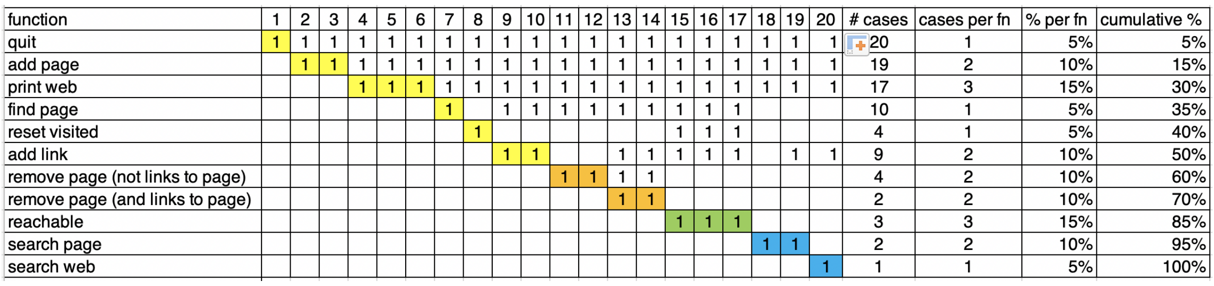
Computation I 5EIA1
C Exam: Search Engine (v1.4, NEW STYLE)
29 October 13:30-16:30/17:00

Important

- In this exam a predefined function is available for each task. For example `predefined_add_page`, `predefined_print_web`. Therefore, if you get stuck on a task or want to skip it then you can use the predefined function instead of your own function.
- If you use the predefined function for a task anywhere in your code then you will not get points for all of the test cases of that task. For example, if instead of writing your own `add_page` in Task 1 you use `predefined_add_page` in later tasks then you will not get the points for test cases 2 and 3 that are unique for Task 1. You will get points for the other test cases, e.g. those that use `predefined_add_page`.
- To use the predefined functions you need to include `#include "predefined.h"` in your program. This should be done automatically when you create a new .c file.

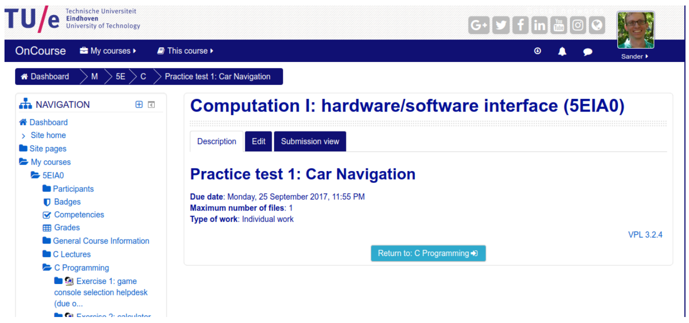
In this exam you will develop a competitor to Google! The user will be able to create web pages, link them, find which pages refer to each other, and search in them.



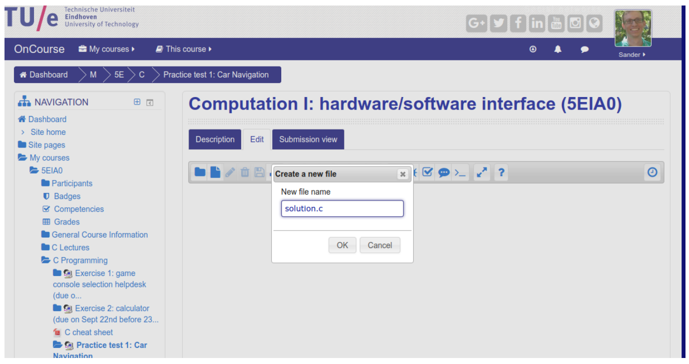
function	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	# cases	cases per fn	% per fn	cumulative %
quit	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	20	1	5%	5%
add page		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	19	2	10%	15%
print web				1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	17	3	15%	30%
find page							1		1	1	1	1	1	1	1	1	1				10	1	5%	35%
reset visited								1							1	1	1				4	1	5%	40%
add link									1	1			1	1	1	1	1		1	1	9	2	10%	50%
remove page (not links to page)											1	1	1	1							4	2	10%	60%
remove page (and links to page)													1	1							2	2	10%	70%
reachable															1	1	1				3	3	15%	85%
search page																		1	1		2	2	10%	95%
search web																				1	1	1	5%	100%

Figure 1: Test cases and points per task. The *remove page* (orange), *reachable* (green) and *search page* (blue) functions can be made independently of one another. It may be good to do the one you find easiest first.

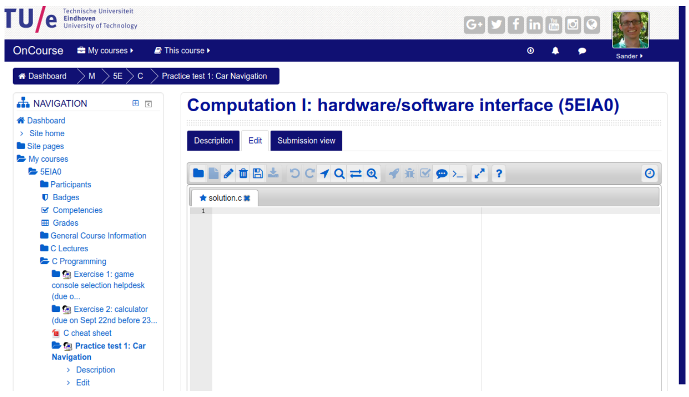
Open the correct C programming exam on exam.oncourse.tue.nl. You will see a screen similar to:


OnCourse
My courses
This course
Dashboard
Practice test 1: Car Navigation
NAVIGATION
Computation I: hardware/software interface (5EIA0)
Description
Edit
Submission view
Practice test 1: Car Navigation
Due date: Monday, 25 September 2017, 11:55 PM
Maximum number of files: 1
Type of work: Individual work
VPL 3.2.4
Return to: C Programming

Select the "Edit" tab and the following screen will appear:



Provide a useful name for your C file (e.g., solution.c) and press "Ok".
You will then see the following screen:



You can write your own C program in the text editor that is now shown in your browser. Once you press "Save" you can "Run" and "Evaluate" your program. Using the "Run" command you will see a terminal where you can provide input to your program. You can use this to debug your program. Using the "Evaluate" command all test cases are evaluated and at the end the results are displayed (grade and errors if present).

