. To keep track of the order of the names we use an array to store the starting position of the names. For this we use an array of pointers `char * startPos[LENGTH]` declared in the main function. Thus each `startPos[i]` is a `char *`, i.e. a pointer to a character.



after initialisation:
nrNames == 0, and
there are no names in the array, and
startPos[nrNames] points to the free space

the address in the variable `names`
is equal to the address of 'P' i.e. `&names[0]`
i.e. `names == &names[0]`

after inserting "Piero" and "Joost", there are two names in the array (nrNames == 2):
startPos[0] contains the address of 'P' i.e. names[0]
startPos[1] contains the address of 'J', i.e. names[6]
startPos[nrNames] points to the free space, i.e. names[12]

Figure 2: The `names` array and the `startPos` array.

After inserting the names `"Piero"` and `"Joost"` `startPos[0]` contains the address of the first letter (`'P'`) of the first name (`"Piero"`), i.e. the address at which the first name starts in the `names` array. Similarly `startPos[nrNames-1]` contains the address of the first letter (`'J'`) of the last (second) name (`"Joost"`), i.e. the address at which the second name starts in the `names` array. Finally, `startPos[nrNames]` contains the first address after the last name, i.e. where the free space in the `names` array starts.
Declare and initialise the `names`, `startPos`, and `nrNames` variables.

Hint: The first element of the `startPos` array must indicate where the free space starts! Recall that an array is a sequence of consecutive elements in memory. The array variable (`names`) contains the address of the first element of the array (`&names[0]`), and therefore `names == &names[0]`. It's up to you which one you prefer to assign to `startPos[0]`.

Task 4. Write the function void insertName(char *startPos[], int *nrNames, char newName[]). It must copy the string newName into the names array. However, as you can see, the names array is not an argument to the function. As shown below startPos[nrNames] points to the free space in the names array. We can therefore reach the free space in names through a pointer stored in the startPos arrays without using the names array directly, see Figure 2.

The first thing to do is to copy the new string into the free space using the strcpy function from the string.h library. (Of course you can write the code yourself too, if you wish.) After that you need to update the startPos array since you've added a new string. After copying, the address startPos[nrNames] already points to the right character, namely the first character of the newly inserted string. But you still need increment nrNames (since there's one more string in the array) and then update startPos[nrNames] to point to the new free space. What is the address where the free space now starts? Note that nrNames is a pointer to an integer (int *), not an integer, because we want to modify it. The following figure shows the three steps, starting from nrNames == 0 (shown in Figure 2).

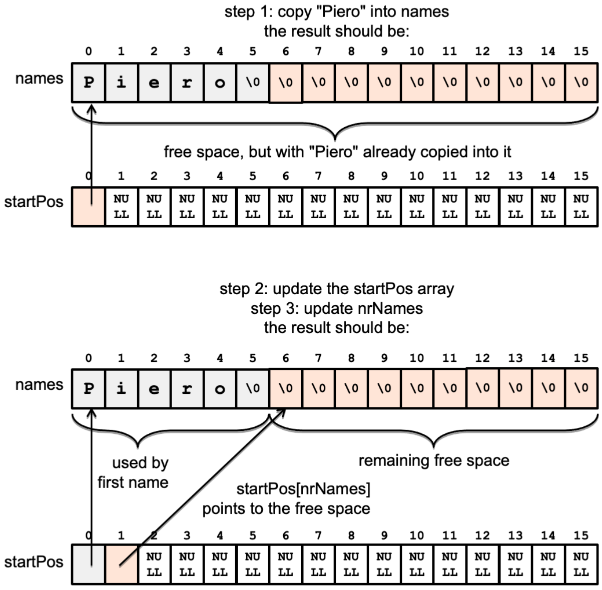


Figure 3: The names array and the startPos array before and after inserting the first name.

