RANDOM TEXT GENERATION (413/413 points)

The goal of this project is to synthesize natural language sentences using information extracted from an existing text corpus.

For this, given a text corpus as input, we will first compute the frequency of all sequences of two words in the original text; then we will use this information to produce new sentences by randomly collating these sequences with the same frequencies.

This method is known under the term of *Markov chain*. From the input text, we compute a transition table that associates to each word the list of words that may appear after it, with their relative frequencies.

For instance, if we examine *"I am a man and my dog is a good dog and a good dog makes a good man"*, delimiting it with `"START"` and `"STOP"` to identify the beginning and end of the sentence, we end up with the transition table on the right.

word	→	next	freq
"START"	→	"I"	100%
"I"	→	"am"	100%
"am"	→	"a"	100%
"a"	→	"man"	25%
	|	"good"	75%
"man"	→	"and"	50%
	|	"STOP"	50%
"and"	→	"my"	50%
	|	"a"	50%
"my"	→	"dog"	100%
"dog"	→	"is"	33%
	|	"and"	33%
	|	"makes"	34%
"good"	→	"dog"	66%
	|	"man"	34%
"is"	→	"a"	100%
"makes"	→	"a"	100%

This table can then be used to generate new text that ressembles the input in the following way: starting from the `"START"` word, choose one of the words that may appear after it, with the probability found in the table, add it to the output, then iterate the process until the `"STOP"` word is found. Below are some example sentences produced using this table.

```
START I am a good  man STOP ; START I am a good dog  is a good dog and
my dog and my  dog is a man and my  dog and a man STOP ;  START I am a
good dog is a man and my dog makes a good man STOP ; START I am a good
dog makes a good dog is a good dog  and a good dog makes a good dog is
a man STOP ; START I am a good dog and a man and a good dog and a good
man and a good dog is a good dog  is a good man and a man STOP ; START
I am a good dog and a good dog and my dog is a man STOP ; START I am a
man STOP ;  START I am a good dog  is a good dog is a  good dog and my
dog is a man STOP ; START I am a good man STOP ; START I am a good dog
makes a good dog and a good dog is a good dog is a good man and my dog
is a good dog and  my dog and a good man and a good  dog is a good man
STOP ; START I am a man and my dog and my dog is a good dog and a good
dog makes a man STOP
```

This project is decomposed in three parts.

A. First, we will build a quick prototype, that goes from an input sentence to a randomly generated sentences via a distribution table as the ones above.

B. Then we will use better data structures to enhance the performance so that we can use larger texts, such as small books, as input.

C. After that, we will enhance the quality of input and output, by analysing in a smarter way the input text corpus, and by considering sequences of more than two words.

Note: this project may take more time to be graded, because it is longer than simple exercises, and because it is tested on large inputs. We suggest that you use the typecheck button and the toplevel extensively, so that you are reasonnably sure of your code before submitting it to the tests. Also, we added a function `grade_only : int list -> unit`, that you may call in your code to select the exercises to grade. All other exercises won't be graded at all, and considered failed. For instance, if you write `grade_only [ 3 ] ;;` at the beginning of your file, only exercise 3 will be tested.

PART A: A FIRST DRAFT

Our first goal will be to build such a table and generate sentences from it, quick and dirty style, using lists and their predefined operators. Consider using as much as possible the `List` module (`List.assoc`, `List.length`, `List.nth`, etc.) and don't think about efficiency.

they are very easy to use and debug. Their major downfall is the complexity of searching for an element.
The type of an associative list that maps `string` keys to `'a` values is simply `(string * 'a) list`. The value associated with a key `"x"` is simply the right component of the first pair in the list whose left component is `"x"`. This lookup is already defined in the standard library as `List.assoc`. Hence, setting the value of `"x"` to `3`, for instance, is just adding `("x",3)` in front of the list. To remove an element, you can just use `List.filter` with the right predicate.

The type of lookup tables for this exercise is

```
type ltable = (string * string list) list
```

1. Write a function `words : string -> string list` that takes a sentence and returns the list of its words. As a first approximation, will work on single sentences in simple english, so you can consider sequences of roman letters and digits as words, and everything else as separators. If you want to build words bit by bit, you can experiment with the Buffer module. Beware, this preliminary function may not be as easy as it seems.
2. Write `build_ltable : string list -> ltable` that builds an associative list mapping each word present in the input text to all its possible successors (including duplicates). The table should also contain `"START"` that points to the first word and `"STOP"` that is pointed by the last word.
   For instance, a correct (and minimal) table for `"x y z y x y"` looks like:

   ```
   [ ("z", [ "y" ]);
     ("x", [ "y" ; "y" ]);
     ("START", [ "x" ]);
     ("y", [ "x" ; "z" ; "STOP" ]) ]
   ```
3. Write the random selection function `next_in_ltable : (string * string list) list -> string - > string` which takes a table, a given word and returns a valid successor of this word. Your function should respect the probability distribution (which should be trivially ensured by the presence of the duplicates in the successor lists).
4. Write the random generation function `walk_ltable : (string * string list) list -> string list` which takes a table and returns a sequence of words that form a valid random sentence (without the `"START"` and `"STOP"`).

You can use `display_quote: string list -> unit` to display the generated texts.

PART B: PERFORMANCE IMPROVEMENTS

Now, we want to use more efficient data structures, so that we can take larger inputs and build bigger transition tables.

In this exercise, we will use hash tables, predefined in OCaml in the Hashtbl module. Used correctly, hash table provide both fast insertion and extraction. Have a look at the documentation of the module. In particular, don't miss the difference between `Hashtbl.add` and `Hashtbl.replace` (you'll probably want to use the latter most of the time).

The types for this exercise are:

```
type distribution =
  { total : int ;
    amounts : (string * int) list }
type htable = (string, distribution) Hashtbl.t
```

5. In the simple version, we stored for each word the complete list of suffixes, including duplicates. This is a valid data structure to use when building the table since adding a new suffix in front of the list is fast. But when generating, it means computing the length of this list

original list and its number of occurrences.
For instance, `compute_distribution ["a";"b";"c";"b";"c";"a";"b";"c";"c";"c"]` should give `{ total = 10 ; amounts = [("c", 5); ("b", 3); ("a", 2)] }`.
Hint: a first step that simplifies the problem is to sort the list.

6. Write a new version of `build_htable : string list -> htable` that creates a hash table instead of an associative list, so that both table building and sentence generation will be faster. Like the associative list, the table is indexed by the words, each word being associated to its successors. But instead of just storing the list of successors, it will use the format of the previous question.
Hint: You can first define an intermediate table of type `(string, string list) Hashtbl.t` that stores the lists of successors with duplicates. Then you traverse this intermediate table with `Hashtbl.iter`, and for each word, you add the result of `compute_distribution` in the final table.
7. Define `next_in_htable : htable -> string -> string` that does the same thing as `next_in_ltable` for the new table format.
8. Finally, define `walk_htable : htable -> string list`

PART C: QUALITY IMPROVEMENTS

9. If we want to generate sentences from larger corpuses, such as the ones of the `ebooks_corpus` given in the prelude, we cannot just ignore the punctuation. We also want to generate text using not only the beginning of the original text, but the start of any sentence in the text.
Define `sentences : string -> string list list` that splits a string into a list of sentences such as:
    - uninterrupted sequences of roman letters, numbers, and non ASCII characters (in the range `'\128'..'\255'`) are words;
    - single punctuation characters `';'`, `','`, `':'`, `'-'`, `'"'`, `'\''`, `'?'`, `'!'` and `'.'` are words;
    - punctuation characters `'?'`, `'!'` and `'.'` terminate sentences;
    - everything else is a separator;
    - and your function should not return any empty sentence.

Now, we will drastically improve the results by matching sequences of more than two words. We will thus update the format of our tables again, and use the following `ptable` type (which looks a lot like the previous one).

```
type ptable =
  { prefix_length : int ;
    table : (string list, distribution) Hashtbl.t }
```

So let's say we want to identify sequences of N words in the text. The `prefix_length` field contains $N-1$. The `table` field associates each list of $N-1$ words from the text with the distribution of its possible successors.

The table on the right gives the lookup table for the example given at the beginning of the project: *"I am a man and my dog is a good dog and a good dog makes a good man"*, and a size of 2. You can see that the branching points are fewer and make a bit more sense.
As you can see, we will use `"STOP"` as an end marker as before. But instead of a single `"START"` we will use as a start marker a prefix of the same size as the others, filled with `"START"`.

10. First, define `start: int -> string list` that makes the start prefix for a given size (`start 0 = []`, `start 1 = [ "START" ]`, `start 2 = [ "START" ; "START" ]`, etc.).

prefix	→	next	freq
["START"; "START"]	→	"I"	100%
["START"; "I"]	→	"am"	100%
["I"; "am"]	→	"a"	100%
["am"; "a"]	→	"man"	100%
["man"; "and"]	→	"my"	100%
["is"; "a"]	→	"good"	100%
["and"; "my"]	→	"dog"	100%
["my"; "dog"]	→	"is"	100%
["makes"; "a"]	→	"good"	100%
["a"; "good"]	→	"man"	33%
	|	"dog"	66%
["dog"; "is"]	→	"a"	100%
["and"; "a"]	→	"good"	100%
["good"; "dog"]	→	"makes"	50%

new element at the end.

["good"; "man"] → "STOP"	100%
["dog"; "makes"] → "a"	100%

(`shift [ "A" ; "B" ; "C" ] "D" = [ "B" ; "C" ; "D" ]` , `shift [ "B" ; "C" ; "D" ] "E" = [ "C" ; "D" ; "E" ]` , etc.).

12. Define `build_ptable : string list -> int -> ptable` that builds a table for a given prefix length, using the two previous functions.
13. Define `walk_ptable : ptable -> string list` that generates a sentence from a given `ptable` . Unless you put specific annotations, `next_in_htable` should be polymorphic enough to work on the field `table` of a `ptable` , so you don't have to rewrite one. If you want, since we now have proper sentence splitting, you can generate multi-sentence texts, by choosing randomly to continue from the start after encountering a `"STOP"` .

Finally, the most funny texts are generated when mixing various kinds of inputs together (pirate stories, history books, recipes, political news, etc.).

14. Define `merge_ptables: ptable list -> ptable` that combines several tables together (you may fail with an exception if the prefix sizes are inconsistent).