Task 1. In this exam you will develop a competitor to the Google! The user will be able to create web pages, link them, find which pages refer to each other, and search in them. Write a C program that asks the user to select the command that needs to be performed. The commands that need to be supported are listed in the following table:

command	operation
q	quit program
p	add page
w	print all pages on the web
f	find page
v	reset the visited flag on all pages
l	add link
P	remove page
r	find all reachable pages
s	search for a word in a page
S	search for a word in all pages on the web

In this task you only need to implement the quit command. In later tasks you will implement the remainder. If an invalid character is given then print the error message `Invalid command 'Z'` (with Z replaced by the unknown command):

```
Command? X
Invalid command 'X'
Command? q
Bye!
```

Your program must produce the *exact* output, including all spaces, capitalisation, quotes, etc.

Your program should now pass test case 1.

Task 2. The World Wide Web is a collection of pages each with a unique Uniform Resource Locator (URL). Your search engine will keep track of all pages in a linked list of `struct pages` that is ordered alphabetically by URL. Each `struct page` contains a fixed-size array of pointers to pages that it refers to.
The `predefined.h` header file contains the following data structure definition and predefined function declarations.

```
#define MAXLINKS 10
struct page {
  char *url; // stores the URL of the page
  int visited; // keep track of whether the page was visited
  struct page *links[MAXLINKS];
  struct page *next;
};
extern void predefined_print_web (struct page *web);
extern struct page *predefined_add_page (struct page *web, char *url);
extern struct page *predefined_find_page (struct page *web, char *url);
extern void predefined_reset_visited (struct page *web) ;
extern void predefined_add_link (struct page *web, char *source_url, char *dest_url);
extern void predefined_remove_page (struct page **web, char *url);
extern void predefined_reachable (struct page *page);
extern int predefined_search_page (struct page *page, char *keyword);
extern void predefined_search_web (struct page *web, char *keyword);
```

Include `#include "predefined.h"` in your program (this should have been done automatically). Define `struct page *web = NULL;` in your `main` function.

Define a function `struct page *add_page (struct page *web, char *url)` that inserts a new page with URL `url` at the correct position in the ordered linked list, and returns the updated list. The `visited` field should be initialised to `-1`, and all elements of the `links` array should be initialised to `NULL`.
Add the 'p' command to the main function to call the `add_page` function after asking for the page URL (use the `"%s"` format string). If the URL already exists on the web then give an error message as shown below.

```
Command? p
Page URL? virtualmachine.compsoc.eu
Command? p
Page URL? compsoc.eu
Command? p
Page URL? compsoc.eu
URL "compsoc.eu" is already on the web
Command? q
Bye!
```

Your program should now pass test cases 1 to 3. If you do not wish to implement the function `add_page` then you can use the `predefined_add_page` function, but you will not get the points for test cases 2 and 3.

Task 3. Next, let's print out all pages on the web with the 'w' command. Add the function "void print_web (struct page *web)" for this.

```
Command? w
Command? p
Page URL? whattheflux.com
Command? w
{URL="whattheflux.com",visited=-1} ->
Command? p
Page URL? atlas.mountains.org
Command? w
{URL="atlas.mountains.org",visited=-1} ->
{URL="whattheflux.com",visited=-1} ->
Command? p
Page URL? zelastone.de
Command? w
{URL="atlas.mountains.org",visited=-1} ->
{URL="whattheflux.com",visited=-1} ->
{URL="zelastone.de",visited=-1} ->
Command? q
Bye!
```

Note there's a space after the `->`. The URLs that the page refers should be printed after the `->` like this:

```
Command? w
{URL="atlas.mountains.org",visited=-1} ->
{URL="whattheflux.com",visited=-1} -> "zelastone.de"
{URL="zelastone.de",visited=-1} -> "whattheflux.com" "atlas.mountains.org"
Command?
```

However, we've not added any links yet. You can add the code to print links now or later, in Task 6.

Your program should now pass test cases 1 to 6. If you do not wish to implement the function `print_web` then you can use the `predefined_print_web` function, but you will not get the points for test cases 4-6.

Task 4. The 'f' command checks if there is a page on the web with a given URL. Implement the "struct page *find_page (struct page *web, char *url)" function that returns a pointer to the page or `NULL` otherwise. Example output is:

```
Command? w
Command? f
Page URL? emptyweb.com
URL "emptyweb.com" is not on the web
Command? p
Page URL? myfirstpage.woordpruts.com
Command? w
{URL="myfirstpage.woordpruts.com",visited=-1} ->
Command? f
Page URL? myfirstpage.woordpruts.com
URL "myfirstpage.woordpruts.com" is on the web
Command? q
Bye!
```

Your program should now pass test cases 1 to 7. If you do not wish to implement the function `find_page` then you can use the `predefined_find_page` function, but you will not get the points for test case 7.