Add the 'i' command in your main loop to ask for a new name and insert it using the insertName function. Names don't contain spaces and you can therefore use the scanf("%s", name); call.

```
Command? p
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? i
Name? Piero
Command? p
Piero\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? i
Name? Joost
Command? p
Piero\0Joost\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? q
Bye!
```

Hint: The new string has a certain length (contains a number of characters). The free space is reduced by this amount, i.e. the start of the free space is increased by this amount. You code will look something like this (it really is only three lines):

```
    // copy the string to the free space
    // startPos[*nrNames+1] = old start of free space + ...
    // increment nrNames
```

Task 5. We're not really interested in looking at the string containing all the names. Instead we wish to print the list of names in the string. Implement the 'n' (names) command to print the list of name in the order indicated by the `startPos` array. Write the function `void printNames(char names[], char *startPos[], int nrNames)` that prints the starting position of each name, the name, and its length including the terminating null character. (Note that the standard `strlen` function does *not* include the null character.) The starting position is the index in the `names` array. Your program output should look like this:

```
Command? n
Command? i
Name? Asterix
Command? n
startPos[0]=00 length=08 string="Asterix"
Command? i
Name? Obelix
Command? n
startPos[0]=00 length=08 string="Asterix"
startPos[1]=08 length=07 string="Obelix"
Command? i
Name? Abraracourix
Command? n
startPos[0]=00 length=08 string="Asterix"
startPos[1]=08 length=07 string="Obelix"
startPos[2]=15 length=13 string="Abraracourix"
Command? p
Asterix\0Obelix\0Abraracourix\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command?
q
Bye!
```

Hint: The `startPos` array contains the starting points of all names (strings), and you know how many elements are valid. You can therefore use a for loop to run through all of them. Since you have the start address of the string, you can print it like a normal string using `printf("%s",...);`.

Hint: `startPos` contains addresses, not offsets into the `names` array. You can subtract one address from another to get the offset in the `names` array, as required.

Hint: Note the formatting of the integers: they are printed in with two digits and have leading zeros. See K&R B1.2 for details.

Task 6. Write the function `int totalLength(char *startPos[], int nrNames)` that returns the sum of the lengths of all names in the `startPos` list. Add the 'e' (lEngth) command.

```
Command? i
Name? Getafix
Command? i
Name? Vitalstatistix
Command? i
Name? Getafix
Command? p
Getafix\0Vitalstatistix\0Getafix\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? n
startPos[0]=00 length=08 string="Getafix"
startPos[1]=08 length=15 string="Vitalstatistix"
startPos[2]=23 length=08 string="Getafix"
Command? e
Total length is 31
Command? q
Bye!
```

Task 7. Write the function `int lookupNamePos(char names[], char *startPos[], int nrNames, char name[])` that returns the *index* of the name in the `names` string. If there are multiple copies of the name then return the one with the largest index. Return -1 if the name is not in the string. Add the 'l' (lookup) command.

```
Command? l
Name? Getafix
"Getafix" has index -1
Command? i
Name? Getafix
Command? i
Name? Vitalstatistix
Command? i
Name? Getafix
Command? p
Getafix\0Vitalstatistix\0Getafix\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? l
Name? Getafix
"Getafix" has index 23
Command? l
Name? Vitalstatistix
"Vitalstatistix" has index 8
Command? q
Bye!
```

Hint: The `strcmp` function from the `string.h` library is very useful.

Hint: Loop through the `startPos` array to find the name. At this point you know the position of the name in `startPos` array (`i` let's say) and its address in memory (somewhere in the `names` array). Now you need to compute the index of the name in the `names` array, which must be in the range 0 up to `LENGTH`. Hint: you need to do arithmetic (subtraction) on two pointers.

Task 8. Since we can add names, we should also be able to remove names. Write the function `void removeName(char *startPos[], int *nrNames, char remName[])` that removes the first occurrence of `remName` in the `startPos` list. Move the last name in the `startPos` list to the position of the name that you want to remove. Do nothing if the name is not in the `startPos` list. Add the 'r' (remove) command where you ask for a name and then call `removeName`.

```
Command? i
Name? one
Command? i
Name? two
Command? i
Name? three
Command? i
Name? four
Command? n
startPos[0]=00 length=04 string="one"
startPos[1]=04 length=04 string="two"
startPos[2]=08 length=06 string="three"
startPos[3]=14 length=05 string="four"
Command? p
one\0two\0three\0four\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? r
Name? two
Command? n
startPos[0]=00 length=04 string="one"
startPos[1]=14 length=05 string="four"
startPos[2]=08 length=06 string="three"
Command? p
one\0two\0three\0four\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? r
Name? one
Command? n
startPos[0]=08 length=06 string="three"
startPos[1]=14 length=05 string="four"
Command? r
Name? three
Command? n
startPos[0]=14 length=05 string="four"
Command? p
one\0two\0three\0four\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? q
Bye!
```