Now you can try and generate some texts using larger inputs, such as short novels! The prelude provides a few samples, otherwise Project Gutemberg is a good source. You can use `display_quote: string list -> unit` to display the generated texts.

```
let sauce_ptable =
   merge_ptables
     (List.map
        (fun s -> build_ptable s 2)
        (sentences some_cookbook_sauce_chapter)) ;;
display_quote (walk_ptable sauce_ptable) ;;
```

THE GIVEN PRELUDE

```
    amounts : (string * int) list }
type htable = (string, distribution) Hashtbl.t
type ptable =
  { prefix_length : int ;
    table : (string list, distribution) Hashtbl.t }

let simple_0 =
  "I am a man and my dog is a good dog and a good dog makes a good man"

let simple_1 =
  "a good dad is proud of his son and a good son is proud of his dad"

let simple_2 =
  "a good woman is proud of her daughter and a good daughter is proud of her mom"

let simple_3 =
  "there is a beer in a fridge in a kitchen in a house in a land where \
   there is a man who has a house where there is no beer in the kitchen"

let multi_1 =
  "A good dad is proud of his son. \
   A good son is proud of his dad."

let multi_2 =
  "A good woman is proud of her daughter. \
   A good daughter is proud of her mom."

let multi_3 =
  "In a land of myths, and a time of magic, \
   the destiny of a great kingdom rests \
   on the shoulders of a young man."

let grimms_travelling_musicians =
  "An honest farmer had once an ass that had been a faithful servant ..."

let grimms_cat_and_mouse_in_partnership =
  "A certain cat had made the acquaintance of a mouse, and ..."

let the_war_of_the_worlds_chapter_one =
  "No one would have believed in the last years ..."

let some_cookbook_sauce_chapter =
  "Wine Chaudeau: Into a lined saucepan put ½ bottle Rhine ..."

let history_of_ocaml =
  "“Caml” was originally an acronym for Categorical ..."
```

YOUR OCAML ENVIRONMENT

```
(* -- Part A ------------------------------------------------------------- *)

let words str =
  let rec rec_split str liste mot =
    if str = "" then liste@[mot] else
    if str.[0] = ' ' then
      rec_split (String.sub str 1 (String.length str - 1)) (liste@[mot]) "" else
      rec_split (String.sub str 1 (String.length str - 1)) liste (mot^(String.make 1 str.[0]))
  in
  rec_split str [] ""
;;

let build_ltable words =
  let first = List.hd words in
  (*first make a table with each word and his successor*)
  let rec build_rec liste ltab = match liste with
    | [] -> []
    | [w] -> ltab@[(w,["STOP"])]
    | hd1::hd2::tl ->
        build_rec (hd2::tl) (ltab)@[(hd1,[hd2])]
  in
  let liste_all = ("START", [first]) :: (List.rev (build_rec words []))
  in
  (* compact the list with same keys*)
  let rec compact liste_init liste_fin = match liste_init with
    | [] -> liste_fin
    | hd::tl -> match hd with
      | (e, l) -> let result = try List.assoc e tl with _ -> [] in
          if result = [] then compact tl (hd::liste_fin) else
            compact ((e, result@l)::(List.remove_assoc e tl)) liste_fin
  in
  compact liste_all []
```

Evaluate >>

Switch >>

Typecheck

Reset Template

Full-screen [+]

Check & Save

v Exercise 1: words Completed, 50 pts

Found words with compatible type.

```
Computing
  words
    "a good woman is proud of her daughter and a good daughter is proud of her mom"
Text splitted as expected.                                                     5 pts
Computing
  words
    "there is a beer in a fridge in a kitchen in a house in a land where there is a man who
Text splitted as expected.                                                     5 pts
Computing words "a good dad is proud of his son and a good son is proud of his dad"
Text splitted as expected.                                                     5 pts
Computing
  words
    "a good woman is proud of her daughter and a good daughter is proud of her mom"
Text splitted as expected.                                                     5 pts
Computing words "a good dad is proud of his son and a good son is proud of his dad"
Text splitted as expected.                                                     5 pts
Computing
  words
    "there is a beer in a fridge in a kitchen in a house in a land where there is a man who
Text splitted as expected.                                                     5 pts
Computing words "a good dad is proud of his son and a good son is proud of his dad"
Text splitted as expected.                                                     5 pts
Computing
  words
    "there is a beer in a fridge in a kitchen in a house in a land where there is a man who
Text splitted as expected.                                                     5 pts
Computing words "a good dad is proud of his son and a good son is proud of his dad"
Text splitted as expected.                                                     5 pts
Computing
  words
    "a good woman is proud of her daughter and a good daughter is proud of her mom"
Text splitted as expected.                                                     5 pts
```

v Exercise 2: build_ltable Completed, 50 pts

Found build_ltable with compatible type.

```
Computing
  build_ltable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
Expected table                                                                 5 pts
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("dad", ["STOP"; "is"]); ("good", ["dad"; "son"]); ("his", ["dad"; "son"]);
   ("is", ["proud"; "proud"]); ("of", ["his"; "his"]); ("proud", ["of"; "of"]);
   ("son", ["and"; "is"])]
Computing
  build_ltable
    ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
     "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
Expected table                                                                 5 pts
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("daughter", ["and"; "is"]); ("good", ["daughter"; "woman"]);
   ("her", ["daughter"; "mom"]); ("is", ["proud"; "proud"]); ("mom", ["STOP"]);
   ("of", ["her"; "her"]); ("proud", ["of"; "of"]); ("woman", ["is"])]
Computing
  build_ltable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
     "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
     "the"; "kitchen"]
Expected table                                                                 5 pts
  [("START", ["there"]);
   ("a", ["beer"; "fridge"; "house"; "house"; "kitchen"; "land"; "man"]);
   ("beer", ["in"; "in"]); ("fridge", ["in"]); ("has", ["a"]);
   ("house", ["in"; "where"]); ("in", ["a"; "a"; "a"; "a"; "the"]);
   ("is", ["a"; "a"; "no"]); ("kitchen", ["STOP"; "in"]); ("land", ["where"]);
   ("man", ["who"]); ("no", ["beer"]); ("the", ["kitchen"]);
   ("there", ["is"; "is"; "is"]); ("where", ["there"; "there"]);
   ("who", ["has"])]
Computing
  build_ltable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
Expected table                                                                 5 pts
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("dad", ["STOP"; "is"]); ("good", ["dad"; "son"]); ("his", ["dad"; "son"]);
   ("is", ["proud"; "proud"]); ("of", ["his"; "his"]); ("proud", ["of"; "of"]);
   ("son", ["and"; "is"])]
Computing
  build_ltable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
   ("a", ["beer"; "fridge"; "house"; "house"; "kitchen"; "land"; "man"]);
   ("beer", ["in"; "in"]); ("fridge", ["in"]); ("has", ["a"]);
   ("house", ["in"; "where"]); ("in", ["a"; "a"; "a"; "a"; "the"]);
   ("is", ["a"; "a"; "no"]); ("kitchen", ["STOP"; "in"]); ("land", ["where"]);
   ("man", ["who"]); ("no", ["beer"]); ("the", ["kitchen"]);
   ("there", ["is"; "is"; "is"]); ("where", ["there"; "there"]);
   ("who", ["has"])]
```

Computing

```
  build_ltable
    ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
     "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
```

Expected table 5 pts

```
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("daughter", ["and"; "is"]); ("good", ["daughter"; "woman"]);
   ("her", ["daughter"; "mom"]); ("is", ["proud"; "proud"]); ("mom", ["STOP"]);
   ("of", ["her"; "her"]); ("proud", ["of"; "of"]); ("woman", ["is"])]
```

Computing

```
  build_ltable
    ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
     "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
```

Expected table 5 pts

```
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("daughter", ["and"; "is"]); ("good", ["daughter"; "woman"]);
   ("her", ["daughter"; "mom"]); ("is", ["proud"; "proud"]); ("mom", ["STOP"]);
   ("of", ["her"; "her"]); ("proud", ["of"; "of"]); ("woman", ["is"])]
```

Computing

```
  build_ltable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
     "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
     "the"; "kitchen"]
```

Expected table 5 pts

```
  [("START", ["there"]);
   ("a", ["beer"; "fridge"; "house"; "house"; "kitchen"; "land"; "man"]);
   ("beer", ["in"; "in"]); ("fridge", ["in"]); ("has", ["a"]);
   ("house", ["in"; "where"]); ("in", ["a"; "a"; "a"; "a"; "the"]);
   ("is", ["a"; "a"; "no"]); ("kitchen", ["STOP"; "in"]); ("land", ["where"]);
   ("man", ["who"]); ("no", ["beer"]); ("the", ["kitchen"]);
   ("there", ["is"; "is"; "is"]); ("where", ["there"; "there"]);
   ("who", ["has"])]
```

Computing

```
  build_ltable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
```

Expected table 5 pts

```
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("dad", ["STOP"; "is"]); ("good", ["dad"; "son"]); ("his", ["dad"; "son"]);
   ("is", ["proud"; "proud"]); ("of", ["his"; "his"]); ("proud", ["of"; "of"]);
   ("son", ["and"; "is"])]
```

Computing

```
  build_ltable
    ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
     "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
```

Expected table 5 pts

```
  [("START", ["a"]); ("a", ["good"; "good"]); ("and", ["a"]);
   ("daughter", ["and"; "is"]); ("good", ["daughter"; "woman"]);
   ("her", ["daughter"; "mom"]); ("is", ["proud"; "proud"]); ("mom", ["STOP"]);
   ("of", ["her"; "her"]); ("proud", ["of"; "of"]); ("woman", ["is"])]
```

v Exercise 3: next_in_ltable Completed, 12 pts

Found next_in_ltable with compatible type.

Computing

```
  next_in_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
    "in"
```

Expected word a 1 pt

Computing

```
  next_in_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
    "good"
```

Expected word daughter 1 pt

Computing

```
  next_in_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
    "proud"
```

Expected word of 1 pt

```
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
    "of"
Expected word her                                                                1 pt
Computing
  next_in_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
    "man"
Expected word who                                                                1 pt
Computing
  next_in_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
    "good"
Expected word son                                                                1 pt
Computing
  next_in_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
    "man"
Expected word who                                                                1 pt
Computing
  next_in_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
    "is"
Expected word proud                                                              1 pt
Computing
  next_in_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
    "her"
Expected word daughter                                                           1 pt
Computing
  next_in_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
    "and"
Expected word a                                                                  1 pt
Found next_in_ltable with compatible type.
Now I will check the probabilities.
Testing on "a b a c a"
Expected distribution                                                            1 pt
Expected distribution for "c"
"c" -> "a" 100%
Expected distribution for "a"
"a" -> "STOP" 33%
"a" -> "b" 33%
"a" -> "c" 32%
Expected distribution for "START"
"START" -> "a" 100%
Expected distribution for "b"
"b" -> "a" 100%
Testing on "a b a c a c a"
Expected distribution                                                            1 pt
Expected distribution for "c"
"c" -> "a" 100%
Expected distribution for "a"
"a" -> "STOP" 25%
"a" -> "b" 25%
"a" -> "c" 48%
Expected distribution for "START"
"START" -> "a" 100%
```

```
Found walk_ltable with compatible type.
Computing
  walk_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
Checking a good dad is proud of his dad is proud of his dad is proud of his son is proud of his dad
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
Checking there is a beer in the kitchen
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
Checking a good daughter is proud of her mom
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
Checking there is a man who has a house in a man who has a house in a beer in a land where there is a man who
has a man who has a man who has a house where there is a fridge in a beer in a beer in a beer in the kitchen in
the kitchen in a man who has a fridge in a house in a land where there is no beer in a beer in the kitchen
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
Checking a good son is proud of his son is proud of his son is proud of his son and a good dad
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("the", ["kitchen"]); ("has", ["a"]); ("man", ["who"]);
     ("land", ["where"]); ("kitchen", ["STOP"; "in"]);
     ("in", ["the"; "a"; "a"; "a"; "a"]);
     ("a", ["house"; "man"; "land"; "house"; "kitchen"; "fridge"; "beer"]);
     ("there", ["is"; "is"; "is"]); ("START", ["there"]);
     ("is", ["no"; "a"; "a"]); ("beer", ["in"; "in"]); ("fridge", ["in"]);
     ("house", ["where"; "in"]); ("where", ["there"; "there"]);
     ("who", ["has"]); ("no", ["beer"])]
Checking there is no beer in a man who has a house where there is no beer in a fridge in a fridge in the kitchen
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
Checking a good daughter is proud of her daughter is proud of her daughter is proud of her mom
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
     ("good", ["son"; "dad"]); ("is", ["proud"; "proud"]);
     ("of", ["his"; "his"]); ("son", ["is"; "and"])]
Checking a good son is proud of his dad
Correct sequence.                                                    1 pt
Computing
  walk_ltable
    [("and", ["a"]); ("his", ["dad"; "son"]); ("proud", ["of"; "of"]);
     ("dad", ["STOP"; "is"]); ("a", ["good"; "good"]); ("START", ["a"]);
```

Correct sequence. 1 pt

Computing
```
  walk_ltable
    [("mom", ["STOP"]); ("daughter", ["is"; "and"]); ("of", ["her"; "her"]);
     ("is", ["proud"; "proud"]); ("good", ["daughter"; "woman"]);
     ("START", ["a"]); ("a", ["good"; "good"]); ("woman", ["is"]);
     ("proud", ["of"; "of"]); ("her", ["mom"; "daughter"]); ("and", ["a"])]
```
Checking a good daughter is proud of her daughter is proud of her daughter is proud of her daughter and a good daughter and a good daughter and a good daughter and a good daughter is proud of her mom

Correct sequence. 1 pt

v Exercise 5: `compute_distribution` Completed, 50 pts

Found `compute_distribution` with compatible type.

