WORKSHEET

A12.9 Web technologies and search

Networks and the web · A level · OCR H446 1.3.4 · about 30 min

Name _____ Class _____ Date _____

What this lesson is about

HTML, CSS and JavaScript, client and server side processing, indexing and PageRank.

- HTML
- CSS
- JavaScript
- Client side and server side
- Search engine indexing
- PageRank

Questions

6 marks in all

1. Which CSS selector styles every element that has class="note"? [1 mark]

- ☐ **A** .note
- ☐ **B** #note
- ☐ **C** note
- ☐ **D** <note>

2. Which HTML tag makes a hyperlink? [1 mark]

- ☐ **A** <a href="...">
- ☐ **B** <link>
- ☐ **C** <p>
- ☐ **D**

3. In the PageRank formula with d = 0.85, page A is linked to only by page B. B has a rank of 2 and 4 outbound links. What is PR(A)? Give it to 3 decimal places. [1 mark]

4. Which are reasons to process a form on the server rather than only in the browser? Choose all that apply. [1 mark]

Tick every answer that is true.

- ☐ **A** Client-side code can be seen, changed or switched off by the user
- ☐ **B** The server can check the data against its database
- ☐ **C** It gives instant feedback with no request
- ☐ **D** It reduces the load on the server

5. What does this program print? [1 mark]

```
index = {}
pages = {'p1': 'robot kit', 'p2': 'robot sim', 'p3': 'kit list'}
for page, words in pages.items():
    for word in words.split():
        index.setdefault(word, []).append(page)
print(index['kit'])
```

---

6. What does a search engine's crawler do? [1 mark]

- ☐ **A** Visits pages and follows their links to find pages to add to the index
- ☐ **B** Ranks pages by how many words they contain
- ☐ **C** Blocks pages that contain malware
- ☐ **D** Runs JavaScript on the server

The task: index and rank

A small website has five pages. `pages` maps each page name to its text; `links` maps each page name to the list of pages it links to. Every page links to at least one other. 1. Build an index: a dictionary mapping each word (split the text on spaces) to the list of pages that contain it, with no page listed twice for one word. 2. Work out the PageRank of every page with d = 0.85. Start every page at 1.0 and do exactly 30 iterations, calculating each iteration's ranks only from the previous iteration's ranks. 3. For each page, in the order of `pages`, print `<page> <rank>` with the rank to 2 decimal places. 4. For each of the words `robot` and `simulator`, in that order, print `search <word>: <pages>`, the pages containing that word from highest rank to lowest, separated by single spaces. That is 7 lines. The robot does not drive.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

pages = {
    "home": "bugbot lessons for your robot in python",
    "sim": "the robot simulator runs python in the browser",
    "kit": "buy a robot kit for your school",
    "blog": "news about the simulator and the kit",
    "faq": "questions about python and the robot",
}
links = {
    "home": ["sim", "kit", "faq"],
    "sim": ["home"],
    "kit": ["home", "sim"],
    "blog": ["sim", "kit"],
    "faq": ["home", "sim"],
}
D = 0.85
rank = {page: 1.0 for page in pages}
```

Plan your program here, then type it in and press Run.

Do it on the robot

www.bugbotlab.com/learn/a12-9-web-technologies-and-search/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Add a page that links to `faq` only. Predict which ranks change before you run it.
2. Stop iterating when no rank changes by more than 0.0001, and print how many iterations that took.
3. Write the HTML for a page that lists the five pages as links, and CSS that shows the highest-ranked page's link in bold.