Hint: Note that moving the last name in the `startPos` array to the position of the name to be removed means that removing `two` has the result:

```
startPos[0]=00 length=04 string="one"
startPos[1]=14 length=05 string="four"
startPos[2]=08 length=06 string="three"
```

and NOT:

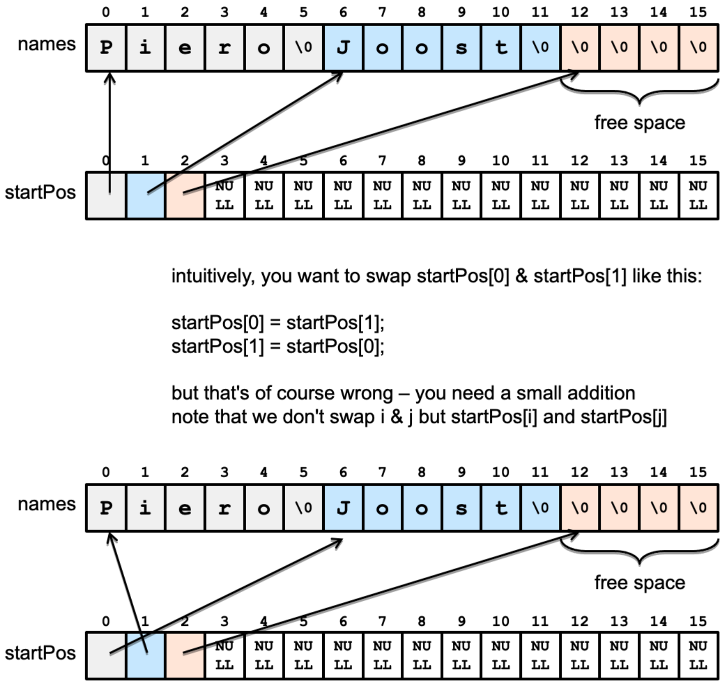
```
startPos[0]=00 length=04 string="one"
startPos[1]=08 length=06 string="three"
startPos[2]=14 length=05 string="four"
```

Hint: With this command we see for the first time the advantage of decoupling the storing of the names in the `names` array and keeping track of the starting positions in the `startPos` array. Now we do not touch the data in the `names` array (you can just leave the removed name there). If we wouldn't have the `startPos` array then we would have had to shift data around in the `names` array.

Task 9. To be able to sort the names in the names array we first implement the `void swapNames(char *startPos[], int i, int j)` function. It swaps the i^{th} and j^{th} entries in the `startPos` array.

```
Command? i
Name? Dogmatix
Command? i
Name? Geriatrix
Command? i
Name? Confoundtheirpolitix
Command? i
Name? Psychoanalytix
Command? n
startPos[0]=00 length=09 string="Dogmatix"
startPos[1]=09 length=10 string="Geriatrix"
startPos[2]=19 length=21 string="Confoundtheirpolitix"
startPos[3]=40 length=15 string="Psychoanalytix"
Command? p
Dogmatix\0Geriatrix\0Confoundtheirpolitix\0Psychoanalytix\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? s
Indices? 1 2
Command? n
startPos[0]=00 length=09 string="Dogmatix"
startPos[1]=19 length=21 string="Confoundtheirpolitix"
startPos[2]=09 length=10 string="Geriatrix"
startPos[3]=40 length=15 string="Psychoanalytix"
Command? p
Dogmatix\0Geriatrix\0Confoundtheirpolitix\0Psychoanalytix\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? q
Bye!
```

Hint: Don't overthink this one; it only requires three lines of code. The swap function was introduced in the lecture.