Computing
```
  compute_distribution
    ["b"; "a"; "c"; "c"; "b"; "a"; "c"; "b"; "a"; "a"; "c"; "b"; "a"; "c"; "b";
     "a"; "b"; "c"; "b"; "c"; "a"; "c"; "a"; "b"; "c"; "b"; "a"; "a"; "c"; "b";
     "a"; "b"; "c"; "c"]
```
Expected distribution `{total = 34; amounts = [("a", 11); ("b", 11); ("c", 12)]}` 5 pts

Computing
```
  compute_distribution
    ["b"; "a"; "b"; "a"; "c"; "c"; "a"; "b"; "c"; "b"; "a"; "a"; "b"; "c"; "c";
     "a"; "b"; "b"; "c"; "a"; "b"; "a"; "c"; "a"; "c"; "b"; "c"; "a"; "b"; "c";
     "a"; "b"; "a"; "b"]
```
Expected distribution `{total = 34; amounts = [("a", 12); ("b", 12); ("c", 10)]}` 5 pts

Computing
```
  compute_distribution
    ["c"; "a"; "b"; "c"; "c"; "a"; "b"; "a"; "b"; "c"; "b"; "a"; "c"; "a"; "c";
     "b"; "c"; "b"; "a"; "a"; "b"; "c"; "a"; "c"; "b"; "b"; "c"; "a"; "a"]
```
Expected distribution `{total = 29; amounts = [("a", 10); ("b", 9); ("c", 10)]}` 5 pts

Computing
```
  compute_distribution
    ["c"; "b"; "b"; "c"; "a"; "a"; "b"; "c"; "b"; "a"; "c"; "a"; "c"; "b"; "a";
     "c"; "b"; "b"; "a"; "c"; "a"; "c"; "b"; "a"; "c"; "b"; "b"; "c"; "a"; "c";
     "a"; "b"; "c"; "b"; "a"; "a"; "c"; "b"]
```
Expected distribution `{total = 38; amounts = [("a", 12); ("b", 13); ("c", 13)]}` 5 pts

Computing
```
  compute_distribution
    ["b"; "a"; "c"; "b"; "a"; "c"; "b"; "c"; "a"; "c"; "a"; "b"; "a"; "c"; "b";
     "c"; "a"; "b"; "a"; "c"; "b"; "c"; "b"; "a"; "a"; "b"; "c"; "a"; "b"; "c";
     "a"; "b"; "c"]
```
Expected distribution `{total = 33; amounts = [("a", 11); ("b", 11); ("c", 11)]}` 5 pts

Computing
```
  compute_distribution
    ["a"; "b"; "c"; "c"; "b"; "a"; "a"; "c"; "b"; "b"; "c"; "a"; "c"; "b"; "a";
     "b"; "a"; "c"; "a"; "b"; "c"; "b"; "c"; "a"; "c"; "a"; "b"; "b"]
```
Expected distribution `{total = 28; amounts = [("a", 9); ("b", 10); ("c", 9)]}` 5 pts

Computing
```
  compute_distribution
    ["c"; "a"; "b"; "c"; "a"; "a"; "b"; "c"; "b"; "a"; "c"; "a"; "c"; "b"; "a";
     "c"; "b"; "b"; "c"; "a"]
```
Expected distribution `{total = 20; amounts = [("a", 7); ("b", 6); ("c", 7)]}` 5 pts

Computing
```
  compute_distribution
    ["c"; "b"; "a"; "a"; "c"; "b"; "c"; "b"; "a"; "b"; "a"; "c"; "a"; "c"; "b";
     "c"; "b"; "a"; "c"; "a"; "b"; "a"; "b"; "c"; "c"; "b"; "a"; "c"; "a"; "b";
     "b"; "a"; "c"; "b"; "c"; "a"]
```
Expected distribution `{total = 36; amounts = [("a", 12); ("b", 12); ("c", 12)]}` 5 pts

Computing
```
  compute_distribution
    ["a"; "b"; "c"; "b"; "c"; "a"; "b"; "a"; "c"; "c"; "a"; "b"; "c"; "b"; "a";
     "a"; "b"; "c"; "a"; "c"; "b"; "c"; "a"; "b"; "a"; "b"; "c"; "b"; "c"; "a";
     "b"]
```
Expected distribution `{total = 31; amounts = [("a", 10); ("b", 11); ("c", 10)]}` 5 pts

Computing
```
  compute_distribution
    ["a"; "c"; "a"; "c"; "b"; "c"; "b"; "a"; "b"; "a"; "c"; "c"; "a"; "b"; "c";
     "a"; "b"; "c"; "a"; "b"; "c"; "a"; "b"; "b"; "a"; "c"; "c"; "a"; "b"]
```
Expected distribution `{total = 29; amounts = [("a", 10); ("b", 9); ("c", 10)]}` 5 pts

v Exercise 6: `build_htable` Completed, 50 pts

Found `build_htable` with compatible type.

Computing
```
  build_htable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
```
Expected table 5 pts
```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table "good" {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
   Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table "son" {total = 2; amounts = [("and", 1); ("is", 1)]} ;
   Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
   Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
   Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
   Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
  build_htable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
     "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
     "the"; "kitchen"]
```

Expected table 5 pts

```
  (let table = Hashtbl.create 16 in
   Hashtbl.add table "a"
     {total = 7;
      amounts =
       [("beer", 1); ("fridge", 1); ("kitchen", 1); ("house", 2); ("land", 1);
        ("man", 1)]} ;
   Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
   Hashtbl.add table "is" {total = 3; amounts = [("a", 2); ("no", 1)]} ;
   Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
   Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
   Hashtbl.add table "in" {total = 5; amounts = [("a", 4); ("the", 1)]} ;
   Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
   Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
   Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
   Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
   Hashtbl.add table "kitchen"
     {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
   Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
   Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
   Hashtbl.add table "house" {total = 2; amounts = [("in", 1); ("where", 1)]} ;
   Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
   table)
```

Computing

```
  build_htable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
```

Expected table 5 pts

```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table "good" {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
   Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table "son" {total = 2; amounts = [("and", 1); ("is", 1)]} ;
   Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
   Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
   Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
   Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
   table)
```

Computing

```
  build_htable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
     "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
     "the"; "kitchen"]
```

Expected table 5 pts

```
  (let table = Hashtbl.create 16 in
   Hashtbl.add table "a"
     {total = 7;
      amounts =
       [("beer", 1); ("fridge", 1); ("kitchen", 1); ("house", 2); ("land", 1);
        ("man", 1)]} ;
   Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
   Hashtbl.add table "is" {total = 3; amounts = [("a", 2); ("no", 1)]} ;
   Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
   Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
   Hashtbl.add table "in" {total = 5; amounts = [("a", 4); ("the", 1)]} ;
   Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
   Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
   Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
   Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
   Hashtbl.add table "kitchen"
     {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
   Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
   Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
   Hashtbl.add table "house" {total = 2; amounts = [("in", 1); ("where", 1)]} ;
   Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
   table)
```

Computing

```
  build_htable
    ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
     "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
```

Expected table 5 pts

```
  (let table = Hashtbl.create 11 in
   Hashtbl.add table "good"
     {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table "her"
     {total = 2; amounts = [("daughter", 1); ("mom", 1)]} ;
   Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
   Hashtbl.add table "daughter"
   Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
   table)
```

Computing

```
 build_htable
   ["a"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "and"; "a";
    "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 11 in
  Hashtbl.add table "good"
    {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
  Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
  Hashtbl.add table "her"
    {total = 2; amounts = [("daughter", 1); ("mom", 1)]} ;
  Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
  Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
  Hashtbl.add table "daughter"
    {total = 2; amounts = [("and", 1); ("is", 1)]} ;
  Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
  table)
```

Computing

```
 build_htable
   ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
    "son"; "is"; "proud"; "of"; "his"; "dad"]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 10 in
  Hashtbl.add table "good" {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
  Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
  Hashtbl.add table "son" {total = 2; amounts = [("and", 1); ("is", 1)]} ;
  Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
  Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
  Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
  Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
  Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
  table)
```

Computing

```
 build_htable
   ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
    "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
    "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
    "the"; "kitchen"]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 16 in
  Hashtbl.add table "a"
    {total = 7;
     amounts =
      [("beer", 1); ("fridge", 1); ("kitchen", 1); ("house", 2); ("land", 1);
       ("man", 1)]} ;
  Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
  Hashtbl.add table "is" {total = 3; amounts = [("a", 2); ("no", 1)]} ;
  Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
  Hashtbl.add table "in" {total = 5; amounts = [("a", 4); ("the", 1)]} ;
  Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
  Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
  Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
  Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
  Hashtbl.add table "kitchen"
    {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
  Hashtbl.add table "house" {total = 2; amounts = [("in", 1); ("where", 1)]} ;
  Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
  table)
```

Computing

```
 build_htable
   ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
    "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
    "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
    "the"; "kitchen"]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 16 in
  Hashtbl.add table "a"
    {total = 7;
     amounts =
      [("beer", 1); ("fridge", 1); ("kitchen", 1); ("house", 2); ("land", 1);
       ("man", 1)]} ;
  Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
  Hashtbl.add table "is" {total = 3; amounts = [("a", 2); ("no", 1)]} ;
  Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
  Hashtbl.add table "in" {total = 5; amounts = [("a", 4); ("the", 1)]} ;
  Hashtbl.add table "kitchen"
    {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
  Hashtbl.add table "house" {total = 2; amounts = [("in", 1); ("where", 1)]} ;
  Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
  table)
```

Computing

```
  build_htable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 10 in
  Hashtbl.add table "good" {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
  Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
  Hashtbl.add table "son" {total = 2; amounts = [("and", 1); ("is", 1)]} ;
  Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
  Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
  Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
  Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
  Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
  table)
```

v Exercise 7: next_in_htable Completed, 12 pts

Found `next_in_htable` with compatible type.

Computing

```
  next_in_htable
    (let table = Hashtbl.create 11 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "daughter"
       {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good"
       {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
     Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "her"
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "a"
```

Expected word good 1 pt

Computing

```
  next_in_htable
    (let table = Hashtbl.create 16 in
     Hashtbl.add table "a"
       {total = 7;
        amounts =
         [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
     Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "is" {total = 3; amounts = [("no", 1); ("a", 2)]} ;
     Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
     Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
     Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
     Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
     Hashtbl.add table "kitchen"
       {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
     Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
     Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
     Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
     Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
     Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
     Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
     Hashtbl.add table "house"
       {total = 2; amounts = [("where", 1); ("in", 1)]} ;
     table)
    "beer"
```

Expected word in 1 pt

Computing

```
  next_in_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
```

```
Computing
  next_in_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "of"
```

Expected word his 1 pt

```
Computing
  next_in_htable
    (let table = Hashtbl.create 16 in
     Hashtbl.add table "a"
       {total = 7;
        amounts =
         [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
     Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "is" {total = 3; amounts = [("no", 1); ("a", 2)]} ;
     Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
     Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
     Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
     Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
     Hashtbl.add table "kitchen"
       {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
     Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
     Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
     Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
     Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
     Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
     Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
     Hashtbl.add table "house"
       {total = 2; amounts = [("where", 1); ("in", 1)]} ;
     table)
    "START"
```

Expected word there 1 pt

```
Computing
  next_in_htable
    (let table = Hashtbl.create 11 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "daughter"
       {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good"
       {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
     Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "her"
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "a"
```

Expected word good 1 pt

```
Computing
  next_in_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "of"
```

Expected word his 1 pt

```
Computing
  next_in_htable
    (let table = Hashtbl.create 16 in
     Hashtbl.add table "a"
       {total = 7;
        amounts =
         [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
    Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
    Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
    Hashtbl.add table "kitchen"
      {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
    Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
    Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
    Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
    Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
    Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
    Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
    Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
    Hashtbl.add table "house"
      {total = 2; amounts = [("where", 1); ("in", 1)]} ;
    table)
   "has"
```

Expected word a	1 pt

Computing
```
  next_in_htable
    (let table = Hashtbl.create 11 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "daughter"
       {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good"
       {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
     Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "her"
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "a"
```

Expected word good	1 pt

Computing
```
  next_in_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
    "his"
```

Expected word son	1 pt

Found `next_in_htable` with compatible type.

Now I will check the probabilities.

