|  |  |
| --- | --- |
| Curso: | Jurimetria |
| Disciplina: | Raspagem (webscraping) de tribunais de justiça e adminstrativos |

**Prezado(a) professor(a),**

Estamos cadastrando todas as questões da **Atividade Final** da disciplina em nossa Fábrica de Provas, pois não teremos mais a prova ao final do curso e todas as atividades serão aplicadas de forma online. Para isso, precisamos cumprir o seguinte padrão na construção das questões:

- Elaborar 15 questões objetivas (com **até** 5 alternativas de resposta);

- Os enunciados devem estar com numeração arábica (1., 2., 3...) automática do Word;

- As alternativas devem estar classificadas automaticamente com letras (a., b., c. e d.);

- A **única** alternativa correta deve estar destacada em amarelo;

- Indicar com o negrito se o aluno deve marcar a alternativa **correta** ou a **incorreta**, quando necessário;

- A justificativa da resposta deve vir após o termo (COMENTÁRIOS) e **não deve indicar a letra da alternativa correta**, pois, assim como as questões, as alternativas também serão randomizadas a cada prova.

1- Quais são, nesse curso, as principais etapas da raspagem de dados de uma página web:

a. Interpretação, inspeção e extração.

b. Iteração, coleta e inspeção.

c. Visualização, requisição, parseamento.

d. Requisição, extração e iteração.

e. Coleta, transformação e análise.

(Comentários)

O curso dividiu o processo de raspagem em três etapas: requisição, extração e iteração.

2 – O que significa dizer que a requisição http é sem estado (stateless)?

a. Ela não requer conexão com a internet.

b. Ela não mantêm memória entre requisições.

c. As conexões entre cliente e servidor são persistentes.

d. Armazena dados do usuário.

e. É mais rápida quando as requisições ao mesmo servidor são repetidas.

(Comentários)

A requisição http não mantêm memória entre requisições. Ela requer conexão com a internet. Não há persistência algumas, pois não há memória. Ela não armazena dado do usuário, quem pode fazer isso são os cookies e outro parâmetros. Como não há memória, ela não se beneficia de requisições repetidas ao mesmo servidor.

3. Quanto ao cookies nas requisições http.

a. São pequenos textos enviados pelo servidor e armazenados no navegador do usuário(cliente).

b. Eles não têm relevância alguma.

c. Eles protegem as informações dos usuários.

d. São arquivos de imagem muito grandes.

e. Eles preservam a memória da navegação pelo usuário.

(Comentários).

Cookies em requisição http são pequenos arquivos de texto criados pelo servidor e armazenados pelo navegador do usuário a fim de preservar a memória da navegação. Eles não protegem as informações dos usuários. Ao contrário, eles podem ser usados para obter, por vezes indevidamente, informações dos usuários.

4. Quais as duas formas de requisição http mais usadas:

a. UPLOAD e EXTRACT

b. GET e FORM

c. GET e POST

d. DELETE e PUT

e. DELETE e GET

(Comentários)

Embora DELETE e PUT também sejam requisições http, as mais comuns e usadas são GET e POST.

5. Que método você usaria para solicitar recursos de um servidor sem ter de preencher alguma formulário?

a. POST

b. DELETE

c. GET

d. EXTRAT

e. PUT

­

(Comentários)

O método GET é aquele que não requer envio de dados no corpo/formulário para solicitar recursos.

6. Qual a principal diferença entre os métodos HTTP GET e POST:

a. A quantidade de dados enviados numa requisição.

b. A forma como os dados são enviados numa requisição.

c. A maneira como se processa uma requisição.

d. A forma como os dados são processados pelo cliente.

e. Nenhuma das respostas anteriores

(Comentários)

Na requisição GET, os dados são enviados como parâmetros de um URL. Na requisição POST eles são enviados no corpo da requisição.