intuitively, you want to swap startPos[0] & startPos[1] like this:

startPos[0] = startPos[1];
startPos[1] = startPos[0];

but that's of course wrong – you need a small addition
note that we don't swap i & j but startPos[i] and startPos[j]

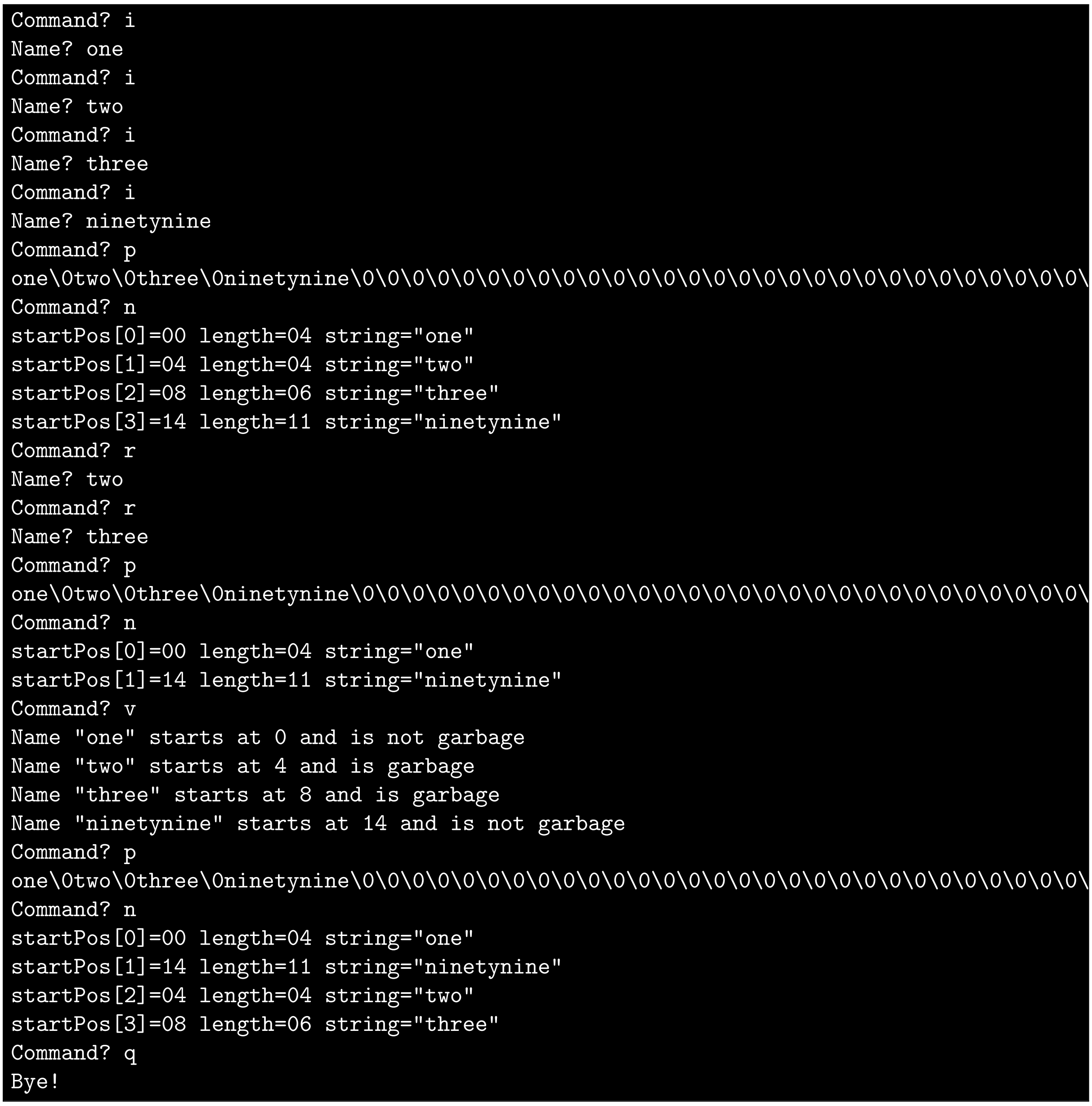
Figure 4: Swapping two names.