Testing on `"a b a c a"`

Expected distribution	1 pt

Expected distribution for `"c"`
```
"c" -> "a" 100%
```
Expected distribution for `"a"`
```
"a" -> "STOP" 33%
"a" -> "b" 33%
"a" -> "c" 34%
```
Expected distribution for `"START"`
```
"START" -> "a" 100%
```
Expected distribution for `"b"`
```
"b" -> "a" 100%
```
Testing on `"a b a c a c a"`

Expected distribution	1 pt

Expected distribution for `"c"`
```
"c" -> "a" 100%
```
Expected distribution for `"a"`
```
"a" -> "STOP" 25%
"a" -> "b" 24%
"a" -> "c" 50%
```
Expected distribution for `"START"`
```
"START" -> "a" 100%
```
Expected distribution for `"b"`
```
"b" -> "a" 100%
```

v Exercise 8: `walk_htable`	Completed, 10 pts

Found `walk_htable` with compatible type.

Computing
```
  walk_htable
    (let table = Hashtbl.create 16 in
     Hashtbl.add table "a"
       {total = 7;
  Hashtbl.add table "is" {total = 3; amounts = [("no", 1); ("a", 2)]} ;
  Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
  Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
  Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
  Hashtbl.add table "kitchen"
    {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
  Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
  Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
  Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
  Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
  Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
  Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table "house"
    {total = 2; amounts = [("where", 1); ("in", 1)]} ;
  table)
```
Checking there is a kitchen

Correct sequence. 1 pt

Computing
```
 walk_htable
   (let table = Hashtbl.create 11 in
    Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
    Hashtbl.add table "daughter"
      {total = 2; amounts = [("is", 1); ("and", 1)]} ;
    Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
    Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
    Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table "good"
      {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
    Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
    Hashtbl.add table "her"
      {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
    Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
    table)
```
Checking a good woman is proud of her daughter is proud of her daughter and a good woman is proud of her daughter is proud of her daughter and a good woman is proud of her mom

Correct sequence. 1 pt

Computing
```
 walk_htable
   (let table = Hashtbl.create 10 in
    Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
    Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
    Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
    Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
    Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
    Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
    Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
    Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
    Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
    table)
```
Checking a good dad is proud of his son is proud of his son is proud of his son and a good son and a good son is proud of his dad is proud of his dad is proud of his son and a good dad is proud of his son is proud of his dad

Correct sequence. 1 pt

Computing
```
 walk_htable
   (let table = Hashtbl.create 11 in
    Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
    Hashtbl.add table "daughter"
      {total = 2; amounts = [("is", 1); ("and", 1)]} ;
    Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
    Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
    Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table "good"
      {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
    Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
    Hashtbl.add table "her"
      {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
    Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
    table)
```
Checking a good daughter is proud of her daughter is proud of her mom

Correct sequence. 1 pt

Computing
```
 walk_htable
   (let table = Hashtbl.create 16 in
    Hashtbl.add table "a"
      {total = 7;
       amounts =
        [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
         ("beer", 1)]} ;
    Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table "is" {total = 3; amounts = [("no", 1); ("a", 2)]} ;
    Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
```

```
       {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
     Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
     Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
     Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
     Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
     Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
     Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
     Hashtbl.add table "house"
       {total = 2; amounts = [("where", 1); ("in", 1)]} ;
     table)
```

Checking there is no beer in the kitchen in a kitchen

Correct sequence. 1 pt

Computing

```
  walk_htable
    (let table = Hashtbl.create 11 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "daughter"
       {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good"
       {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
     Hashtbl.add table "woman" {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "her"
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
```

Checking a good woman is proud of her daughter and a good daughter and a good daughter and a good daughter and a good woman is proud of her daughter and a good woman is proud of her mom

Correct sequence. 1 pt

Computing

```
  walk_htable
    (let table = Hashtbl.create 16 in
     Hashtbl.add table "a"
       {total = 7;
        amounts =
         [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
     Hashtbl.add table "has" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "is" {total = 3; amounts = [("no", 1); ("a", 2)]} ;
     Hashtbl.add table "there" {total = 3; amounts = [("is", 3)]} ;
     Hashtbl.add table "man" {total = 1; amounts = [("who", 1)]} ;
     Hashtbl.add table "where" {total = 2; amounts = [("there", 2)]} ;
     Hashtbl.add table "beer" {total = 2; amounts = [("in", 2)]} ;
     Hashtbl.add table "kitchen"
       {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
     Hashtbl.add table "no" {total = 1; amounts = [("beer", 1)]} ;
     Hashtbl.add table "land" {total = 1; amounts = [("where", 1)]} ;
     Hashtbl.add table "who" {total = 1; amounts = [("has", 1)]} ;
     Hashtbl.add table "fridge" {total = 1; amounts = [("in", 1)]} ;
     Hashtbl.add table "the" {total = 1; amounts = [("kitchen", 1)]} ;
     Hashtbl.add table "in" {total = 5; amounts = [("the", 1); ("a", 4)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("there", 1)]} ;
     Hashtbl.add table "house"
       {total = 2; amounts = [("where", 1); ("in", 1)]} ;
     table)
```

Checking there is a house where there is no beer in a fridge in a beer in the kitchen

Correct sequence. 1 pt

Computing

```
  walk_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
```

Checking a good son and a good dad

Correct sequence. 1 pt

Computing

```
  walk_htable
    (let table = Hashtbl.create 11 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "daughter"
       {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "mom" {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("her", 2)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "her"
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
```
Checking a good woman is proud of her mom

Correct sequence. 1 pt

Computing
```
  walk_htable
    (let table = Hashtbl.create 10 in
     Hashtbl.add table "a" {total = 2; amounts = [("good", 2)]} ;
     Hashtbl.add table "son" {total = 2; amounts = [("is", 1); ("and", 1)]} ;
     Hashtbl.add table "is" {total = 2; amounts = [("proud", 2)]} ;
     Hashtbl.add table "of" {total = 2; amounts = [("his", 2)]} ;
     Hashtbl.add table "and" {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table "good" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "dad" {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
     Hashtbl.add table "proud" {total = 2; amounts = [("of", 2)]} ;
     Hashtbl.add table "his" {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
     Hashtbl.add table "START" {total = 1; amounts = [("a", 1)]} ;
     table)
```
Checking a good son is proud of his dad

Correct sequence. 1 pt

v Exercise 9: `sentences` Completed, 50 pts

Found `sentences` with compatible type.

Computing
```
  sentences
    "a good woman is proud of her daughter and a good daughter is proud of her mom"
```
Text splitted as expected. 5 pts

Computing
```
  sentences
    "there is a beer in a fridge in a kitchen in a house in a land where there is a man who
```
Text splitted as expected. 5 pts

Computing
```
  sentences
    "\nAn honest farmer had once an ass that had been a faithful servant to him\na great man
```
Text splitted as expected. 5 pts

Computing `sentences "a good dad is proud of his son and a good son is proud of his dad"`

Text splitted as expected. 5 pts

Computing
```
  sentences
    "\nWine Chaudeau: Into a lined saucepan put \194\189 bottle Rhine wine,\n4 tablespoonful
```
Text splitted as expected. 5 pts

Computing `sentences "A good dad is proud of his son. A good son is proud of his dad."`

Text splitted as expected. 5 pts

Computing
```
  sentences
    "In a land of myths, and a time of magic, the destiny of a great kingdom rests on the sh
```
Text splitted as expected. 5 pts

Computing
```
  sentences
    "A good woman is proud of her daughter. A good daughter is proud of her mom."
```
Text splitted as expected. 5 pts

Computing
```
  sentences
    "\n\226\128\156Caml\226\128\157 was originally an acronym for Categorical Abstract Machi
```
Text splitted as expected. 5 pts

Computing
```
  sentences
    "\nA certain cat had made the acquaintance of a mouse, and had said so much\nto her abou
```
Text splitted as expected. 5 pts

v Exercise 10: `start` Completed, 5 pts

Found `start` with compatible type.

Computing `start 0`

Correct value `[]` 1 pt

Computing `start 1`

Correct value `["START"]` 1 pt

Computing `start 2`

Correct value `["START"; "START"]` 1 pt

Computing `start 3`

Correct value `["START"; "START"; "START"]` 1 pt

Computing `start 4`

Correct value `["START"; "START"; "START"; "START"]` 1 pt

v Exercise 11: `shift` Completed, 4 pts

Found `shift` with compatible type.

Computing `shift ["A"; "B"; "C"] "D"`

```
Computing shift ["before"] "after"
Correct value ["after"]                                                          1 pt
Computing shift ["an"; "never"; "gonna"] "give"
Correct value ["never"; "gonna"; "give"]                                         1 pt
v Exercise 12: build_ptable                                          Completed, 50 pts
Found build_ptable with compatible type.
Computing
  build_ptable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
    2
Expected table                                                                   5 pts
  (let table = Hashtbl.create 14 in
   Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["of"; "his"]
     {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
   Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["his"; "son"] {total = 1; amounts = [("and", 1)]} ;
   Hashtbl.add table ["a"; "good"]
     {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
   Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["son"; "and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("his", 2)]} ;
   Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
   table)
Computing build_ptable ["A"; "good"; "son"; "is"; "proud"; "of"; "his"; "dad"; "."] 2
Expected table                                                                   5 pts
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("dad", 1)]} ;
   Hashtbl.add table ["dad"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
   Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("son", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
   table)
Computing build_ptable ["A"; "good"; "son"; "is"; "proud"; "of"; "his"; "dad"; "."] 1
Expected table                                                                   5 pts
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["dad"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["good"] {total = 1; amounts = [("son", 1)]} ;
   Hashtbl.add table ["is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["his"] {total = 1; amounts = [("dad", 1)]} ;
   Hashtbl.add table ["son"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["of"] {total = 1; amounts = [("his", 1)]} ;
   table)
Computing
  build_ptable
    ["A"; "good"; "woman"; "is"; "proud"; "of"; "her"; "daughter"; "."]
    1
Expected table                                                                   5 pts
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["her"] {total = 1; amounts = [("daughter", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["good"] {total = 1; amounts = [("woman", 1)]} ;
   Hashtbl.add table ["is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["daughter"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["of"] {total = 1; amounts = [("her", 1)]} ;
   table)
Computing build_ptable ["A"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "."] 1
Expected table                                                                   5 pts
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["dad"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["good"] {total = 1; amounts = [("dad", 1)]} ;
   Hashtbl.add table ["is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["son"] {total = 1; amounts = [(".", 1)]} ;
```

Computing
```
  build_ptable
    ["A"; "good"; "daughter"; "is"; "proud"; "of"; "her"; "mom"; "."]
    2
```
Expected table 5 pts
```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["good"; "daughter"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["daughter"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
   Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("daughter", 1)]} ;
   Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
   Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
   table)
```
Computing
```
  build_ptable
    ["In"; "a"; "land"; "of"; "myths"; ","; "and"; "a"; "time"; "of"; "magic";
     ","; "the"; "destiny"; "of"; "a"; "great"; "kingdom"; "rests"; "on";
     "the"; "shoulders"; "of"; "a"; "young"; "man"; "."]
    1
```
Expected table 5 pts
```
  (let table = Hashtbl.create 20 in
   Hashtbl.add table ["time"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["In"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["on"] {total = 1; amounts = [("the", 1)]} ;
   Hashtbl.add table ["rests"] {total = 1; amounts = [("on", 1)]} ;
   Hashtbl.add table ["young"] {total = 1; amounts = [("man", 1)]} ;
   Hashtbl.add table ["destiny"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table [","] {total = 2; amounts = [("and", 1); ("the", 1)]} ;
   Hashtbl.add table ["the"]
     {total = 2; amounts = [("destiny", 1); ("shoulders", 1)]} ;
   Hashtbl.add table ["a"]
     {total = 4;
      amounts = [("land", 1); ("time", 1); ("great", 1); ("young", 1)]} ;
   Hashtbl.add table ["man"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["magic"] {total = 1; amounts = [(",", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("In", 1)]} ;
   Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["great"] {total = 1; amounts = [("kingdom", 1)]} ;
   Hashtbl.add table ["shoulders"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["kingdom"] {total = 1; amounts = [("rests", 1)]} ;
   Hashtbl.add table ["land"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["myths"] {total = 1; amounts = [(",", 1)]} ;
   Hashtbl.add table ["of"]
     {total = 4; amounts = [("myths", 1); ("magic", 1); ("a", 2)]} ;
   table)
```
Computing
```
  build_ptable
    ["a"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "and"; "a"; "good";
     "son"; "is"; "proud"; "of"; "his"; "dad"]
    1
```
Expected table 5 pts
```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["dad"] {total = 2; amounts = [("is", 1); ("STOP", 1)]} ;
   Hashtbl.add table ["a"] {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["proud"] {total = 2; amounts = [("of", 2)]} ;
   Hashtbl.add table ["good"] {total = 2; amounts = [("dad", 1); ("son", 1)]} ;
   Hashtbl.add table ["is"] {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table ["son"] {total = 2; amounts = [("and", 1); ("is", 1)]} ;
   Hashtbl.add table ["his"] {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
   Hashtbl.add table ["of"] {total = 2; amounts = [("his", 2)]} ;
   table)
```
Computing `build_ptable ["A"; "good"; "dad"; "is"; "proud"; "of"; "his"; "son"; "."] 2`