Task 5. The visited field in each page will be used in a later task to keep track of whether we visited the page already. For now, implement the 'v' command that sets the visited field of all pages to zero. Define and use the function "void reset_visited (struct page *web)" for this.

```
Command? w
Command? v
Command? w
Command? p
Page URL? one.no
Command? w
{URL="one.no",visited=-1} ->
Command? v
Command? w
{URL="one.no",visited=0} ->
Command? p
Page URL? two.tw
Command? p
Page URL? three.tv
Command? w
{URL="one.no",visited=0} ->
{URL="three.tv",visited=-1} ->
{URL="two.tw",visited=-1} ->
Command? v
Command? w
{URL="one.no",visited=0} ->
{URL="three.tv",visited=0} ->
{URL="two.tw",visited=0} ->
Command? q
Bye!
```

Your program should now pass test cases 1 to 8. If you do not wish to implement the function reset_visited then you can use the predefined_reset_visited function, but you will not get the points for test case 8.

Task 6. The web wouldn't be a web without links. In this task implement the 'l' (links) command with the "void add_link (struct page *web, char *source_url, char *dest_url)" function. Ask for two URLs (source and destination) and then add the link from source to destination. This means you have to add a pointer to the destination page in the links field of the source page. Use the first NULL position in the array for the new link.
If either source or destination URL does not exist, if they are the same, or if the link already exists then give an error message (as shown below). When trying to insert a new page when all MAXLINKS links are used the error message "Maximum number of links reached" should be printed and the link should not be inserted.

Hint: The find_page function that you've written above is useful here.

See output on next page.

Your program should now pass test cases 1 to 10. If you do not wish to implement the function add_link then you can use the predefined_add_link function, but you will not get the points for test cases 9-10.

```
Command? p
Page URL? tracker.com
Command? p
Page URL? goggle.gr
Command? p
Page URL? fizzbook.fi
Command? p
Page URL? twotter.bu
Command? w
{URL="fizzbook.fi",visited=-1} ->
{URL="goggle.gr",visited=-1} ->
{URL="tracker.com",visited=-1} ->
{URL="twotter.bu",visited=-1} ->
Command? l
Source & destination URL? twotter.bu tracker.com
Command? w
{URL="fizzbook.fi",visited=-1} ->
{URL="goggle.gr",visited=-1} ->
{URL="tracker.com",visited=-1} ->
{URL="twotter.bu",visited=-1} -> "tracker.com"
Command? l
Source & destination URL? twotter.bu tracker.com
"tracker.com" is already a destination for "twotter.bu"
Command? l
Source & destination URL? twotter.bu fizzbook.fi
Command? w
{URL="fizzbook.fi",visited=-1} ->
{URL="goggle.gr",visited=-1} ->
{URL="tracker.com",visited=-1} ->
{URL="twotter.bu",visited=-1} -> "tracker.com" "fizzbook.fi"
Command? l
Source & destination URL? goggle.gr twotter.bu
Command? w
{URL="fizzbook.fi",visited=-1} ->
{URL="goggle.gr",visited=-1} -> "twotter.bu"
{URL="tracker.com",visited=-1} ->
{URL="twotter.bu",visited=-1} -> "tracker.com" "fizzbook.fi"
Command? l
Source & destination URL? twotter.bu twotter.bu
Source and destination URL cannot be the same
Command? l
Source & destination URL? no-page twotter.bu
Source URL "no-page" is not on the web
Command? l
Source & destination URL? twotter.bu no-page
Destination URL "no-page" is not on the web
Command? l
Source & destination URL? not1 not2
Source URL "not1" is not on the web
Command? q
Bye!
```

Task 7. Now we will implement the command 'P' with the function "`void remove_page (struct page **web, char *url)`". In this task the function only has to remove pages that have no links to them. (We complete the function in the next task.) Remove the page from the linked list and free the space used for `url` and the `struct page`. Give an error message if the URL does not exist.

```
Command? P
Page URL? no.page.pl
URL "no.page.pl" is not on the web
Command? p
Page URL? oncourse.tue.nl
Command? p
Page URL? exam.oncourse.tue.nl
Command? p
Page URL? canvas.tue.nl
Command? w
{URL="canvas.tue.nl",visited=-1} ->
{URL="exam.oncourse.tue.nl",visited=-1} ->
{URL="oncourse.tue.nl",visited=-1} ->
Command? P
Page URL? exam.oncourse.tue.nl
Command? w
{URL="canvas.tue.nl",visited=-1} ->
{URL="oncourse.tue.nl",visited=-1} ->
Command? P
Page URL? oncourse.tue.nl
Command? w
{URL="canvas.tue.nl",visited=-1} ->
Command? q
Bye!
```

Your program should now pass test cases 1 to 12. If you do not wish to implement the function `remove_page` then you can use the `predefined_remove_page` function, but you will not get the points for test cases 11-12.

Task 8. Extend the `remove_page` function such that it also correctly removes pages that have links to them from other pages (i.e. other pages refer to them in their `links` array). Set entries in the `links` array that refer to the page to be removed to `NULL`.

