Les petits plats

Functionality Investigation Sheet

Feature: Global search bar	**Feature #2**
Problem: To be able to filter recipes using the "global search bar" using	

Option 1: Linear search
In this option, the linear search algorithm is used to search for the keyword in the array of keywords of each recipe.
Linear Search: Search an array starting from the leftmost element and one by one comparing the searched value with each element of the array. If a match is found return a Boolean true, If no match is found with any element, return a Boolean false.

Benefits	Disadvantages
• No specific array pre-treatment • Relatively efficient on small arrays	• The bigger the array, the slower the search • Access data sequentially

Equality comparison
Time complexity: -O(n)

Option 2: Binary search
In this option, the binary search algorithm is used to search for the keyword in the sorted array of keywords of each recipe.
Binary Search: Search a sorted array by repeatedly dividing the search interval in half. Begin with an interval covering the whole array. If the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half. Otherwise narrow it to the upper half. Repeatedly check until the value is found or the interval is empty.

Benefits	Disadvantages
• Efficient on big array	• Array needs to be sorted • Access data randomly

Ordering comparison
Time complexity: -O(log n)

Option 3: Interpolation search
In this option, the interpolation search algorithm is used to search for the keyword in the sorted array of keywords of each recipe.
Interpolation Search: it is an improvement over Binary Search for instances, where the values in a sorted array are uniformly distributed. Binary Search always goes to the middle element to check. On the other hand, interpolation search may go to different locations according to the value of the key being searched. For example, if the value of the key is closer to the last element, interpolation search is likely to start search toward the end side.

Benefits	Disadvantages
• Very efficient on uniform big array	• Array needs to be sorted • Better on uniformly distributed array • Not made for string search

Ordering comparison
Time complexity: -log(log n) up to O(n) (depends on the uniformity of the distribution)

Solution retention:
As said in the disadvantages of the 3rd option, the interpolation search is not made for a string search, to be able to still use it, it is necessary to add a function to convert the strings in numerical value (with many different possible implementation, more or less complex, and more or less working) therefor adding quite some processing time.

Regarding the comparison between option 1 & 2:
Theory wise:
On the case of the option 1, if the array is not sorted, then the processing time cannot be guessed, if it is sorted, then the 'lower' the searched value, the faster the search will be. In any case, at best, the searched word is found instantly, in worst case, the searched is found on the very last iteration.
On the case of the option 2: each iteration of the algorithm, divides by 2 the amount of word in the array. Therefore, at best, the searched word is found on the first iteration, and at worst on the log2(N), N = 50 in our case, so on the ~6th iteration.

In practice:
After execution time testing, if we look at the table Figure1. We can see that the binary search is almost 2 times faster than the linear one. Moreover, the binary search also offers an execution time 2 times more stable than the linear one.

It is thus the binary search that has been chosen for the search algorithm.

Keyword	Execution time (in ms)	
	Linear Search	Binary Search
app	0.316894531	0.248779297
appl	0.201171875	0.128173828
apple	0.178955078	0.126220703
egg	0.213134766	0.116943359
lem	0.227783203	0.124023438
lemo	0.206054688	0.115966797
lemon	0.187988281	0.123046875
mas	0.217041016	0.133056641
mash	0.197021484	0.108154297
mashe	0.180175781	0.105957031
mashed	0.159912109	0.126953125
tun	0.225097656	0.136230469
tuna	0.260253906	0.136230469
whi	0.212890625	0.127929688
whit	0.400878906	0.125
white	0.115722656	0.129882813
average	0.218811035	0.132034302
diff max-min	0.28515625	0.142822266