Expected table 5 pts
```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
   Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
   Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
   Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
   table)
```
Computing
```
  build_ptable
    ["there"; "is"; "a"; "beer"; "in"; "a"; "fridge"; "in"; "a"; "kitchen";
     "in"; "a"; "house"; "in"; "a"; "land"; "where"; "there"; "is"; "a"; "man";
     "who"; "has"; "a"; "house"; "where"; "there"; "is"; "no"; "beer"; "in";
     "the"; "kitchen"]
    1
```
Expected table 5 pts

```
   amounts =
    [("beer", 1); ("fridge", 1); ("kitchen", 1); ("house", 2); ("land", 1);
     ("man", 1)]} ;
Hashtbl.add table ["man"] {total = 1; amounts = [("who", 1)]} ;
Hashtbl.add table ["START"] {total = 1; amounts = [("there", 1)]} ;
Hashtbl.add table ["has"] {total = 1; amounts = [("a", 1)]} ;
Hashtbl.add table ["fridge"] {total = 1; amounts = [("in", 1)]} ;
Hashtbl.add table ["house"]
  {total = 2; amounts = [("in", 1); ("where", 1)]} ;
Hashtbl.add table ["there"] {total = 3; amounts = [("is", 3)]} ;
Hashtbl.add table ["kitchen"]
  {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
Hashtbl.add table ["is"] {total = 3; amounts = [("a", 2); ("no", 1)]} ;
Hashtbl.add table ["beer"] {total = 2; amounts = [("in", 2)]} ;
Hashtbl.add table ["land"] {total = 1; amounts = [("where", 1)]} ;
Hashtbl.add table ["no"] {total = 1; amounts = [("beer", 1)]} ;
Hashtbl.add table ["where"] {total = 2; amounts = [("there", 2)]} ;
Hashtbl.add table ["in"] {total = 5; amounts = [("a", 4); ("the", 1)]} ;
Hashtbl.add table ["who"] {total = 1; amounts = [("has", 1)]} ;
table)
```

v Exercise 13: walk_ptable — Completed, 10 pts

Found walk_ptable with compatible type.

Computing

```
walk_ptable
  (let prefix_length = 1 in
   let table = Hashtbl.create 21 in
   Hashtbl.add table ["woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["mom"] {total = 2; amounts = [(".", 1); ("STOP", 1)]} ;
   Hashtbl.add table ["mom"] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"]
     {total = 3; amounts = [("of", 1); ("of", 2)]} ;
   Hashtbl.add table ["proud"] {total = 2; amounts = [("of", 2)]} ;
   Hashtbl.add table ["a"] {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table ["her"]
     {total = 3; amounts = [("mom", 1); ("mom", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["her"]
     {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["is"]
     {total = 3; amounts = [("proud", 1); ("proud", 2)]} ;
   Hashtbl.add table ["is"] {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table ["START"] {total = 2; amounts = [("A", 1); ("a", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["good"]
     {total = 3; amounts = [("daughter", 1); ("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["good"]
     {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["daughter"]
     {total = 3; amounts = [("is", 1); ("is", 1); ("and", 1)]} ;
   Hashtbl.add table ["daughter"]
     {total = 2; amounts = [("is", 1); ("and", 1)]} ;
   Hashtbl.add table ["of"] {total = 3; amounts = [("her", 1); ("her", 2)]} ;
   Hashtbl.add table ["of"] {total = 2; amounts = [("her", 2)]} ;
   { prefix_length ; table })
```

Checking A good woman is proud of her mom .

Correct sequence. — 1 pt

Computing

```
walk_ptable
  (let prefix_length = 1 in
   let table = Hashtbl.create 21 in
   Hashtbl.add table ["woman"]
     {total = 2; amounts = [("is", 1); ("is", 1)]} ;
   Hashtbl.add table ["woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["mom"] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"]
     {total = 3; amounts = [("of", 1); ("of", 2)]} ;
   Hashtbl.add table ["proud"] {total = 2; amounts = [("of", 2)]} ;
   Hashtbl.add table ["a"] {total = 2; amounts = [("good", 2)]} ;
   Hashtbl.add table ["her"]
     {total = 3; amounts = [("daughter", 1); ("mom", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["her"]
     {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["is"]
     {total = 3; amounts = [("proud", 1); ("proud", 2)]} ;
   Hashtbl.add table ["is"] {total = 2; amounts = [("proud", 2)]} ;
   Hashtbl.add table ["START"] {total = 2; amounts = [("A", 1); ("a", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["good"]
     {total = 3; amounts = [("woman", 1); ("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["good"]
     {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["daughter"]
     {total = 3; amounts = [(".", 1); ("is", 1); ("and", 1)]} ;
     { prefix_length ; table })
```
Checking a good woman is proud of her daughter .

Correct sequence. 1 pt

Computing
```
  walk_ptable
    (let prefix_length = 1 in
     let table = Hashtbl.create 36 in
     Hashtbl.add table ["there"] {total = 3; amounts = [("is", 3)]} ;
     Hashtbl.add table ["the"]
       {total = 3;
        amounts = [("shoulders", 1); ("destiny", 1); ("kitchen", 1)]} ;
     Hashtbl.add table ["the"] {total = 1; amounts = [("kitchen", 1)]} ;
     Hashtbl.add table ["man"] {total = 2; amounts = [(".", 1); ("who", 1)]} ;
     Hashtbl.add table ["magic"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["man"] {total = 1; amounts = [("who", 1)]} ;
     Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["who"] {total = 1; amounts = [("has", 1)]} ;
     Hashtbl.add table ["has"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["young"] {total = 1; amounts = [("man", 1)]} ;
     Hashtbl.add table ["destiny"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["a"]
       {total = 11;
        amounts =
         [("young", 1); ("time", 1); ("land", 1); ("great", 1); ("man", 1);
          ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
     Hashtbl.add table ["a"]
       {total = 7;
        amounts =
         [("man", 1); ("land", 1); ("kitchen", 1); ("house", 2); ("fridge", 1);
          ("beer", 1)]} ;
     Hashtbl.add table ["shoulders"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["land"]
       {total = 2; amounts = [("of", 1); ("where", 1)]} ;
     Hashtbl.add table ["land"] {total = 1; amounts = [("where", 1)]} ;
     Hashtbl.add table ["myths"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["in"] {total = 5; amounts = [("the", 1); ("a", 4)]} ;
     Hashtbl.add table ["fridge"] {total = 1; amounts = [("in", 1)]} ;
     Hashtbl.add table ["on"] {total = 1; amounts = [("the", 1)]} ;
     Hashtbl.add table ["rests"] {total = 1; amounts = [("on", 1)]} ;
     Hashtbl.add table ["is"] {total = 3; amounts = [("no", 1); ("a", 2)]} ;
     Hashtbl.add table ["beer"] {total = 2; amounts = [("in", 2)]} ;
     Hashtbl.add table ["START"]
       {total = 2; amounts = [("In", 1); ("there", 1)]} ;
     Hashtbl.add table ["START"] {total = 1; amounts = [("there", 1)]} ;
     Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["great"] {total = 1; amounts = [("kingdom", 1)]} ;
     Hashtbl.add table ["kitchen"]
       {total = 2; amounts = [("in", 1); ("STOP", 1)]} ;
     Hashtbl.add table ["where"] {total = 2; amounts = [("there", 2)]} ;
     Hashtbl.add table ["time"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["In"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["house"]
       {total = 2; amounts = [("where", 1); ("in", 1)]} ;
     Hashtbl.add table ["no"] {total = 1; amounts = [("beer", 1)]} ;
     Hashtbl.add table [","] {total = 2; amounts = [("the", 1); ("and", 1)]} ;
     Hashtbl.add table ["kingdom"] {total = 1; amounts = [("rests", 1)]} ;
     Hashtbl.add table ["of"]
       {total = 4; amounts = [("myths", 1); ("magic", 1); ("a", 2)]} ;
     { prefix_length ; table })
```
Checking there is a great kingdom rests on the destiny of magic , and a time of myths , the shoulders of a house where there is a house in a man who has a young man .

Correct sequence. 1 pt

Computing
```
  walk_ptable
    (let prefix_length = 1 in
     let table = Hashtbl.create 30 in
     Hashtbl.add table ["the"]
       {total = 2; amounts = [("shoulders", 1); ("destiny", 1)]} ;
     Hashtbl.add table ["man"] {total = 1; amounts = [(".", 1)]} ;
     Hashtbl.add table ["magic"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["and"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["his"] {total = 1; amounts = [("dad", 1)]} ;
     Hashtbl.add table ["proud"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["young"] {total = 1; amounts = [("man", 1)]} ;
     Hashtbl.add table ["destiny"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["dad"] {total = 1; amounts = [(".", 1)]} ;
     Hashtbl.add table ["a"]
       {total = 4;
        amounts = [("young", 1); ("time", 1); ("land", 1); ("great", 1)]} ;
     Hashtbl.add table ["shoulders"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["land"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["myths"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["on"] {total = 1; amounts = [("the", 1)]} ;
     Hashtbl.add table ["rests"] {total = 1; amounts = [("on", 1)]} ;
     Hashtbl.add table ["is"] {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["START"] {total = 2; amounts = [("A", 1); ("In", 1)]} ;
     Hashtbl.add table ["START"] {total = 1; amounts = [("In", 1)]} ;
     Hashtbl.add table ["."]
     Hashtbl.add table ["In"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table ["son"] {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table [","] {total = 2; amounts = [("the", 1); ("and", 1)]} ;
     Hashtbl.add table ["good"] {total = 1; amounts = [("son", 1)]} ;
     Hashtbl.add table ["kingdom"] {total = 1; amounts = [("rests", 1)]} ;
     Hashtbl.add table ["of"]
       {total = 5;
        amounts = [("his", 1); ("myths", 1); ("magic", 1); ("a", 2)]} ;
     Hashtbl.add table ["of"]
       {total = 4; amounts = [("myths", 1); ("magic", 1); ("a", 2)]} ;
     { prefix_length ; table })
```
Checking In a young man .

Correct sequence. 1 pt

Computing
```
  walk_ptable
    (let prefix_length = 2 in
     let table = Hashtbl.create 24 in
     Hashtbl.add table ["daughter"; "and"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
     Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
     Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["proud"; "of"]
       {total = 3; amounts = [("her", 2); ("his", 1)]} ;
     Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
     Hashtbl.add table ["good"; "daughter"]
       {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
     Hashtbl.add table ["START"; "START"]
       {total = 2; amounts = [("a", 1); ("A", 1)]} ;
     Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
     Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table ["a"; "good"]
       {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
     Hashtbl.add table ["daughter"; "is"]
       {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["of"; "her"]
       {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
     Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table ["her"; "daughter"]
       {total = 1; amounts = [("and", 1)]} ;
     Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["is"; "proud"]
       {total = 3; amounts = [("of", 2); ("of", 1)]} ;
     Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
     { prefix_length ; table })
```
Checking a good woman is proud of her daughter and a good daughter is proud of her mom

Correct sequence. 1 pt

Computing
```
  walk_ptable
    (let prefix_length = 2 in
     let table = Hashtbl.create 20 in
     Hashtbl.add table ["START"; "A"]
       {total = 2; amounts = [("good", 1); ("good", 1)]} ;
     Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table ["A"; "good"]
       {total = 2; amounts = [("daughter", 1); ("dad", 1)]} ;
     Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
     Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
     Hashtbl.add table ["proud"; "of"]
       {total = 2; amounts = [("her", 1); ("his", 1)]} ;
     Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
     Hashtbl.add table ["good"; "daughter"]
       {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
     Hashtbl.add table ["START"; "START"]
       {total = 2; amounts = [("A", 1); ("A", 1)]} ;
     Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
     Hashtbl.add table ["daughter"; "is"]
       {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
     Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
     Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["is"; "proud"]
       {total = 2; amounts = [("of", 1); ("of", 1)]} ;
     Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
     { prefix_length ; table })
```
Checking A good daughter is proud of her mom .

