JavaScript : Pagination

Implementació de lògica en el component *Pagination* de Bootstrap

Aquest és el component que usarem:
https://getbootstrap.com/docs/5.0/components/pagination/

HTML

Dins la «section» trobem varis «div», cadascun d'ells conté informació, un cop hagueu finalitzat el programa, aquesta informació únicament serà mostrada quan el botó de la llista de paginació corresponent sigui clickat, la llista està actualment buida ja que els seus elements es generen dinàmicament a través de JavaScript.
Aquí pots trobar el fitxer HTML base per realitzar l'activitat:
https://github.com/xbaubes/DAM/tree/main/MP4-Llenguatges-marques-SGI/JavaScript/exercise-pagination/part1

CSS

Els elements amb classe «infoOcult» han d'estar ocults.
Recorda que els estils de la paginació ja els entrega Bootstrap.

JS

Integrem i atorguem funcionalitat al component.
Aquestes funcions separen la terminologia emprada per Bootstrap dels elements HTML que nosaltres hem creat.
Implementa, com a mínim, les següents funcions:

```
function createPageButton(pageNumber, classes)
```

pageNumber: Nom que mostrarà el botó, en aquest exercici serà un dígit de 1 a 5.
classes: String amb les classes opcionals que inclourà l'etiqueta «<li>», en aquest exercici serà «active» o string buit.
Retorna un string (o un element) corresponent a una etiqueta «<li>» i amb una etiqueta «<a>» al seu interior preparats per ser afegit a una llista.

> Amb aquesta funció es defineix un botó de canvi de pàgina amb Bootstrap. Així és un element standard de la llista:
>
> ```
> <li class="page-item">
>   <a class="page-link" href="#">page number</a>
> </li>
> ```
>
> El botó que correspongui a la pàgina mostrada tindrà la classe «active» a l'etiqueta «<li>».

```
function initializePageButtons(nButtons, paginationContent)
```

nButtons: Nombre de botons de paginació que s'han de generar, en aquesta part de l'exercici seran 5.
paginationContent: Element de tipus llista «ul» on afegirem els elements «li», en aquest exercici és l'element amb id «llistaPagines».

A l'iniciar-se el programa afegim els botons indicats a la llista de paginació, tingues en compte que en l'estat inicial la paginació es troba a la pàgina 1.
Els crea usant la funció *createPageButton* i els afegeix al DOM.

```
function eventsCreation (paginationContent, contentPages, hiddenClass)
```

paginationContent: Element contenidor de totes les etiquetes «a» que es troben dins les etiquetes «li» de la llista de paginació, en aquest exercici és l'element amb id «llistaPagines».
contentPages: Llista amb totes les pàgines contenidores, en aquest exercici són els «div» amb classe «info» dins de «#ContenidorInfo».
hiddenClass: String identificador de classe per ocultar les pàgines que no s'ha demanat que es mostrin, en aquest exercici serà «#infoOcult». Aquest valor s'ha de passar com a paràmetre ja que no forma part de la selecció de pàgina, que és el que ara estem realitzant.

Generem els EventListener que capturaran i realitzaran accions quan es clicki qualsevol dels botons de paginació. Pots utilitzar un bucle per generar-los.
El que faran els EventListener és mostrar el contenidor sol·licitat per l'usuari i ocultar la resta, això es fa a través de la classe «infoOcult» dels «div». Per altre banda, ressaltaran el botó clickat afegint a l'etiqueta «li» la classe «active», com que només pot haver-hi un botó clickat s'eliminarà la classe de qualsevol altre botó. En la llista de paginació l'etiqueta «li» és el pare de l'etiqueta «a».

```
function createPagination(paginationContent, nButtons, contentPages, hiddenClass)
{
	initializePageButtons(nButtons, paginationContent);
	eventsCreation (paginationContent, contentPages, hiddenClass);
}
```

Còpia la següent funció. Aquesta funció actua com a interfície al permetre crear una paginació cridant una sola funció i que aquesta cridi a les dos funcions principals.

Obtenció del contingut d'un fitxer CSV per entregar-lo en diferents pàgines (part 2)

CSV

Aquesta pàgina web ha de permetre llegir i mostrar qualsevol fitxer .csv que contingui el següent:

- Una capçalera indicativa del contingut de les columnes.
- Un separador de columnes únic i no usat en les dades.
- Un separador de files únic i no usat en les dades.