Hint: Note that `NULL` entries and non-`NULL` (pointing to other pages) may be arbitrarily interleaved in the `links` array. Recall that when inserting a new page it should use the first `NULL` entry. This is illustrated below.

```
Command? w
{URL="affligem.be",visited=-1} -> "heineken.nl" "duvel.be"
{URL="bochkarev.ru",visited=-1} ->
{URL="duvel.be",visited=-1} -> "affligem.be"
{URL="heineken.nl",visited=-1} ->
{URL="moretti.it",visited=-1} -> "heineken.nl"
Command? P
Page URL? heineken.nl
Command? w
{URL="affligem.be",visited=-1} -> "duvel.be"
{URL="bochkarev.ru",visited=-1} ->
{URL="duvel.be",visited=-1} -> "affligem.be"
{URL="moretti.it",visited=-1} ->
Command? l
Source & destination URL? affligem.be bochkarev.ru
Command? w
{URL="affligem.be",visited=-1} -> "bochkarev.ru" "duvel.be"
{URL="bochkarev.ru",visited=-1} ->
{URL="duvel.be",visited=-1} -> "affligem.be"
{URL="moretti.it",visited=-1} ->
Command? l
Source & destination URL? affligem.be moretti.it
Command? w
{URL="affligem.be",visited=-1} -> "bochkarev.ru" "duvel.be" "moretti.it"
{URL="bochkarev.ru",visited=-1} ->
{URL="duvel.be",visited=-1} -> "affligem.be"
{URL="moretti.it",visited=-1} ->
Command? q
Bye!
```

Your program should now pass test cases 1 to 14. If you do not wish to implement the function `remove_page` then you can use the `predefined_remove_page` function, but you will not get the points for test cases 13-14. Notice that the cases 11-14 for this task are independent from cases 15-17 (reachable) and case 18-20 (search).

Task 9. The 'r' command that uses the "`void reachable (struct page *page)`" function displays all the pages that are reachable from page. The pages reachable from page A are page A itself, all pages in its `links` list, and all pages reachable from those. Implement a recursive function that displays all pages reachable from page in a depth-first order. Given an error message (shown below) if the start page does not exist.

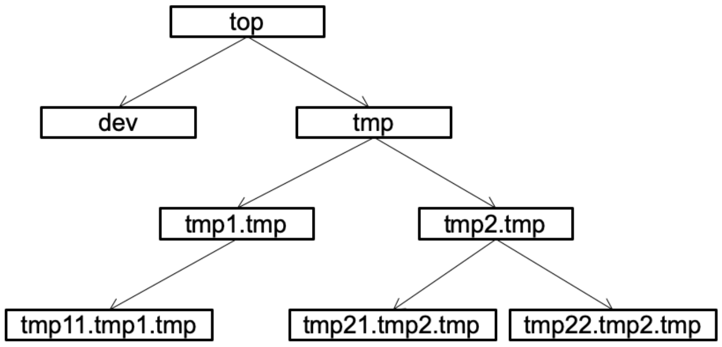


Figure 2: The web in the example output below (also shown by the first 'w' command).

```
Command? r
Page URL? hello.org
No such page "hello.org"
Command? ... insert all pages and links ...
Command? w
{URL="dev",visited=-1} ->
{URL="tmp",visited=-1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=-1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=-1} ->
{URL="tmp2.tmp",visited=-1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=-1} ->
{URL="tmp22.tmp2.tmp",visited=-1} ->
{URL="top",visited=-1} -> "dev" "tmp"
Command? r
Page URL? tmp11.tmp1.tmp
tmp11.tmp1.tmp
Command? r
Page URL? tmp1.tmp
tmp1.tmp
tmp11.tmp1.tmp
Command? r
Page URL? top
top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? q
Bye!
```

Hint: If the starting page exists, then you will need to first reset and then use the `visited` field.

Your program should now pass test cases 1 to 17. If you do not wish to implement the function `reachable` then you can use the `predefined_reachable` function, but you will not get the points for test cases 15-17. Notice that the cases 15-17 for this task are independent from cases 11-14 (remove page) and case 18-20 (search).

Task 10. We will now automate searching the entire web for a key word. Since the web is too large, the directory in which your program runs contains copies of only the following web pages:

- `alcohol.co.uk`

```
stones
xxxx
guinness stout lager
two-packets-of-crisps
glenfiddich
```

- `beers.be`

```
leffe
chimay chouffe
stella
```

- `drink4all.com`

```
stones guinness stout lager leffe
glenfiddich
bordeaux champagne
```

- `water.nl`

```
bar-le-duc
```

- `wines.fr`

```
chateau-neuf-du-pape
bordeaux
beaujolais
champagne burgundy shiraz zinfadel
condrieu
```

In your program you can open these files for reading using the filename `"beers.be"`, etc.
First implement the 's' command with the "`int search_page (struct page *page, char *keyword)`" function. The function opens the file with name equal to the URL (i.e. `page->url`, e.g. `"beers.be"`) in the directory in which the program runs. It then compares the `keyword` with all words in the file. The function returns 1 if the key word was found in the file and 0 otherwise. An error message is given if the file cannot be opened. This is an example output:

```
Command? w
Command? s
Search word in page? leffe beers.be
No page "beers.be"
Command? p
Page URL? beers.be
Command? w
{URL="beers.be",visited=-1} ->
Command? s
Search word in page? heineken beers.be
Page "beers.be" does not contain the word "heineken"
Command? s
Search word in page? leffe beers.be
Page "beers.be" contains the word "leffe"
Command? s
Search word in page? leffe no-beers.be
No page "no-beers.be"
Command? q
Bye!
```

Hint: As illustrated by the first commands a page must be added before you can search it.

Hint: Recall that successive words can be read from the standard input with `scanf("%s",&s);` automatically skipping spaces and newlines. Use something similar in your function.

Your program should now pass test cases 1 to 19. If you do not wish to implement the function `search_page` then you can use the `predefined_search_page` function, but you will not get the points for test cases 18-19. Notice that the cases 18-19 for this task are independent from cases 11-14 (remove page) and 15-17 (reachable).