Correct sequence. 1 pt

Computing

```
      {total = 4; amounts = [("son", 1); ("dad", 1); ("son", 1); ("dad", 1)]} ;
    Hashtbl.add table ["of"; "his"]
      {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
    Hashtbl.add table ["son"; "and"]
      {total = 2; amounts = [("a", 1); ("a", 1)]} ;
    Hashtbl.add table ["son"; "and"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["proud"; "of"]
      {total = 4; amounts = [("his", 2); ("his", 2)]} ;
    Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("his", 2)]} ;
    Hashtbl.add table ["his"; "son"]
      {total = 2; amounts = [("and", 1); ("and", 1)]} ;
    Hashtbl.add table ["his"; "son"] {total = 1; amounts = [("and", 1)]} ;
    Hashtbl.add table ["START"; "START"]
      {total = 2; amounts = [("a", 1); ("a", 1)]} ;
    Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["and"; "a"]
      {total = 2; amounts = [("good", 1); ("good", 1)]} ;
    Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
    Hashtbl.add table ["good"; "son"]
      {total = 2; amounts = [("is", 1); ("is", 1)]} ;
    Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table ["a"; "good"]
      {total = 4; amounts = [("son", 1); ("dad", 1); ("son", 1); ("dad", 1)]} ;
    Hashtbl.add table ["a"; "good"]
      {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
    Hashtbl.add table ["son"; "is"]
      {total = 2; amounts = [("proud", 1); ("proud", 1)]} ;
    Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
    Hashtbl.add table ["good"; "dad"]
      {total = 2; amounts = [("is", 1); ("is", 1)]} ;
    Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table ["START"; "a"]
      {total = 2; amounts = [("good", 1); ("good", 1)]} ;
    Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
    Hashtbl.add table ["his"; "dad"]
      {total = 2; amounts = [("STOP", 1); ("STOP", 1)]} ;
    Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table ["dad"; "is"]
      {total = 2; amounts = [("proud", 1); ("proud", 1)]} ;
    Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
    Hashtbl.add table ["is"; "proud"]
      {total = 4; amounts = [("of", 2); ("of", 2)]} ;
    Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
    { prefix_length ; table })
```

Checking a good son is proud of his dad

Correct sequence. 1 pt

Computing

```
  walk_ptable
    (let prefix_length = 2 in
     let table = Hashtbl.create 37 in
     Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
     Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
     Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("daughter", 1)]} ;
     Hashtbl.add table ["the"; "shoulders"]
       {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
     Hashtbl.add table ["kingdom"; "rests"]
       {total = 1; amounts = [("on", 1)]} ;
     Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
     Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
     Hashtbl.add table ["a"; "great"] {total = 1; amounts = [("kingdom", 1)]} ;
     Hashtbl.add table ["good"; "daughter"]
       {total = 1; amounts = [("is", 1)]} ;
     Hashtbl.add table ["START"; "START"]
       {total = 2; amounts = [("A", 1); ("In", 1)]} ;
     Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("In", 1)]} ;
     Hashtbl.add table ["on"; "the"]
       {total = 1; amounts = [("shoulders", 1)]} ;
     Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
     Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
     Hashtbl.add table ["daughter"; "is"]
       {total = 1; amounts = [("proud", 1)]} ;
     Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
     Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
     Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
     Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
     Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
     Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
     Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
     Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
     Hashtbl.add table ["of"; "a"]
       {total = 2; amounts = [("young", 1); ("great", 1)]} ;
     Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
     Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
     {total = 1; amounts = [("rests", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
   { prefix_length ; table })
```
Checking A good daughter is proud of her mom .

Correct sequence. 1 pt

Computing
```
walk_ptable
  (let prefix_length = 1 in
   let table = Hashtbl.create 20 in
   Hashtbl.add table ["woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["his"] {total = 1; amounts = [("dad", 1)]} ;
   Hashtbl.add table ["proud"]
     {total = 2; amounts = [("of", 1); ("of", 1)]} ;
   Hashtbl.add table ["proud"] {total = 1; amounts = [("of", 1)]} ;
   Hashtbl.add table ["dad"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["her"] {total = 1; amounts = [("daughter", 1)]} ;
   Hashtbl.add table ["is"]
     {total = 2; amounts = [("proud", 1); ("proud", 1)]} ;
   Hashtbl.add table ["is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["START"] {total = 2; amounts = [("A", 1); ("A", 1)]} ;
   Hashtbl.add table ["START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["."]
     {total = 2; amounts = [("STOP", 1); ("STOP", 1)]} ;
   Hashtbl.add table ["."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["A"]
     {total = 2; amounts = [("good", 1); ("good", 1)]} ;
   Hashtbl.add table ["A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["son"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["good"]
     {total = 2; amounts = [("son", 1); ("woman", 1)]} ;
   Hashtbl.add table ["good"] {total = 1; amounts = [("woman", 1)]} ;
   Hashtbl.add table ["daughter"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["of"] {total = 2; amounts = [("his", 1); ("her", 1)]} ;
   Hashtbl.add table ["of"] {total = 1; amounts = [("her", 1)]} ;
   { prefix_length ; table })
```
Checking A good woman is proud of her daughter .

Correct sequence. 1 pt

Computing
```
walk_ptable
  (let prefix_length = 2 in
   let table = Hashtbl.create 39 in
   Hashtbl.add table ["daughter"; "and"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
   Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
   Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
   Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("her", 2)]} ;
   Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
   Hashtbl.add table ["good"; "daughter"]
     {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["START"; "START"]
     {total = 2; amounts = [("there", 1); ("a", 1)]} ;
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
   Hashtbl.add table ["in"; "a"]
     {total = 4;
      amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
   Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
   Hashtbl.add table ["there"; "is"]
     {total = 3; amounts = [("no", 1); ("a", 2)]} ;
   Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
   Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
   Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
   Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["beer"; "in"]
     {total = 2; amounts = [("the", 1); ("a", 1)]} ;
   Hashtbl.add table ["a"; "good"]
     {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
   Hashtbl.add table ["daughter"; "is"]
     {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["the"; "kitchen"]
     {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["of"; "her"]
     {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
   Hashtbl.add table ["house"; "where"]
     {total = 1; amounts = [("there", 1)]} ;
   Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
   Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["land"; "where"]
     {total = 1; amounts = [("there", 1)]} ;
   Hashtbl.add table ["her"; "daughter"]
     {total = 1; amounts = [("and", 1)]} ;
```

```
    Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
    Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
    { prefix_length ; table })
```

Checking a good daughter is proud of her mom

Correct sequence. 1 pt

v Exercise 14: `merge_ptables` Completed, 50 pts

Found `merge_ptables` with compatible type.

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 25 in
      Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
      Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
      Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
      Hashtbl.add table ["START"; "START"]
        {total = 1; amounts = [("there", 1)]} ;
      Hashtbl.add table ["in"; "a"]
        {total = 4;
         amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
      Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["there"; "is"]
        {total = 3; amounts = [("no", 1); ("a", 2)]} ;
      Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
      Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
      Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
      Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["beer"; "in"]
        {total = 2; amounts = [("the", 1); ("a", 1)]} ;
      Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["the"; "kitchen"]
        {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["house"; "where"]
        {total = 1; amounts = [("there", 1)]} ;
      Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["land"; "where"]
        {total = 1; amounts = [("there", 1)]} ;
      Hashtbl.add table ["is"; "a"]
        {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
      Hashtbl.add table ["a"; "house"]
        {total = 2; amounts = [("where", 1); ("in", 1)]} ;
      Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
      { prefix_length ; table })]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 25 in
  Hashtbl.add table ["is"; "a"]
    {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
  Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
  Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["beer"; "in"]
    {total = 2; amounts = [("the", 1); ("a", 1)]} ;
  Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table ["a"; "house"]
    {total = 2; amounts = [("where", 1); ("in", 1)]} ;
  Hashtbl.add table ["the"; "kitchen"] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["in"; "a"]
    {total = 4;
     amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
  Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["there"; "is"]
    {total = 3; amounts = [("no", 1); ("a", 2)]} ;
  Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
  Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
  Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["house"; "where"] {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
  Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
  Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
  Hashtbl.add table ["land"; "where"] {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
  table)
```

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
```

```
 Hashtbl.add table ["daughter"; "."]
   {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["of"; "her"]
   {total = 1; amounts = [("daughter", 1)]} ;
 Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
 Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
 { prefix_length ; table });
(let prefix_length = 2 in
 let table = Hashtbl.create 14 in
 Hashtbl.add table ["daughter"; "and"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("her", 2)]} ;
 Hashtbl.add table ["good"; "daughter"]
   {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
 Hashtbl.add table ["a"; "good"]
   {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
 Hashtbl.add table ["daughter"; "is"]
   {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["of"; "her"]
   {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
 Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
 Hashtbl.add table ["her"; "daughter"]
   {total = 1; amounts = [("and", 1)]} ;
 Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
 { prefix_length ; table });
(let prefix_length = 2 in
 let table = Hashtbl.create 25 in
 Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
 Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
 Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
 Hashtbl.add table ["START"; "START"]
   {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["in"; "a"]
   {total = 4;
    amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
 Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["there"; "is"]
   {total = 3; amounts = [("no", 1); ("a", 2)]} ;
 Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
 Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
 Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
 Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["beer"; "in"]
   {total = 2; amounts = [("the", 1); ("a", 1)]} ;
 Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["the"; "kitchen"]
   {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["house"; "where"]
   {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["land"; "where"]
   {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["is"; "a"]
   {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
 Hashtbl.add table ["a"; "house"]
   {total = 2; amounts = [("where", 1); ("in", 1)]} ;
 Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
 { prefix_length ; table })]
```

Expected table 5 pts

```
(let table = Hashtbl.create 41 in
 Hashtbl.add table ["daughter"; "and"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
 Hashtbl.add table ["a"; "good"]
   {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
 Hashtbl.add table ["beer"; "in"]
   {total = 2; amounts = [("the", 1); ("a", 1)]} ;
 Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
 Hashtbl.add table ["daughter"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["daughter"; "."] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["the"; "kitchen"] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
 Hashtbl.add table ["of"; "her"]
   {total = 3; amounts = [("daughter", 1); ("mom", 1); ("daughter", 1)]} ;
 Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["house"; "where"] {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
 Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
```

```
  Hashtbl.add table ["proud"; "of"]
    {total = 3; amounts = [("her", 1); ("her", 2)]} ;
  Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["her"; "daughter"]
    {total = 2; amounts = [(".", 1); ("and", 1)]} ;
  Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table ["good"; "daughter"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["is"; "a"]
    {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
  Hashtbl.add table ["START"; "START"]
    {total = 3; amounts = [("A", 1); ("a", 1); ("there", 1)]} ;
  Hashtbl.add table ["a"; "house"]
    {total = 2; amounts = [("where", 1); ("in", 1)]} ;
  Hashtbl.add table ["in"; "a"]
    {total = 4;
     amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
  Hashtbl.add table ["good"; "woman"]
    {total = 2; amounts = [("is", 1); ("is", 1)]} ;
  Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["there"; "is"]
    {total = 3; amounts = [("no", 1); ("a", 2)]} ;
  Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
  Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
  Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
  Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
  Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
  Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["is"; "proud"]
    {total = 3; amounts = [("of", 1); ("of", 2)]} ;
  Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
  table)
```