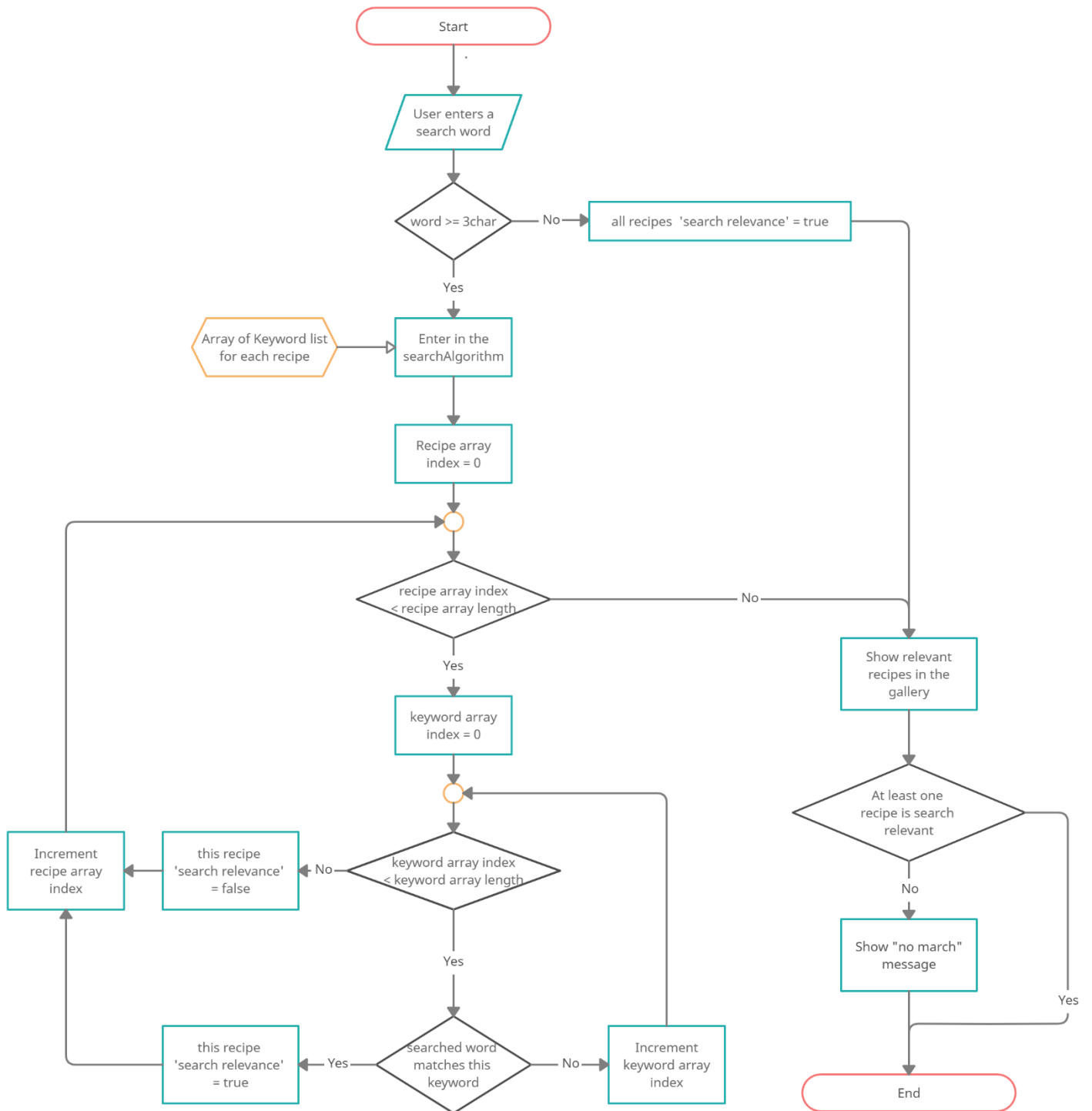*Figure 1 : Comparative table of Linear / Binary search algorithm's execution time*