Task 11. The final task is to search all pages on the web (command 'S') with the function "`void search_web (struct page *web, char *keyword)`".
In the example output `stella` is only in `beers.be`. `leffe` is in both `drink4all.com` and `beers.be`.

```
Command? p
Page URL? alcohol.co.uk
Command? p
Page URL? beers.be
Command? p
Page URL? drink4all.com
Command? w
{URL="alcohol.co.uk",visited=-1} ->
{URL="beers.be",visited=-1} ->
{URL="drink4all.com",visited=-1} ->
Command? S
Search web for word? stella
beers.be
Command? S
Search web for word? leffe
beers.be
drink4all.com
Command? S
Search web for word? guinness
alcohol.co.uk
drink4all.com
Command? S
Search web for word? grolsch
Command? q
Bye!
```

Your program should now pass test cases 1 to 20. If you do not wish to implement the function `search_web` then you can use the `search_web` function, but you will not get the points for test case 20. Notice that the case 20 for this task is independent from cases 11-14 (remove page) and 15-17 (reachable).

Input / output test cases

Case 01

Input:

```
q
```

Output:

```
Command? Bye!
```

Case 02

Input:

```
p
duckduckgo.com
p
tue.nl
p
itisgoodtobetheking.com
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Bye!
```

Case 03

Input:

```
p
bing.com
p
bing.com
p
bong.nl
p
bang.tv
q
```

Output:

```
Command? Page URL? Command? Page URL? URL "bing.com" is already on the web
Command? Page URL? Command? Page URL? Command? Bye!
```

Case 04

Input:

```
w
p
mountains.org
w
p
lakes.com
w
p
seas.de
w
q
```

Output:

```
Command? Command? Page URL? Command? {URL="mountains.org",visited=-1} ->
Command? Page URL? Command? {URL="lakes.com",visited=-1} ->
{URL="mountains.org",visited=-1} ->
Command? Page URL? Command? {URL="lakes.com",visited=-1} ->
{URL="mountains.org",visited=-1} ->
{URL="seas.de",visited=-1} ->
Command? Bye!
```

Case 05

Input:

```
w
p
ee.tue.nl
p
au.tue.nl
p
cs.tue.nl
p
wis.tue.nl
p
ieis.tue.nl
p
phy.tue.nl
p
computation.tue.nl
w
q
```

Output:

```
Command? Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? C
{URL="computation.tue.nl",visited=-1} ->
{URL="cs.tue.nl",visited=-1} ->
{URL="ee.tue.nl",visited=-1} ->
{URL="ieis.tue.nl",visited=-1} ->
{URL="phy.tue.nl",visited=-1} ->
{URL="wis.tue.nl",visited=-1} ->
Command? Bye!
```

Case 06

Input:

```
w
p
0
p
1
p
2
p
3
p
4
p
5
p
6
p
7
p
8
p
9
p
00
p
01
p
02
p
03
p
04
p
05
p
06
p
07
p
08
p
09
p
10
p
11
p
12
p
13
p
14
p
15
p
16
p
17
p
18
p
19
```

Output:

```
Command? Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? C
{URL="00",visited=-1} ->
{URL="01",visited=-1} ->
{URL="02",visited=-1} ->
{URL="03",visited=-1} ->
{URL="04",visited=-1} ->
{URL="05",visited=-1} ->
{URL="06",visited=-1} ->
{URL="07",visited=-1} ->
{URL="08",visited=-1} ->
{URL="09",visited=-1} ->
{URL="1",visited=-1} ->
{URL="10",visited=-1} ->
{URL="11",visited=-1} ->
{URL="12",visited=-1} ->
{URL="13",visited=-1} ->
{URL="14",visited=-1} ->
{URL="15",visited=-1} ->
{URL="16",visited=-1} ->
{URL="17",visited=-1} ->
{URL="18",visited=-1} ->
{URL="19",visited=-1} ->
{URL="2",visited=-1} ->
{URL="20",visited=-1} ->
{URL="21",visited=-1} ->
{URL="22",visited=-1} ->
{URL="23",visited=-1} ->
{URL="24",visited=-1} ->
{URL="25",visited=-1} ->
{URL="26",visited=-1} ->
{URL="27",visited=-1} ->
{URL="28",visited=-1} ->
{URL="29",visited=-1} ->
{URL="3",visited=-1} ->
{URL="30",visited=-1} ->
{URL="31",visited=-1} ->
{URL="32",visited=-1} ->
{URL="33",visited=-1} ->
{URL="34",visited=-1} ->
{URL="35",visited=-1} ->
{URL="36",visited=-1} ->
{URL="37",visited=-1} ->
{URL="38",visited=-1} ->
{URL="39",visited=-1} ->
{URL="4",visited=-1} ->
{URL="5",visited=-1} ->
{URL="6",visited=-1} ->
{URL="7",visited=-1} ->
{URL="8",visited=-1} ->
{URL="9",visited=-1} ->
Command? Bye!
```

Case 07

Input:

```
w
f
notfound.com
p
found.za
f
found.za
p
lost.br
p
last.ar
p
lust.fr
p
list.ru
p
lest.be
w
f
lost.br
f
last.ar
f
list.ru
f
lest.be
f
fles.be
f
found.za
f
lust.fr
f
l
w
q
```