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
      Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
      Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 14 in
      Hashtbl.add table ["of"; "his"]
        {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
      Hashtbl.add table ["son"; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("his", 2)]} ;
      Hashtbl.add table ["his"; "son"] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["a"; "good"]
        {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
      Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 14 in
      Hashtbl.add table ["of"; "his"]
        {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
      Hashtbl.add table ["son"; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 2; amounts = [("his", 2)]} ;
      Hashtbl.add table ["his"; "son"] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["a"; "good"]
        {total = 2; amounts = [("son", 1); ("dad", 1)]} ;
      Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
      { prefix_length ; table })]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 17 in
  Hashtbl.add table ["a"; "good"]
    {total = 4; amounts = [("son", 1); ("dad", 1); ("son", 1); ("dad", 1)]} ;
  Hashtbl.add table ["his"; "son"]
    {total = 3; amounts = [(".", 1); ("and", 1); ("and", 1)]} ;
    {total = 2; amounts = [("proud", 1); ("proud", 1)]} ;
  Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
  Hashtbl.add table ["his"; "dad"]
    {total = 2; amounts = [("STOP", 1); ("STOP", 1)]} ;
  Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["of"; "his"]
    {total = 5;
     amounts = [("son", 1); ("son", 1); ("dad", 1); ("son", 1); ("dad", 1)]} ;
  Hashtbl.add table ["good"; "dad"]
    {total = 3; amounts = [("is", 1); ("is", 1); ("is", 1)]} ;
  Hashtbl.add table ["and"; "a"]
    {total = 2; amounts = [("good", 1); ("good", 1)]} ;
  Hashtbl.add table ["START"; "a"]
    {total = 2; amounts = [("good", 1); ("good", 1)]} ;
  Hashtbl.add table ["son"; "and"]
    {total = 2; amounts = [("a", 1); ("a", 1)]} ;
  Hashtbl.add table ["good"; "son"]
    {total = 2; amounts = [("is", 1); ("is", 1)]} ;
  Hashtbl.add table ["proud"; "of"]
    {total = 5; amounts = [("his", 1); ("his", 2); ("his", 2)]} ;
  Hashtbl.add table ["dad"; "is"]
    {total = 3; amounts = [("proud", 1); ("proud", 1); ("proud", 1)]} ;
  Hashtbl.add table ["is"; "proud"]
    {total = 5; amounts = [("of", 1); ("of", 2); ("of", 2)]} ;
  table)
Computing
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("son", 1)]} ;
      Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("dad", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["dad"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"]
        {total = 1; amounts = [("daughter", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
      Hashtbl.add table ["good"; "daughter"]
        {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["daughter"; "is"]
        {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
      Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 27 in
      Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
      Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["the"; "shoulders"]
        {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["kingdom"; "rests"]
        {total = 1; amounts = [("on", 1)]} ;
      Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
      Hashtbl.add table ["a"; "great"]
        {total = 1; amounts = [("kingdom", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("In", 1)]} ;
      Hashtbl.add table ["on"; "the"]
        {total = 1; amounts = [("shoulders", 1)]} ;
      Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
      Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
      Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
      Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
      Hashtbl.add table ["of"; "a"]
        {total = 2; amounts = [("young", 1); ("great", 1)]} ;
      Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
```

```
 (let table = Hashtbl.create 41 in
  Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
  Hashtbl.add table ["dad"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["START"; "A"]
    {total = 2; amounts = [("good", 1); ("good", 1)]} ;
  Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
  Hashtbl.add table ["daughter"; "is"] {total = 1; amounts = [("proud", 1)]} ;
  Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
  Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["A"; "good"]
    {total = 2; amounts = [("son", 1); ("daughter", 1)]} ;
  Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
  Hashtbl.add table ["the"; "shoulders"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("dad", 1)]} ;
  Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
  Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
  Hashtbl.add table ["kingdom"; "rests"] {total = 1; amounts = [("on", 1)]} ;
  Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["proud"; "of"]
    {total = 2; amounts = [("his", 1); ("her", 1)]} ;
  Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
  Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
  Hashtbl.add table ["a"; "great"] {total = 1; amounts = [("kingdom", 1)]} ;
  Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["good"; "daughter"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
  Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
  Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
  Hashtbl.add table ["START"; "START"]
    {total = 3; amounts = [("A", 1); ("A", 1); ("In", 1)]} ;
  Hashtbl.add table ["on"; "the"] {total = 1; amounts = [("shoulders", 1)]} ;
  Hashtbl.add table ["of"; "a"]
    {total = 2; amounts = [("young", 1); ("great", 1)]} ;
  Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
  Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [(".", 1)]} ;
  Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
  Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
  Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
  Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["great"; "kingdom"]
    {total = 1; amounts = [("rests", 1)]} ;
  Hashtbl.add table ["is"; "proud"]
    {total = 2; amounts = [("of", 1); ("of", 1)]} ;
  table)
```

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 27 in
      Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
      Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["the"; "shoulders"]
        {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["kingdom"; "rests"]
        {total = 1; amounts = [("on", 1)]} ;
      Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
      Hashtbl.add table ["a"; "great"]
        {total = 1; amounts = [("kingdom", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("In", 1)]} ;
      Hashtbl.add table ["on"; "the"]
        {total = 1; amounts = [("shoulders", 1)]} ;
      Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
      Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
      Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
      Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
      Hashtbl.add table ["of"; "a"]
        {total = 2; amounts = [("young", 1); ("great", 1)]} ;
      Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["great"; "kingdom"]
        {total = 1; amounts = [("rests", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 14 in
        {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["a"; "good"]
        {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
      Hashtbl.add table ["daughter"; "is"]
        {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["of"; "her"]
        {total = 2; amounts = [("mom", 1); ("daughter", 1)]} ;
      Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["her"; "daughter"]
        {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 2; amounts = [("of", 2)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"]
        {total = 1; amounts = [("daughter", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
      Hashtbl.add table ["good"; "daughter"]
        {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["daughter"; "is"]
        {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
      Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table })]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 42 in
  Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
  Hashtbl.add table ["daughter"; "and"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "good"]
    {total = 2; amounts = [("woman", 1); ("daughter", 1)]} ;
  Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
  Hashtbl.add table ["daughter"; "is"]
    {total = 2; amounts = [("proud", 1); ("proud", 1)]} ;
  Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
  Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("daughter", 1)]} ;
  Hashtbl.add table ["of"; "her"]
    {total = 3; amounts = [("mom", 1); ("daughter", 1); ("mom", 1)]} ;
  Hashtbl.add table ["the"; "shoulders"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
  Hashtbl.add table ["START"; "a"] {total = 1; amounts = [("good", 1)]} ;
  Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
  Hashtbl.add table ["kingdom"; "rests"] {total = 1; amounts = [("on", 1)]} ;
  Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
  Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["proud"; "of"]
    {total = 3; amounts = [("her", 2); ("her", 1)]} ;
  Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [("and", 1)]} ;
  Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
  Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
  Hashtbl.add table ["a"; "great"] {total = 1; amounts = [("kingdom", 1)]} ;
  Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["good"; "daughter"]
    {total = 2; amounts = [("is", 1); ("is", 1)]} ;
  Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
  Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
  Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
  Hashtbl.add table ["START"; "START"]
    {total = 3; amounts = [("In", 1); ("a", 1); ("A", 1)]} ;
  Hashtbl.add table ["on"; "the"] {total = 1; amounts = [("shoulders", 1)]} ;
  Hashtbl.add table ["of"; "a"]
    {total = 2; amounts = [("young", 1); ("great", 1)]} ;
  Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
  Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["and"; "a"]
    {total = 2; amounts = [("time", 1); ("good", 1)]} ;
  Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
  Hashtbl.add table ["her"; "mom"]
    {total = 2; amounts = [("STOP", 1); (".", 1)]} ;
  Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
  Hashtbl.add table ["great"; "kingdom"]
    {total = 1; amounts = [("rests", 1)]} ;
  Hashtbl.add table ["is"; "proud"]
    {total = 3; amounts = [("of", 2); ("of", 1)]} ;
  table)
```

Computing

```
    Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
    Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
    Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
    Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
    Hashtbl.add table ["START"; "START"]
      {total = 1; amounts = [("there", 1)]} ;
    Hashtbl.add table ["in"; "a"]
      {total = 4;
       amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
    Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
    Hashtbl.add table ["there"; "is"]
      {total = 3; amounts = [("no", 1); ("a", 2)]} ;
    Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
    Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
    Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
    Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table ["beer"; "in"]
      {total = 2; amounts = [("the", 1); ("a", 1)]} ;
    Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
    Hashtbl.add table ["the"; "kitchen"]
      {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table ["house"; "where"]
      {total = 1; amounts = [("there", 1)]} ;
    Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
    Hashtbl.add table ["land"; "where"]
      {total = 1; amounts = [("there", 1)]} ;
    Hashtbl.add table ["is"; "a"]
      {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
    Hashtbl.add table ["a"; "house"]
      {total = 2; amounts = [("where", 1); ("in", 1)]} ;
    Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
    { prefix_length ; table });
   (let prefix_length = 2 in
    let table = Hashtbl.create 10 in
    Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
    Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
    Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
    Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
    Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
    Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
    Hashtbl.add table ["daughter"; "."]
      {total = 1; amounts = [("STOP", 1)]} ;
    Hashtbl.add table ["of"; "her"]
      {total = 1; amounts = [("daughter", 1)]} ;
    Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
    Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
    { prefix_length ; table })]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 34 in
  Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
  Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["beer"; "in"]
    {total = 2; amounts = [("the", 1); ("a", 1)]} ;
  Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
  Hashtbl.add table ["the"; "kitchen"] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["daughter"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
  Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("daughter", 1)]} ;
  Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["house"; "where"] {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
  Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
  Hashtbl.add table ["land"; "where"] {total = 1; amounts = [("there", 1)]} ;
  Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
  Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
  Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
  Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
  Hashtbl.add table ["is"; "a"]
    {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
  Hashtbl.add table ["START"; "START"]
    {total = 2; amounts = [("there", 1); ("A", 1)]} ;
  Hashtbl.add table ["a"; "house"]
    {total = 2; amounts = [("where", 1); ("in", 1)]} ;
  Hashtbl.add table ["in"; "a"]
    {total = 4;
     amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
  Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
  Hashtbl.add table ["there"; "is"]
    {total = 3; amounts = [("no", 1); ("a", 2)]} ;
  Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
  Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
  Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
```

FUN

Rechercher un cours

Computing
```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
      Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["daughter"; "."]
        {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "her"]
        {total = 1; amounts = [("daughter", 1)]} ;
      Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("dad", 1)]} ;
      Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
      Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table })]
```
Expected table 5 pts
```
  (let table = Hashtbl.create 15 in
   Hashtbl.add table ["son"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("son", 1)]} ;
   Hashtbl.add table ["good"; "dad"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["his"; "son"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["START"; "START"]
     {total = 2; amounts = [("A", 1); ("A", 1)]} ;
   Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["START"; "A"]
     {total = 2; amounts = [("good", 1); ("good", 1)]} ;
   Hashtbl.add table ["daughter"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"; "of"]
     {total = 2; amounts = [("her", 1); ("his", 1)]} ;
   Hashtbl.add table ["A"; "good"]
     {total = 2; amounts = [("woman", 1); ("dad", 1)]} ;
   Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("daughter", 1)]} ;
   Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["dad"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["is"; "proud"]
     {total = 2; amounts = [("of", 1); ("of", 1)]} ;
   table)
```
Computing
```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("son", 1)]} ;
      Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("dad", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("his", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["dad"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["his"; "dad"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 25 in
      Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
      Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
      Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
      Hashtbl.add table ["START"; "START"]
        {total = 1; amounts = [("there", 1)]} ;
      Hashtbl.add table ["in"; "a"]
        {total = 4;
         amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
      Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
      Hashtbl.add table ["there"; "is"]
        {total = 3; amounts = [("no", 1); ("a", 2)]} ;
      Hashtbl.add table ["where"; "there"] {total = 2; amounts = [("is", 2)]} ;
      Hashtbl.add table ["is"; "no"] {total = 1; amounts = [("beer", 1)]} ;
      Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
      Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
```

```
   {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["house"; "where"]
   {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["land"; "where"]
   {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["is"; "a"]
   {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
 Hashtbl.add table ["a"; "house"]
   {total = 2; amounts = [("where", 1); ("in", 1)]} ;
 Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("where", 1)]} ;
 { prefix_length ; table });
(let prefix_length = 2 in
 let table = Hashtbl.create 10 in
 Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
 Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
 Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
 Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
 Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["daughter"; "."]
   {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["of"; "her"]
   {total = 1; amounts = [("daughter", 1)]} ;
 Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
 Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
 { prefix_length ; table });
(let prefix_length = 2 in
 let table = Hashtbl.create 10 in
 Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
 Hashtbl.add table ["A"; "good"]
   {total = 1; amounts = [("daughter", 1)]} ;
 Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
 Hashtbl.add table ["good"; "daughter"]
   {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
 Hashtbl.add table ["daughter"; "is"]
   {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
 Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
 Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
 { prefix_length ; table })]
```