El següent fitxer compleix amb totes les premisses:
https://github.com/xbaubes/DAM/blob/main/MP4-Llenguatges-marques-SGI/JavaScript/exercise-pagination/part2/oscar_age_male.csv

HTML

Disposa d'una etiqueta que permet seleccionar el fitxer .csv del que volem veure el contingut, section «#SelectorCSV».
«#ContenidorInfo» inclourà el contingut del .csv fragmentat en diferents elements,està actualment buit ja que es generà dinàmicament a través de JavaScript.
Deixa d'interactuar directament amb la llibreria que hem creat i passa a importar com a mòdul un fitxer intermediari. El fet de ser un mòdul ens obliga a executar el codi des d'un servidor.
Aquí pots trobar el fitxer HTML base per realitzar l'activitat:
https://github.com/xbaubes/DAM/blob/main/MP4-Llenguatges-marques-SGI/JavaScript/exercise-pagination/part2/index.html

CSS

L'estil és lliure, procura atorgar un aspecte vistós al contenidor d'informació i a la pàgina en general.
Aquí pots trobar el fitxer CSS de la pàgina principal amb els imports als CSS de les llibreries:
https://github.com/xbaubes/DAM/blob/main/MP4-Llenguatges-marques-SGI/JavaScript/exercise-pagination/part2/main.css

JS

Per separar les diferents implementacions s'afegeix un fitxer *main.js* des del que es cridaran les funcions de les nostres llibreries *pagination.js* i *csv.js*.
Aquí pots trobar el fitxer JS de la pàgina principal:
https://github.com/xbaubes/DAM/blob/main/MP4-Llenguatges-marques-SGI/JavaScript/exercise-pagination/part2/main.js

Implementa, com a mínim, les següents funcions:

```
function createOneDivInfo(pos,classes)
```

pos: Valor numèric que indica la pàgina a la que correspon el contenidor i que formarà part de l'id de l'element.
classes: String amb el valor «infoOcult» si l'element no s'ha de veure i «» si s'ha de veure.
Retorna un string (o un element) corresponent a un dels contenidors de la informació que canviaran segons la pàgina seleccionada.

Tal com vam veure a la primera part, aquesta és la estructura mínima que hem de generar i retornar:

```
<div id="info1" class="info">
```

Les pàgines que no corresponguin a la seleccionada en la paginació inclouran la classe «infoOcult».

```
function createAllDivInfo(numPages,classes)
```

numPages: Quantitat de blocs d'informació que s'han de generar. Dependrà del volum de línies a mostrar i del nombre de línies per pàgina especificat.
classes: String amb el valor «infoOcult» si l'element no s'ha de veure i «» si s'ha de veure.
Retorna un string (o conjunt d'elements) per poder inserir el DOM tots els contenidors necessaris per mostrar la informació del fitxer .csv.

Genera tants blocs d'informació com indiqui *numPages*, els crea usant la funció *createOneDivInfo* i els retorna.

```
function createRowContent(rowSplitted)
```

rowSplitted: Array amb la informació de la línia separada en columnes.
Retorna un string (o conjunt d'elements) per poder inserir el DOM una línia del .csv.

Aquesta funció ha d'evitar que s'insereixin files buides del .csv al contenidor. També podria encarregar-se d'eliminar la columna índex, si existeix.
La informació es pot mostrar de vàries maneres, per exemple la pots estructurar com una taula o usant divs.

```
function processCSV(pagesContainer, CSVContent, hiddenClass, linesByPage, useIndex, colSeparator, rowSeparator)
```

pagesContainer: Contenidor de les pàgines amb la informació .csv (#ContenidorInfo).
CSVContent: Contingut del fitxer .csv seleccionat.
hiddenClass: String identificador de classe per ocultar les pàgines que no s'ha demanat que es mostrin, en aquest exercici serà «#infoOcult». És important declarar una sola vegada els elements i llavors passar-los per evitar duplicitats.
linesByPage: Valor numèric que indica el nombre de línies del fitxer .csv que es mostraran per cada pàgina.
useIndex: Booleà que indica si el fitxer .csv utilitza índex a la primera columna o no.
colSeparator: String usat com a separador de columnes de la fila.
rowSeparator: String usat com a separador de files.
Retorna un array amb els elements «.info» generats.

Funció d'entrada de la llibreria, des d'aquí es criden les demés funcions.
Realitza o delega les següents tasques:

- Separar el contingut del .csv en línies.
- Separar el contingut corresponent a la capçalera de la resta.
- Descartar l'índex si és necessari.
- Afegir la capçalera a l'HTML.
- Calcular el nombre de pàgines que és necessari generar per poder mostrar tota la informació.
- Afegir les pàgines on apareix la informació a l'HTML.
- Afegir la informació a l'HTML.