Task 10. The final task is to sort the `names` array with the `void insertionSortNames(char *startPos[], int nrNames)` function. Insertion sorting repeatedly swaps elements until the array is sorted. See `https://en.wikipedia.org/wiki/Insertion_sort` for a description. Use your `swapNames` function to swap the names. Sort with the 'o' command.

```
Command? i
Name? Ordalfabetrix
Command? i
Name? Kostunrix
Command? i
Name? Amaryllix
Command? i
Name? Aerobiks
Command? i
Name? Fishmix
Command? n
startPos[0]=00 length=14 string="Ordalfabetrix"
startPos[1]=14 length=10 string="Kostunrix"
startPos[2]=24 length=10 string="Amaryllix"
startPos[3]=34 length=09 string="Aerobiks"
startPos[4]=43 length=08 string="Fishmix"
Command? o
Command? n
startPos[0]=34 length=09 string="Aerobiks"
startPos[1]=24 length=10 string="Amaryllix"
startPos[2]=43 length=08 string="Fishmix"
startPos[3]=14 length=10 string="Kostunrix"
startPos[4]=00 length=14 string="Ordalfabetrix"
Command? q
Bye!
```

Hint: Again we see the advantage of decoupling the storing of the names in the `names` array and keeping track of the starting positions in the `startPos` array. We do not touch the data in the `names` array but only sort the start positions.

Task 11. This is an optional task. You can receive a passing grade without this task. As you will have noticed, removing a name doesn't delete the name but only deletes the entry from the index (the `startPos` array). Just like software to recover deleted files from a hard disk, we can recover these removed names. Implement the 'v' (recoVer) command with the function `void recoverNames(char names[], char *startPos[], int *nrNames)` that runs through the `names` array and checks if each name is in the `startPos` array. If not, it adds the name. The function prints out each name it finds and whether it is "garbage" (i.e. deleted) or not.



```
Command? i
Name? one
Command? i
Name? two
Command? i
Name? three
Command? i
Name? ninetynine
Command? p
one\0two\0three\0ninetynine\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? n
startPos[0]=00 length=04 string="one"
startPos[1]=04 length=04 string="two"
startPos[2]=08 length=06 string="three"
startPos[3]=14 length=11 string="ninetynine"
Command? r
Name? two
Command? r
Name? three
Command? p
one\0two\0three\0ninetynine\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? n
startPos[0]=00 length=04 string="one"
startPos[1]=14 length=11 string="ninetynine"
Command? v
Name "one" starts at 0 and is not garbage
Name "two" starts at 4 and is garbage
Name "three" starts at 8 and is garbage
Name "ninetynine" starts at 14 and is not garbage
Command? p
one\0two\0three\0ninetynine\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
Command? n
startPos[0]=00 length=04 string="one"
startPos[1]=14 length=11 string="ninetynine"
startPos[2]=04 length=04 string="two"
startPos[3]=08 length=06 string="three"
Command? q
Bye!
```

Submission: Your final solution must be submitted through OnCourse which will automatically grade this submission. For this you must upload `sorting.c` on Oncourse. You can resubmit as often as you want until the deadline.

- **14/7 v0.1** Baseline.

- **14/7 v0.2** Clean up.
- **23/7 v0.3** Bye! is with newline (of course).
- **29/7 v0.4** Clarified initial copying of Asterix.
- **22/9 v0.5** Fixed minor error in hint.

Input / output test cases

Long lines have been wrapped at 70 characters for legibility. When your program output is compared to the expected output lines will not be wrapped.

Case 01

Input:

```
X
q
```

Output:

```
Command? Unknown command 'X'
Command? Bye!
```

Case 02

Input:

```
p
q
```

Output:

```
Command?
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Bye!
```

Case 03

Input:

```
p
i
Bjorn
p
i
Agneta
p
i
Benny
p
i
Frida
p
q
```

Output:

```
Command?
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Name? Command?
Bjorn\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Name? Command?
Bjorn\0Agneta\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Name? Command?
Bjorn\0Agneta\0Benny\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Name? Command?
Bjorn\0Agneta\0Benny\0Frida\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Bye!
```