1. O que são cabeçalhos ou headers em uma requisição HTTP?

a. Os dados que são enviados no corpo da requisição.

b. Informações adicionais que são enviadas na requisição.

1. O tipo de requisição HTTP que está sendo feita.
2. Informações enviadas no corpo/body da requisição.
3. Qual é a sintaxe básica de um XPath?
4. /tag
5. //tag
6. tag[]
7. @tag
8. ///tag

(Comentários)

Duas barras antes do elemento ou tag indicam para ir diretamente aquele tag, saltando elementos diferentes anteriores a ele.

1. Como você selecionaria todos os elementos "p" dentro de uma "div"?
   1. //div/p
   2. //p/div
   3. /div/p
   4. //div[@p]
   5. /div//p

(Comentários)

Lembre-se sempre que o primeiro elemento deve ser precedido de duas barras. O próximo, se estiver imediatamente dentro do anterior, requer apenas uma barra. No entanto, você pode usar duas barras se quiser.

1. Como você selecionaria o elemento "h1" que é o primeiro filho de uma "div" com a classe "conteúdo"?
2. //div/@class="conteúdo"/h1[1]
3. //div[h1][@class="conteúdo"][1]
4. //div[h1][@class="conteúdo"]
5. /div/h1[1][@class="conteúdo"]
6. //div[@class="conteúdo"]/h1[1]

(Comentários)

Possivelmente, há mais de um “div” com vários “h1” dentro dele. O atributo “classe=’conteúdo’” serve justamente para distinguir esse “div” de outros. [1] depois de “h1” indica para pegar apenas o primeiro “h1”.

1. O que é o pacote purrr do R?
   1. Um pacote para trabalhar com dados espaciais
   2. Um pacote para criar gráficos interativos
   3. Um pacote para programação funcional em R
   4. Um pacote para análise de dados de séries temporais

(Comentários)

O purrr é um pacote para programação funcional, ou seja, suas funções tomam outras funções como argumento.

1. Se eu tenho a seguinte lista com os vetores a e b dentro dela: x <- list(a = c(1,5,8), b = c(8,3,2)), e usar a função map do pacote purrr para calcular a média de cada um dos vetores, como devo chamar essa função no R.
2. map(x, mean(a, b))
3. map(x, mean(.x))
4. map(x, ~mean(.x))
5. map(a, b, mean(x))
6. map(x, mean(.x))

(Comentários)

Para usar map, você tem de colocar primeiramente o objeto sobre o qual irá iterar, no caso a lista x e em seguida a função que será aplicada sobre cada elemento. No entanto, deve colocar um til antes dela, o qual indica uma função anônima. Dentro da função, no caso mean, use .x (ponto x) para indicar cada elemento de x, ou seja, cada vetor.

1. Se eu tenho a seguinte lista com os vetores a e b dentro dela: x <- list(a = c(1,5,8), b = c(8,3,2)), e usar a função map do pacote purrr para calcular a média de cada um dos vetores, que resultado irei obter:
2. 4.66 e 4.33
3. 3.44 e 6.33
4. 3.11 e 5.22
5. 3.44 e 6.33
6. 4.33 e 3.66

(Comentários)

1. Qual a função do pacote purrr mais adequada para obter como resultado final um dataframe a partir do empilhamento de outros dataframes criados a cada iteração?
2. map
3. map\_chr
4. map\_dfr
5. walk
6. walk2

(Comentários)

A função map\_dfr serve para empilhar dataframes parciais em um único dataframe final. É muito útil em jurimetria.

1. Quando eu quero iterar sobre cada elemento de um objeto, mas não quero nenhum que retorne nenhum objeto, apenas salvar cada resultado parcial em disco, qual a função do purrr mais apropriada:
2. walk
3. reduce
4. map
5. map\_dfr
6. map\_dfc

(Comentários)

A função walk não retorna objeto algum, ela tem apenas efeitos colaterais. Ela é muito útil para gerar muitos gráficos ou salvar resultados parciais em arquivos.