Expected table 5 pts

```
(let table = Hashtbl.create 43 in
 Hashtbl.add table ["a"; "kitchen"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["kitchen"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["house"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["in"; "the"] {total = 1; amounts = [("kitchen", 1)]} ;
 Hashtbl.add table ["dad"; "."] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["beer"; "in"]
   {total = 2; amounts = [("the", 1); ("a", 1)]} ;
 Hashtbl.add table ["a"; "fridge"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["START"; "A"]
   {total = 3; amounts = [("good", 1); ("good", 1); ("good", 1)]} ;
 Hashtbl.add table ["son"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["daughter"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["the"; "kitchen"] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["daughter"; "."] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["A"; "good"]
   {total = 3; amounts = [("son", 1); ("woman", 1); ("daughter", 1)]} ;
 Hashtbl.add table ["of"; "her"]
   {total = 2; amounts = [("daughter", 1); ("mom", 1)]} ;
 Hashtbl.add table ["fridge"; "in"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["house"; "where"] {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["of"; "his"] {total = 1; amounts = [("dad", 1)]} ;
 Hashtbl.add table ["man"; "who"] {total = 1; amounts = [("has", 1)]} ;
 Hashtbl.add table ["a"; "beer"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
 Hashtbl.add table ["land"; "where"] {total = 1; amounts = [("there", 1)]} ;
 Hashtbl.add table ["proud"; "of"]
   {total = 3; amounts = [("his", 1); ("her", 1); ("her", 1)]} ;
 Hashtbl.add table ["who"; "has"] {total = 1; amounts = [("a", 1)]} ;
 Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
 Hashtbl.add table ["a"; "man"] {total = 1; amounts = [("who", 1)]} ;
 Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
 Hashtbl.add table ["good"; "daughter"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["is"; "a"]
   {total = 2; amounts = [("man", 1); ("beer", 1)]} ;
 Hashtbl.add table ["START"; "START"]
   {total = 4; amounts = [("A", 1); ("there", 1); ("A", 1); ("A", 1)]} ;
 Hashtbl.add table ["a"; "house"]
   {total = 2; amounts = [("where", 1); ("in", 1)]} ;
 Hashtbl.add table ["in"; "a"]
   {total = 4;
    amounts = [("land", 1); ("kitchen", 1); ("house", 1); ("fridge", 1)]} ;
 Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["no"; "beer"] {total = 1; amounts = [("in", 1)]} ;
 Hashtbl.add table ["there"; "is"]
   {total = 3; amounts = [("no", 1); ("a", 2)]} ;
 Hashtbl.add table ["has"; "a"] {total = 1; amounts = [("house", 1)]} ;
 Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
 Hashtbl.add table ["good"; "son"] {total = 1; amounts = [("is", 1)]} ;
 Hashtbl.add table ["is"; "proud"]
   {total = 3; amounts = [("of", 1); ("of", 1); ("of", 1)]} ;
 Hashtbl.add table ["START"; "there"] {total = 1; amounts = [("is", 1)]} ;
 table)
```

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
      Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["daughter"; "."]
        {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "her"]
        {total = 1; amounts = [("daughter", 1)]} ;
      Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      { prefix_length ; table })]
```

Expected table 5 pts

```
  (let table = Hashtbl.create 10 in
   Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
   Hashtbl.add table ["woman"; "is"] {total = 1; amounts = [("proud", 1)]} ;
   Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
   Hashtbl.add table ["daughter"; "."] {total = 1; amounts = [("STOP", 1)]} ;
   Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
   Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("woman", 1)]} ;
   Hashtbl.add table ["her"; "daughter"] {total = 1; amounts = [(".", 1)]} ;
   Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("daughter", 1)]} ;
   Hashtbl.add table ["good"; "woman"] {total = 1; amounts = [("is", 1)]} ;
   Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
   table)
```

Computing

```
  merge_ptables
    [(let prefix_length = 2 in
      let table = Hashtbl.create 27 in
      Hashtbl.add table ["land"; "of"] {total = 1; amounts = [("myths", 1)]} ;
      Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["the"; "shoulders"]
        {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["kingdom"; "rests"]
        {total = 1; amounts = [("on", 1)]} ;
      Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
      Hashtbl.add table ["a"; "great"]
        {total = 1; amounts = [("kingdom", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("In", 1)]} ;
      Hashtbl.add table ["on"; "the"]
        {total = 1; amounts = [("shoulders", 1)]} ;
      Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
      Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
      Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
      Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
      Hashtbl.add table ["of"; "a"]
        {total = 2; amounts = [("young", 1); ("great", 1)]} ;
      Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["great"; "kingdom"]
        {total = 1; amounts = [("rests", 1)]} ;
      { prefix_length ; table });
     (let prefix_length = 2 in
      let table = Hashtbl.create 10 in
      Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
      Hashtbl.add table ["A"; "good"]
        {total = 1; amounts = [("daughter", 1)]} ;
      Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
      Hashtbl.add table ["good"; "daughter"]
        {total = 1; amounts = [("is", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("A", 1)]} ;
      Hashtbl.add table ["daughter"; "is"]
        {total = 1; amounts = [("proud", 1)]} ;
      Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
      Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["her"; "mom"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table [","; "and"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["the"; "shoulders"]
        {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["destiny"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["myths"; ","] {total = 1; amounts = [("and", 1)]} ;
      Hashtbl.add table ["kingdom"; "rests"]
        {total = 1; amounts = [("on", 1)]} ;
      Hashtbl.add table ["time"; "of"] {total = 1; amounts = [("magic", 1)]} ;
      Hashtbl.add table ["a"; "great"]
        {total = 1; amounts = [("kingdom", 1)]} ;
      Hashtbl.add table ["START"; "START"] {total = 1; amounts = [("In", 1)]} ;
      Hashtbl.add table ["on"; "the"]
        {total = 1; amounts = [("shoulders", 1)]} ;
      Hashtbl.add table ["magic"; ","] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table ["and"; "a"] {total = 1; amounts = [("time", 1)]} ;
      Hashtbl.add table ["of"; "magic"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["START"; "In"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["In"; "a"] {total = 1; amounts = [("land", 1)]} ;
      Hashtbl.add table ["man"; "."] {total = 1; amounts = [("STOP", 1)]} ;
      Hashtbl.add table ["of"; "myths"] {total = 1; amounts = [(",", 1)]} ;
      Hashtbl.add table ["the"; "destiny"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["rests"; "on"] {total = 1; amounts = [("the", 1)]} ;
      Hashtbl.add table [","; "the"] {total = 1; amounts = [("destiny", 1)]} ;
      Hashtbl.add table ["a"; "young"] {total = 1; amounts = [("man", 1)]} ;
      Hashtbl.add table ["of"; "a"]
        {total = 2; amounts = [("young", 1); ("great", 1)]} ;
      Hashtbl.add table ["shoulders"; "of"] {total = 1; amounts = [("a", 1)]} ;
      Hashtbl.add table ["a"; "land"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["young"; "man"] {total = 1; amounts = [(".", 1)]} ;
      Hashtbl.add table ["a"; "time"] {total = 1; amounts = [("of", 1)]} ;
      Hashtbl.add table ["great"; "kingdom"]
        {total = 1; amounts = [("rests", 1)]} ;
      { prefix_length ; table })]
```

Expected table 5 pts

```
 (let table = Hashtbl.create 36 in
  Hashtbl.add table ["land"; "of"]
    {total = 2; amounts = [("myths", 1); ("myths", 1)]} ;
  Hashtbl.add table ["START"; "A"] {total = 1; amounts = [("good", 1)]} ;
  Hashtbl.add table ["daughter"; "is"] {total = 1; amounts = [("proud", 1)]} ;
  Hashtbl.add table ["of"; "magic"]
    {total = 2; amounts = [(",", 1); (",", 1)]} ;
  Hashtbl.add table [","; "and"] {total = 2; amounts = [("a", 1); ("a", 1)]} ;
  Hashtbl.add table ["A"; "good"] {total = 1; amounts = [("daughter", 1)]} ;
  Hashtbl.add table ["of"; "her"] {total = 1; amounts = [("mom", 1)]} ;
  Hashtbl.add table ["the"; "shoulders"]
    {total = 2; amounts = [("of", 1); ("of", 1)]} ;
  Hashtbl.add table ["START"; "In"]
    {total = 2; amounts = [("a", 1); ("a", 1)]} ;
  Hashtbl.add table ["destiny"; "of"]
    {total = 2; amounts = [("a", 1); ("a", 1)]} ;
  Hashtbl.add table ["In"; "a"]
    {total = 2; amounts = [("land", 1); ("land", 1)]} ;
  Hashtbl.add table ["myths"; ","]
    {total = 2; amounts = [("and", 1); ("and", 1)]} ;
  Hashtbl.add table ["kingdom"; "rests"]
    {total = 2; amounts = [("on", 1); ("on", 1)]} ;
  Hashtbl.add table ["man"; "."]
    {total = 2; amounts = [("STOP", 1); ("STOP", 1)]} ;
  Hashtbl.add table ["proud"; "of"] {total = 1; amounts = [("her", 1)]} ;
  Hashtbl.add table ["time"; "of"]
    {total = 2; amounts = [("magic", 1); ("magic", 1)]} ;
  Hashtbl.add table ["of"; "myths"]
    {total = 2; amounts = [(",", 1); (",", 1)]} ;
  Hashtbl.add table ["a"; "great"]
    {total = 2; amounts = [("kingdom", 1); ("kingdom", 1)]} ;
  Hashtbl.add table ["mom"; "."] {total = 1; amounts = [("STOP", 1)]} ;
  Hashtbl.add table ["good"; "daughter"] {total = 1; amounts = [("is", 1)]} ;
  Hashtbl.add table ["rests"; "on"]
    {total = 2; amounts = [("the", 1); ("the", 1)]} ;
  Hashtbl.add table ["the"; "destiny"]
    {total = 2; amounts = [("of", 1); ("of", 1)]} ;
  Hashtbl.add table [","; "the"]
    {total = 2; amounts = [("destiny", 1); ("destiny", 1)]} ;
  Hashtbl.add table ["a"; "young"]
    {total = 2; amounts = [("man", 1); ("man", 1)]} ;
  Hashtbl.add table ["START"; "START"]
    {total = 3; amounts = [("In", 1); ("A", 1); ("In", 1)]} ;
  Hashtbl.add table ["on"; "the"]
    {total = 2; amounts = [("shoulders", 1); ("shoulders", 1)]} ;
  Hashtbl.add table ["of"; "a"]
    {total = 4;
     amounts = [("young", 1); ("great", 1); ("young", 1); ("great", 1)]} ;
  Hashtbl.add table ["magic"; ","]
    {total = 2; amounts = [("the", 1); ("the", 1)]} ;
  Hashtbl.add table ["shoulders"; "of"]
    {total = 2; amounts = [("a", 1); ("a", 1)]} ;
  Hashtbl.add table ["a"; "land"]
    {total = 2; amounts = [("of", 1); ("of", 1)]} ;
  Hashtbl.add table ["and"; "a"]
    {total = 2; amounts = [("time", 1); ("time", 1)]} ;
  {total = 2; amounts = [("of", 1); ("of", 1)]} ;
Hashtbl.add table ["great"; "kingdom"]
  {total = 2; amounts = [("rests", 1); ("rests", 1)]} ;
Hashtbl.add table ["is"; "proud"] {total = 1; amounts = [("of", 1)]} ;
table)
```