Case 04

Input:

```
p
n
i
Bjorn
p
n
i
Agneta
p
n
i
Benny
p
n
i
Frida
p
n
i
AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBAAAAAAAAAAAAAAAA
AA!!
p
n
q
```

Output:

```
Command?
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? Command? Name? Command?
Bjorn\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="Bjorn"
Command? Name? Command?
Bjorn\0Agneta\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="Bjorn"
startPos[1]=06 length=07 string="Agneta"
Command? Name? Command?
Bjorn\0Agneta\0Benny\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="Bjorn"
startPos[1]=06 length=07 string="Agneta"
startPos[2]=13 length=06 string="Benny"
Command? Name? Command?
Bjorn\0Agneta\0Benny\0Frida\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="Bjorn"
startPos[1]=06 length=07 string="Agneta"
startPos[2]=13 length=06 string="Benny"
startPos[3]=19 length=06 string="Frida"
Command? Name? Command?
Bjorn\0Agneta\0Benny\0Frida\0AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBAAAAAAAAAAAAAAAAAA!!\0
Command? startPos[0]=00 length=06 string="Bjorn"
startPos[1]=06 length=07 string="Agneta"
startPos[2]=13 length=06 string="Benny"
startPos[3]=19 length=06 string="Frida"
startPos[4]=25 length=75
string="AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBAAAAAAAA
AAAAAAAAAA!!"
Command? Bye!
```

Case 05

Input:

```
e
i
Bjorn
e
i
Agneta
e
i
Benny
e
i
Frida
i
AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBAAAAAAAAAAAAAAAA
AA!!
p
n
e
q
```

Output:

```
Command? Total length is 0
Command? Name? Command? Total length is 6
Command? Name? Command? Total length is 13
Command? Name? Command? Total length is 19
Command? Name? Command? Name? Command?
Bjorn\0Agneta\0Benny\0Frida\0AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBB
BBBBBBBBBBBBBAAAAAAAAAAAAAAAAAA!!\0
Command? startPos[0]=00 length=06 string="Bjorn"
startPos[1]=06 length=07 string="Agneta"
startPos[2]=13 length=06 string="Benny"
startPos[3]=19 length=06 string="Frida"
startPos[4]=25 length=75
string="AAAAAAAAAAAAAAAAAABBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBAAAAAAAA
AAAAAAAAAA!!"
Command? Total length is 100
Command? Bye!
```

Case 06

Input:

```
i
Tarzan
i
Jane
i
Jill
i
Jill
i
Jane
i
Hulk
p
n
l
Tarzan
l
Jane
l
Jill
l
Hulk
l
Spiderman
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command? Name? Command?
Tarzan\0Jane\0Jill\0Jill\0Jane\0Hulk\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=07 string="Tarzan"
startPos[1]=07 length=05 string="Jane"
startPos[2]=12 length=05 string="Jill"
startPos[3]=17 length=05 string="Jill"
startPos[4]=22 length=05 string="Jane"
startPos[5]=27 length=05 string="Hulk"
Command? Name? "Tarzan" has index 0
Command? Name? "Jane" has index 22
Command? Name? "Jill" has index 17
Command? Name? "Hulk" has index 27
Command? Name? "Spiderman" has index -1
Command? Bye!
```