Output:

```
Command? Command? Page URL? URL "notfound.com" is not on the web
Command? Page URL? Command? Page URL? URL "found.za" is on the web
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="last.ar",visited=-1} ->
{URL="lest.be",visited=-1} ->
{URL="list.ru",visited=-1} ->
{URL="lost.br",visited=-1} ->
{URL="lust.fr",visited=-1} ->
Command? Page URL? URL "lost.br" is on the web
Command? Page URL? URL "last.ar" is on the web
Command? Page URL? URL "list.ru" is on the web
Command? Page URL? URL "lest.be" is on the web
Command? Page URL? URL "fles.be" is not on the web
Command? Page URL? URL "found.za" is on the web
Command? Page URL? URL "lust.fr" is on the web
Command? Page URL? URL "l" is not on the web
Command? {URL="found.za",visited=-1} ->
{URL="last.ar",visited=-1} ->
{URL="lest.be",visited=-1} ->
{URL="list.ru",visited=-1} ->
{URL="lost.br",visited=-1} ->
{URL="lust.fr",visited=-1} ->
Command? Bye!
```

Case 08

Input:

```
w
p
first-page.domain.com
p
second-page.domain.com
p
third-page.domain.com
w
v
w
p
fourth-page.domain.com
w
q
```

Output:

```
Command? Command? Page URL? Command? Page URL? Command? Page URL? Command? {URL="first
{URL="second-page.domain.com",visited=-1} ->
{URL="third-page.domain.com",visited=-1} ->
Command? Command? {URL="first-page.domain.com",visited=0} ->
{URL="second-page.domain.com",visited=0} ->
{URL="third-page.domain.com",visited=0} ->
Command? Page URL? Command? {URL="first-page.domain.com",visited=0} ->
{URL="fourth-page.domain.com",visited=-1} ->
{URL="second-page.domain.com",visited=0} ->
{URL="third-page.domain.com",visited=0} ->
Command? Bye!
```

Case 09

Input:

```
p
deepweb.tor
p
darkweb.tor
p
examsolutions.darkweb.tor
p
humbug.tor
p
duckduckgo.com
w
l
duckduckgo.com
deepweb.tor
l
duckduckgo.com
darkweb.tor
l
darkweb.tor
examsolutions.darkweb.tor
l
humbug.tor
examsolutions.darkweb.tor
w
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="deepweb.tor",visited=-1} ->
{URL="duckduckgo.com",visited=-1} ->
{URL="examsolutions.darkweb.tor",visited=-1} ->
{URL="humbug.tor",visited=-1} ->
Command? Source & destination URL? Command? Source & destination URL? Command? Source
{URL="deepweb.tor",visited=-1} ->
{URL="duckduckgo.com",visited=-1} -> "deepweb.tor" "darkweb.tor"
{URL="examsolutions.darkweb.tor",visited=-1} ->
{URL="humbug.tor",visited=-1} -> "examsolutions.darkweb.tor"
Command? Bye!
```

Case 10

Input:

```
p
duckduckgo.com
w
l
duckduckgo.com
duckduckgo.com
l
no
yes
l
duckduckgo.com
no
l
yes
duckduckgo.co
p
ddg.com
l
ddg.com duckduckgo.com
w
l
ddg.com duckduckgo.com
w
p
10
p
20
p
30
p
40
p
50
w
l
10 20
l
10 30
l
20 30
l
10 40
l
40 10
l
40 20
l
40 30
l
20 40
l
30 50
l
50 30
w
q
```

Output:

```
Command? Page URL? Command? {URL="duckduckgo.com",visited=-1} ->
Command? Source & destination URL? Source and destination URL cannot be the same
Command? Source & destination URL? Source URL "no" is not on the web
Command? Source & destination URL? Destination URL "no" is not on the web
Command? Source & destination URL? Source URL "yes" is not on the web
Command? Page URL? Command? Source & destination URL? Command? {URL="ddg.com",visited=
{URL="duckduckgo.com",visited=-1} ->
Command? Source & destination URL? "duckduckgo.com" is already a destination for "ddg.
Command? {URL="ddg.com",visited=-1} -> "duckduckgo.com"
{URL="duckduckgo.com",visited=-1} ->
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="20",visited=-1} ->
{URL="30",visited=-1} ->
{URL="40",visited=-1} ->
{URL="50",visited=-1} ->
{URL="ddg.com",visited=-1} -> "duckduckgo.com"
{URL="duckduckgo.com",visited=-1} ->
Command? Source & destination URL? Command? Source & destination URL? Command? Source
{URL="20",visited=-1} -> "30" "40"
{URL="30",visited=-1} -> "50"
{URL="40",visited=-1} -> "10" "20" "30"
{URL="50",visited=-1} -> "30"
{URL="ddg.com",visited=-1} -> "duckduckgo.com"
{URL="duckduckgo.com",visited=-1} ->
Command? Bye!
```

Case 11

Input:

```
w
p
google
p
bing
p
ddg
p
altavista
p
startpage
w
P
altavista
w
P
startpage
w
P
ddg
w
P
google
w
p
again1
p
again2
p
google
p
altavista
w
q
```

Output:

```
Command? Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? C
{URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
{URL="startpage",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
{URL="startpage",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="google",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? {
{URL="again2",visited=-1} ->
{URL="altavista",visited=-1} ->
{URL="bing",visited=-1} ->
{URL="google",visited=-1} ->
Command? Bye!
```

Case 12

Input:

```
w
P
google
p
google
p
bing
p
ddg
p
altavista
p
startpage
w
P
altavista
w
P
startpage
w
P
ddg
w
P
google
w
p
again1
p
again2
p
google
w
P
google
w
P
google
p
altavista
w
q
```