Annexes

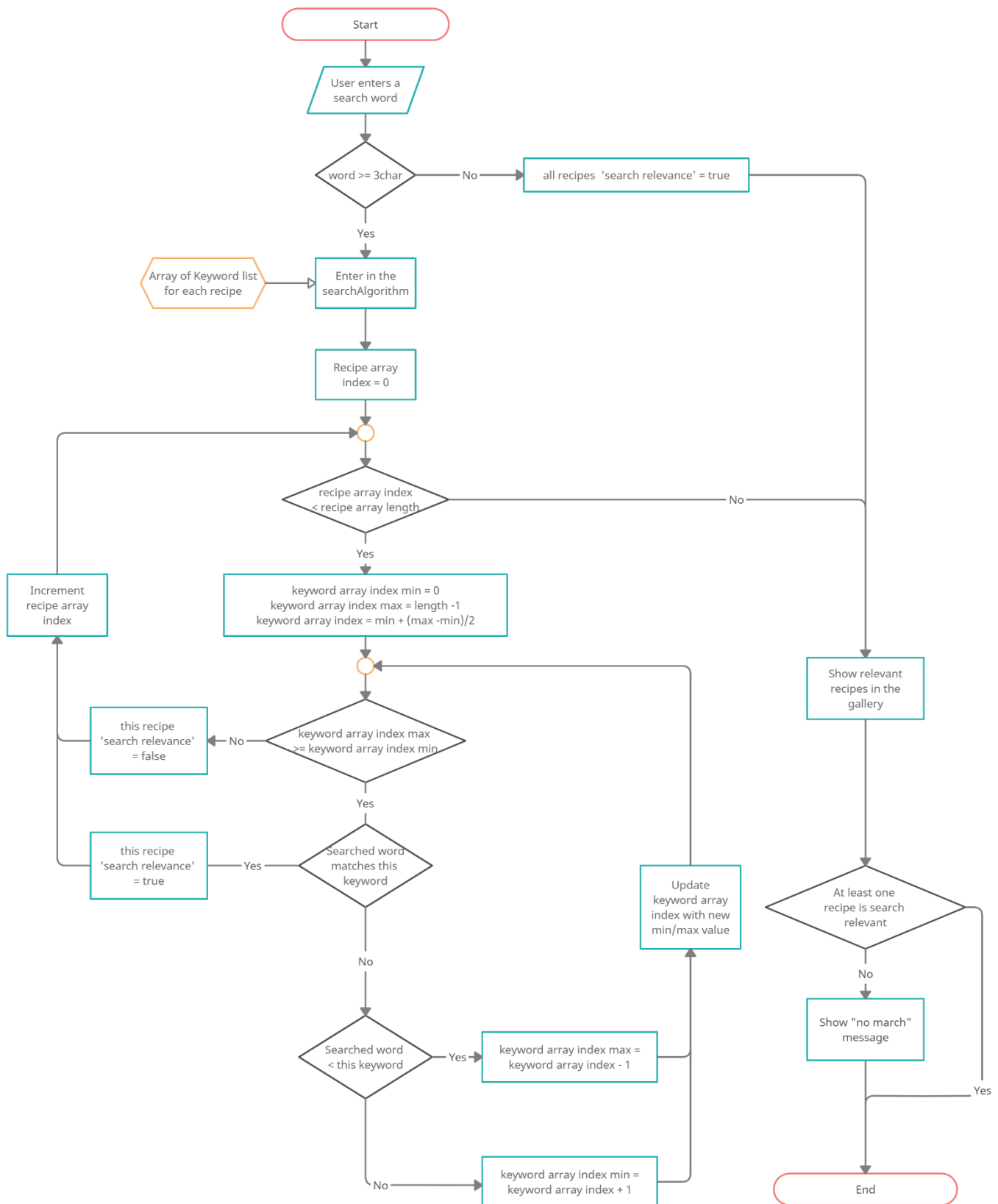
Figure 2: Workflow Diagram Linear Search Algorithm

Les petits plats

Figure 3: Workflow Diagram Binary Search Algorithm

Les petits plats

Start

User enters a
search word

*Formula:
arr[] ==> Array where elements need to be
searched
x   ==> Element to be searched
lo   ==> Starting index in arr[]
hi   ==> Ending index in arr[]
pos = lo + (x - arr[lo]) * (hi -lo)/(arr[hi] - arr[lo]

word >= 3char

No

all recipes 'search relevance' = true

Yes

Array of Keyword list
for each recipe

Enter in the
searchAlgorithm

Recipe array
index = 0

recipe array index
< recipe array length

No

Yes

Increment
recipe array
index

keyword array index min = 0
keyword array index max = length -1
keyword array index = formula*