Case 07

Input:

```
r
Who
i
ABBA
i
Beegees
i
50cents
i
1dollar
i
MinniVanilli
i
Who
p
n
e
l
Who
r
Who
p
n
e
l
Who
r
1dollar
p
n
e
r
1dollar
p
n
e
i
Beegees
i
ABBA
p
n
e
l
Beegees
r
Beegees
p
n
e
l
Beegees
r
Beegees
p
n
e
l
Beegees
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command? Name? Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=05 length=08 string="Beegees"
startPos[2]=13 length=08 string="50cents"
startPos[3]=21 length=08 string="1dollar"
startPos[4]=29 length=13 string="MinniVanilli"
startPos[5]=42 length=04 string="Who"
Command? Total length is 46
Command? Name? "Who" has index 42
Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=05 length=08 string="Beegees"
startPos[2]=13 length=08 string="50cents"
startPos[3]=21 length=08 string="1dollar"
startPos[4]=29 length=13 string="MinniVanilli"
Command? Total length is 42
Command? Name? "Who" has index -1
Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=05 length=08 string="Beegees"
startPos[2]=13 length=08 string="50cents"
startPos[3]=29 length=13 string="MinniVanilli"
Command? Total length is 34
Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=05 length=08 string="Beegees"
startPos[2]=13 length=08 string="50cents"
startPos[3]=29 length=13 string="MinniVanilli"
Command? Total length is 34
Command? Name? Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0Beegees\0ABBA\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=05 length=08 string="Beegees"
startPos[2]=13 length=08 string="50cents"
startPos[3]=29 length=13 string="MinniVanilli"
startPos[4]=46 length=08 string="Beegees"
startPos[5]=54 length=05 string="ABBA"
Command? Total length is 47
Command? Name? "Beegees" has index 46
Command? Name? Command?
ABBA\0Beegees\050cents\01dollar\0MinniVanilli\0Who\0Beegees\0ABBA\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0
Command? startPos[0]=00 length=05 string="ABBA"
startPos[1]=54 length=05 string="ABBA"
startPos[2]=13 length=08 string="50cents"
startPos[3]=29 length=13 string="MinniVanilli"
startPos[4]=46 length=08 string="Beegees"
```

Case 08

Input:

```
i
I
i
like
i
writing
i
C
i
code
p
n
e
l
code
s
2 3
p
n
e
s
3 4
p
n
e
l
code
s
0 4
n
s
0 0
n
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=07 length=08 string="writing"
startPos[3]=15 length=02 string="C"
startPos[4]=17 length=05 string="code"
Command? Total length is 22
Command? Name? "code" has index 17
Command? Indices? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=07 length=08 string="writing"
startPos[4]=17 length=05 string="code"
Command? Total length is 22
Command? Indices? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=07 length=08 string="writing"
Command? Total length is 22
Command? Name? "code" has index 17
Command? Indices? Command? startPos[0]=07 length=08 string="writing"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=00 length=02 string="I"
Command? Indices? Command? startPos[0]=07 length=08 string="writing"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=00 length=02 string="I"
Command? Bye!
```

Case 09

Input:

```
i
I
i
like
i
writing
i
C
i
code
p
n
e
l
code
s
2 3
p
n
e
s
3 4
p
n
e
l
code
s
0 4
n
s
0 0
n
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=07 length=08 string="writing"
startPos[3]=15 length=02 string="C"
startPos[4]=17 length=05 string="code"
Command? Total length is 22
Command? Name? "code" has index 17
Command? Indices? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=07 length=08 string="writing"
startPos[4]=17 length=05 string="code"
Command? Total length is 22
Command? Indices? Command?
I\0like\0writing\0C\0code\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\
0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=02 string="I"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=07 length=08 string="writing"
Command? Total length is 22
Command? Name? "code" has index 17
Command? Indices? Command? startPos[0]=07 length=08 string="writing"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=00 length=02 string="I"
Command? Indices? Command? startPos[0]=07 length=08 string="writing"
startPos[1]=02 length=05 string="like"
startPos[2]=15 length=02 string="C"
startPos[3]=17 length=05 string="code"
startPos[4]=00 length=02 string="I"
Command? Bye!
```

Case 10

Input:

```
i
5five
i
7seven
i
4four
i
1one
i
6six
i
2two
i
3three
p
n
e
o
p
n
e
i
8eight
s
0 7
n
o
n
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command? Name? Command? Name? Command?
5five\07seven\04four\01one\06six\02two\03three\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="5five"
startPos[1]=06 length=07 string="7seven"
startPos[2]=13 length=06 string="4four"
startPos[3]=19 length=05 string="1one"
startPos[4]=24 length=05 string="6six"
startPos[5]=29 length=05 string="2two"
startPos[6]=34 length=07 string="3three"
Command? Total length is 41
Command? Command?
5five\07seven\04four\01one\06six\02two\03three\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=19 length=05 string="1one"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
Command? Total length is 41
Command? Name? Command? Indices? Command? startPos[0]=41 length=07
string="8eight"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
startPos[7]=19 length=05 string="1one"
Command? Command? startPos[0]=19 length=05 string="1one"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
startPos[7]=41 length=07 string="8eight"
Command? Bye!
```