Output:

```
Command? Command? Page URL? URL "google" is not on the web
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
{URL="startpage",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
{URL="startpage",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="ddg",visited=-1} ->
{URL="google",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
{URL="google",visited=-1} ->
Command? Page URL? Command? {URL="bing",visited=-1} ->
Command? Page URL? Command? Page URL? Command? Page URL? Command? {URL="again1",visite
{URL="again2",visited=-1} ->
{URL="bing",visited=-1} ->
{URL="google",visited=-1} ->
Command? Page URL? Command? {URL="again1",visited=-1} ->
{URL="again2",visited=-1} ->
{URL="bing",visited=-1} ->
Command? Page URL? URL "google" is not on the web
Command? Page URL? Command? {URL="again1",visited=-1} ->
{URL="again2",visited=-1} ->
{URL="altavista",visited=-1} ->
{URL="bing",visited=-1} ->
Command? Bye!
```

Case 13

Input:

```
p
100
p
200
p
300
p
400
p
500
w
l
100 200
l
100 300
l
100 400
l
100 500
w
P
400
w
P
200
w
P
300
w
p
200
p
300
p
400
p
600
w
P
100
w
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Source & destination URL? Command? Source & destination URL? Command? Source
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "200" "300" "500"
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "300" "500"
{URL="300",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "500"
{URL="500",visited=-1} ->
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? {
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
{URL="600",visited=-1} ->
Command? Page URL? Command? {URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
{URL="600",visited=-1} ->
Command? Bye!
```

Case 14

Input:

```
P
100
p
100
p
200
p
300
p
400
p
500
w
l
100 200
l
100 300
l
100 400
l
100 500
w
P
400
w
p
400
l
100
400
w
P
200
w
P
300
w
p
300
p
200
w
l
100 300
l
100 200
w
P
200
P
300
P
400
P
500
w
P
100
w
q
```

Output:

```
Command? Page URL? URL "100" is not on the web
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Source & destination URL? Command? Source & destination URL? Command? Source
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "200" "300" "500"
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? Source & destination URL? Command? {URL="100",visited=-1}
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "300" "400" "500"
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? {URL="100",visited=-1} -> "400" "500"
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? Page URL? Command? {URL="100",visited=-1} -> "400" "500"
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Source & destination URL? Command? Source & destination URL? Command? {URL="1
{URL="200",visited=-1} ->
{URL="300",visited=-1} ->
{URL="400",visited=-1} ->
{URL="500",visited=-1} ->
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? {
Command? Page URL? Command? Command? Bye!
```

Case 15

Input:

```
p
top
p
dev
p
tmp
p
tmp1.tmp
p
tmp2.tmp
p
tmp11.tmp1.tmp
p
tmp21.tmp2.tmp
p
tmp22.tmp2.tmp
l
top dev
l
top tmp
l
tmp tmp1.tmp
l
tmp tmp2.tmp
l
tmp1.tmp tmp11.tmp1.tmp
l
tmp2.tmp tmp21.tmp2.tmp
l
tmp2.tmp tmp22.tmp2.tmp
w
r
dev
r
tmp11.tmp1.tmp
r
tmp1.tmp
r
tmp
r
top
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="tmp",visited=-1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=-1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=-1} ->
{URL="tmp2.tmp",visited=-1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=-1} ->
{URL="tmp22.tmp2.tmp",visited=-1} ->
{URL="top",visited=-1} -> "dev" "tmp"
Command? Page URL? dev
Command? Page URL? tmp11.tmp1.tmp
Command? Page URL? tmp1.tmp
tmp11.tmp1.tmp
Command? Page URL? tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Page URL? top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Bye!
```

Case 16

Input:

```
r
where
p
top
p
dev
p
tmp
p
tmp1.tmp
p
tmp2.tmp
p
tmp11.tmp1.tmp
p
tmp21.tmp2.tmp
p
tmp22.tmp2.tmp
l
top dev
l
top tmp
l
tmp tmp1.tmp
l
tmp tmp2.tmp
l
tmp1.tmp tmp11.tmp1.tmp
l
tmp2.tmp tmp21.tmp2.tmp
l
tmp2.tmp tmp22.tmp2.tmp
w
r
top
l
tmp1.tmp tmp2.tmp
w
r
tmp11.tmp1.tmp
r
tmp2.tmp
r
tmp
r
top
q
```

Output:

```
Command? Page URL? No such page "where"
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="tmp",visited=-1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=-1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=-1} ->
{URL="tmp2.tmp",visited=-1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=-1} ->
{URL="tmp22.tmp2.tmp",visited=-1} ->
{URL="top",visited=-1} -> "dev" "tmp"
Command? Page URL? top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Source & destination URL? Command? {URL="dev",visited=1} ->
{URL="tmp",visited=1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=1} -> "tmp11.tmp1.tmp" "tmp2.tmp"
{URL="tmp11.tmp1.tmp",visited=1} ->
{URL="tmp2.tmp",visited=1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=1} ->
{URL="tmp22.tmp2.tmp",visited=1} ->
{URL="top",visited=1} -> "dev" "tmp"
Command? Page URL? tmp11.tmp1.tmp
Command? Page URL? tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Page URL? tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Page URL? top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Bye!
```

Case 17

Input:

```
p
loop1
p
loop2
l
loop1 loop2
l
loop2 loop1
w
r
loop1
p
top
p
dev
p
tmp
p
tmp1.tmp
p
tmp2.tmp
p
tmp11.tmp1.tmp
p
tmp21.tmp2.tmp
p
tmp22.tmp2.tmp
l
top dev
l
top tmp
l
tmp tmp1.tmp
l
tmp tmp2.tmp
l
tmp1.tmp tmp11.tmp1.tmp
l
tmp2.tmp tmp21.tmp2.tmp
l
tmp2.tmp tmp21.tmp2.tmp
l
tmp2.tmp tmp22.tmp2.tmp
l
tmp21.tmp2.tmp tmp22.tmp2.tmp
l
tmp22.tmp2.tmp tmp21.tmp2.tmp
l
tmp11.tmp1.tmp tmp1
w
r
dev
r
top
w
r
dev
r
tmp11.tmp1.tmp
r
tmp21.tmp2.tmp
```