Show relevant
recipes in the
gallery

keyword array index min ==
keyword array index max

Update
keyword array
index with new
min/max value

Yes

this recipe
'search relevance'
= true

Yes

Searched word
matches keyword at
min/max index

No

At least one
recipe is search
relevant

Yes

No

No

Searched word
matches this
keyword

No

Show "no march"
message

No

Searched word
< this keyword

Yes

keyword array index max =
keyword array index - 1

Yes

this recipe
'search relevance'
= false

No

Searched word
> this keyword

Yes
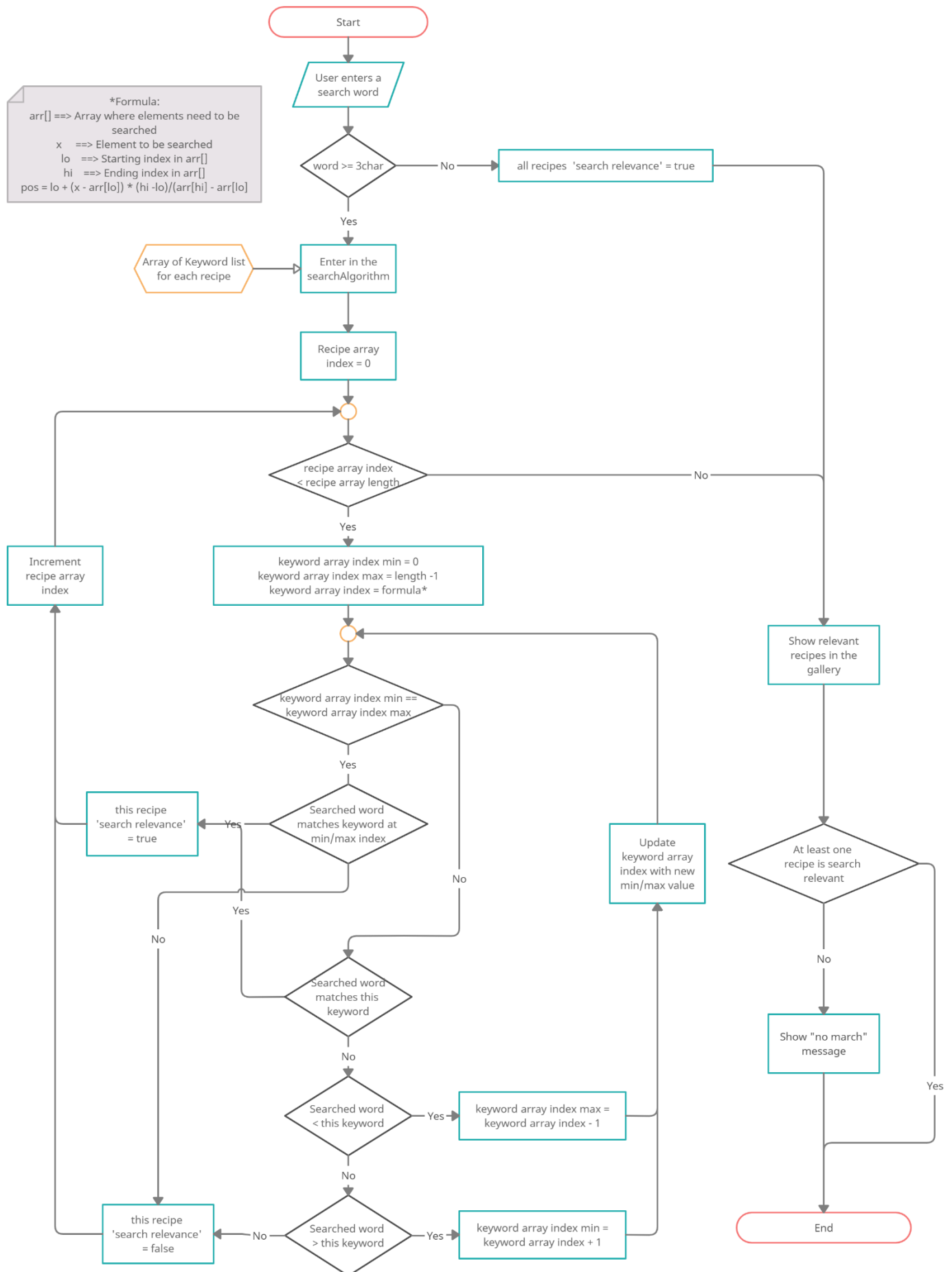
keyword array index min =
keyword array index + 1

End

Figure 4: Workflow Diagram Interpolation Search Algorithm