Case 11

Input:

```
i
5five
i
7seven
i
4four
i
1one
i
6six
i
2two
i
3three
p
n
e
o
p
n
e
i
8eight
s
0 7
n
o
n
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command? Name? Command? Name? Command?
5five\07seven\04four\01one\06six\02two\03three\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="5five"
startPos[1]=06 length=07 string="7seven"
startPos[2]=13 length=06 string="4four"
startPos[3]=19 length=05 string="1one"
startPos[4]=24 length=05 string="6six"
startPos[5]=29 length=05 string="2two"
startPos[6]=34 length=07 string="3three"
Command? Total length is 41
Command? Command?
5five\07seven\04four\01one\06six\02two\03three\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=19 length=05 string="1one"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
Command? Total length is 41
Command? Name? Command? Indices? Command? startPos[0]=41 length=07
string="8eight"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
startPos[7]=19 length=05 string="1one"
Command? Command? startPos[0]=19 length=05 string="1one"
startPos[1]=29 length=05 string="2two"
startPos[2]=34 length=07 string="3three"
startPos[3]=13 length=06 string="4four"
startPos[4]=00 length=06 string="5five"
startPos[5]=24 length=05 string="6six"
startPos[6]=06 length=07 string="7seven"
startPos[7]=41 length=07 string="8eight"
Command? Bye!
```

Case 12

Input:

```
i
5five
i
7seven
i
4four
i
1one
i
6six
i
2two
i
3three
i
8eight
p
n
l
1one
l
2two
l
4four
r
5five
r
1one
r
2two
n
o
l
2two
r
3three
n
v
p
n
e
l
1one
l
2two
l
4four
q
```

Output:

```
Command? Name? Command? Name? Command? Name? Command? Name? Command?
Name? Command? Name? Command? Name? Command? Name? Command?
5five\07seven\04four\01one\06six\02two\03three\08eight\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=00 length=06 string="5five"
startPos[1]=06 length=07 string="7seven"
startPos[2]=13 length=06 string="4four"
startPos[3]=19 length=05 string="1one"
startPos[4]=24 length=05 string="6six"
startPos[5]=29 length=05 string="2two"
startPos[6]=34 length=07 string="3three"
startPos[7]=41 length=07 string="8eight"
Command? Name? "1one" has index 19
Command? Name? "2two" has index 29
Command? Name? "4four" has index 13
Command? Name? Command? Name? Command? Name? Command? startPos[0]=41
length=07 string="8eight"
startPos[1]=06 length=07 string="7seven"
startPos[2]=13 length=06 string="4four"
startPos[3]=34 length=07 string="3three"
startPos[4]=24 length=05 string="6six"
Command? Command? Name? "2two" has index -1
Command? Name? Command? startPos[0]=41 length=07 string="8eight"
startPos[1]=13 length=06 string="4four"
startPos[2]=24 length=05 string="6six"
startPos[3]=06 length=07 string="7seven"
Command? Name "5five" starts at 0 and is garbage
Name "7seven" starts at 6 and is not garbage
Name "4four" starts at 13 and is not garbage
Name "1one" starts at 19 and is garbage
Name "6six" starts at 24 and is not garbage
Name "2two" starts at 29 and is garbage
Name "3three" starts at 34 and is garbage
Name "8eight" starts at 41 and is not garbage
Command?
5five\07seven\04four\01one\06six\02two\03three\08eight\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0
\0\0\0\0\0\0\0\0\0\0
Command? startPos[0]=41 length=07 string="8eight"
startPos[1]=13 length=06 string="4four"
startPos[2]=24 length=05 string="6six"
startPos[3]=06 length=07 string="7seven"
startPos[4]=00 length=06 string="5five"
startPos[5]=19 length=05 string="1one"
startPos[6]=29 length=05 string="2two"
startPos[7]=34 length=07 string="3three"
Command? Total length is 48
Command? Name? "1one" has index 19
Command? Name? "2two" has index 29
Command? Name? "4four" has index 13
Command? Bye!
```