Output:

```
Command? Page URL? Command? Page URL? Command? Source & destination URL? Command? Sour
{URL="loop2",visited=-1} -> "loop1"
Command? Page URL? loop1
loop2
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
Command? Source & destination URL? Command? Source & destination URL? Command? Source
Command? {URL="dev",visited=-1} ->
{URL="loop1",visited=1} -> "loop2"
{URL="loop2",visited=1} -> "loop1"
{URL="tmp",visited=-1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=-1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=-1} ->
{URL="tmp2.tmp",visited=-1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=-1} -> "tmp22.tmp2.tmp"
{URL="tmp22.tmp2.tmp",visited=-1} -> "tmp21.tmp2.tmp"
{URL="top",visited=-1} -> "dev" "tmp"
Command? Page URL? dev
Command? Page URL? top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? {URL="dev",visited=1} ->
{URL="loop1",visited=0} -> "loop2"
{URL="loop2",visited=0} -> "loop1"
{URL="tmp",visited=1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=1} ->
{URL="tmp2.tmp",visited=1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=1} -> "tmp22.tmp2.tmp"
{URL="tmp22.tmp2.tmp",visited=1} -> "tmp21.tmp2.tmp"
{URL="top",visited=1} -> "dev" "tmp"
Command? Page URL? dev
Command? Page URL? tmp11.tmp1.tmp
Command? Page URL? tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Page URL? tmp1.tmp
tmp11.tmp1.tmp
Command? Page URL? tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? Page URL? top
dev
tmp
tmp1.tmp
tmp11.tmp1.tmp
tmp2.tmp
tmp21.tmp2.tmp
tmp22.tmp2.tmp
Command? {URL="dev",visited=1} ->
{URL="loop1",visited=0} -> "loop2"
{URL="loop2",visited=0} -> "loop1"
{URL="tmp",visited=1} -> "tmp1.tmp" "tmp2.tmp"
{URL="tmp1.tmp",visited=1} -> "tmp11.tmp1.tmp"
{URL="tmp11.tmp1.tmp",visited=1} ->
{URL="tmp2.tmp",visited=1} -> "tmp21.tmp2.tmp" "tmp22.tmp2.tmp"
{URL="tmp21.tmp2.tmp",visited=1} -> "tmp22.tmp2.tmp"
```

Case 18

Input:

```
p
beers.be
p
wines.fr
p
alcohol.co.uk
p
drink4all.com
w
s
leffe beers.be
s
leffe wines.fr
s
bordeaux wines.fr
s
bordeaux drink4all.com
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? {
{URL="beers.be",visited=-1} ->
{URL="drink4all.com",visited=-1} ->
{URL="wines.fr",visited=-1} ->
Command? Search word in page? Page "beers.be" contains the word "leffe"
Command? Search word in page? Page "wines.fr" does not contain the word "leffe"
Command? Search word in page? Page "wines.fr" contains the word "bordeaux"
Command? Search word in page? Page "drink4all.com" contains the word "bordeaux"
Command? Bye!
```

Case 19

Input:

```
p
beers.be
p
wines.fr
p
alcohol.co.uk
p
drink4all.com
l
drink4all.com beers.be
l
drink4all.com wines.fr
l
alcohol.co.uk beers.be
l
alcohol.co.uk wines.fr
w
s
water teetotal.fi
w
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? S
{URL="beers.be",visited=-1} ->
{URL="drink4all.com",visited=-1} -> "beers.be" "wines.fr"
{URL="wines.fr",visited=-1} ->
Command? Search word in page? No page "teetotal.fi"
Command? {URL="alcohol.co.uk",visited=-1} -> "beers.be" "wines.fr"
{URL="beers.be",visited=-1} ->
{URL="drink4all.com",visited=-1} -> "beers.be" "wines.fr"
{URL="wines.fr",visited=-1} ->
Command? Bye!
```

Case 20

Input:

```
p
water.nl
p
beers.be
p
wines.fr
p
alcohol.co.uk
p
drink4all.com
l
drink4all.com water.nl
l
drink4all.com beers.be
l
drink4all.com wines.fr
l
beers.be alcohol.co.uk
l
wines.fr alcohol.co.uk
l
drink4all.com alcohol.co.uk
l
alcohol.co.uk beers.be
w
S
leffe
S
bordeaux
S
stones
S
stoned
S
guinness
S
bar-le-duc
q
```

Output:

```
Command? Page URL? Command? Page URL? Command? Page URL? Command? Page URL? Command? P
{URL="beers.be",visited=-1} -> "alcohol.co.uk"
{URL="drink4all.com",visited=-1} -> "water.nl" "beers.be" "wines.fr" "alcohol.co.uk"
{URL="water.nl",visited=-1} ->
{URL="wines.fr",visited=-1} -> "alcohol.co.uk"
Command? Search web for word? beers.be
drink4all.com
Command? Search web for word? drink4all.com
wines.fr
Command? Search web for word? alcohol.co.uk
drink4all.com
Command? Search web for word? Command? Search web for word? alcohol.co.uk
drink4all.com
Command? Search web for word? water.nl
Command? Bye!
```