Respostas "Atividade 4 Fases"

Fase 1:

1) b → length retorna a quantidade de elementos da lista
2) c → pop( ) removeu o ultimo item da lista
3) a → shift( ) removeu o primeiro item da lista
4) b → unshift( ) adicionou um item no inicio da lista
5) b → slice( ) fez criar na lista b uma parte da lista a, a partir do item no espaço 1
6) b → splice( ) retirou os itens nos espaços 1 e 2. O programa retornou os itens mantidos na lista
7) c → indexOf( ) ao receber 'c' como parâmetro fez com que o programa retorne a posição de 'c' na lista.
8) b → o metodo map( ) percorreu cada item da lista itens impondo uma condição, onde a lista r é formado per estas condições.
9) c → Graças ao método filter( ), a lista filtrado é formada pelos numeros da lista nums que são maiores que 6
10) d → Graças ao reduce( ), a lista soma é formada por um unico valor, sendo a soma dos itens da lista arr.
11) a → include( ) faz retornar true ou false caso o item citado esteja na lista referenciada
12) c → join( ) junta todos os itens de uma lista em uma única string, separados por um delimitador, no caso o -
13) c → Graças ao concat( ), a lista arr agora é formada pela junção dos itens já existentes com nova lista trazida no parametro do método.
14) b → reverse( ) inverte a ordem dos itens da lista
15) c → Graças ao find( ), r será formada pelo primeiro item que corresponde a condição imposta pelo método na lista dados.

Fase 2:

16) filter() manteve apenas os números ímpares da lista; map() percorreu esses números, impondo sua condição; reduce() somou todos os valores, a partir do 10

17) Duves do filter, usei um if dentro do for pra selecionar apenas os multiplos de 3; daí criei manualmente o obj dentro do laço pra suprir a falta do map. New no lugar do reduce acumulei o valor de metade na variavel processado (+=).

18) O método splice remove os números nos espaços 1 e 2 da lista "lista", formando a lista r com esses números removidos. Uma forma de evitar a mutação é usar slice().

19) A solução foi criar um objeto agrupado por cidade, onde usei o reduce pra percorrer cada objeto do array e o acumulador "acc" começando com { }. Daí pra cada pessoa eu via se já existe chave com o nome da cidade, onde eu criava uma lista vazia caso não houvesse e colocava a pessoa nesta lista.

20) A função find retorna o primeiro elemento que satisfaz a condição, pegando o primeiro valor igual a 8 no código; O filter retorna uma nova lista com todos os elementos que entram na condição, pegando todos os valores iguais a 8; Caso haja algum valor igual a 8 no código, a função retorna true ou false, nesse caso foi retornado true.

Fase 3:

21) a → O programa usou a desestruturação para separar os itens no objeto (no caso a propriedade "nome") e o restante integrou (por meio do spread) um novo objeto "resto".

22) a → O programa desestruturou a lista numeros, fazendo com que "a" receba o item 1 e "b" receba o item 3. O item 2 foi ignorado por causa do espaço entre as virgulas. "c" usou spread pra abrigar a lista com os iteus 4 e 5

23) a → O código fez uma desestruturação, tirando "cidade" dentro do objeto "endereco". Dessa forma, "cidade" recebeu "RJ" sem precisar guardar o "endereco" inteiro.

24) b → Esse código usou spread pra combinar "base" e "extra" na formação do objeto "combinado". Daí quando há chaves repetidas, prevalece o último objeto espalhado.

25) b → A função recebeu {x:5}, então x deixou de valer 10 pra valer 5 e y continou valendo 20.

26) c → "Arr3" foi montado com a yunção dos elementos das outras duas listas, sem alteração e com 05 no final.

27) a → O código renomeou a propriedade "tema" para "modo" e agrupou o restante em "opcoes". Daí o console exibiu "escuro" e depois "Arial" (vindo do opcoes.fonte.).

28) b → O spread distribuiu os valores da lista como parâmetros da função, e o "resto" yuntou os extras em outra lista.

29) a → A desestruturação acessou diretamente a lista "habilidades", atribuindo o primeiro e o terceiro valores.

30) b → O spread fez uma cópia de "obj1", aew "obj1.b" e "obj2.b" apontam pro mesmo objeto. Por isso, ao alterar "obj2b.x" o valor em "obj1.b.x" tambem mudou pra 99.

Fase 4:

31) b → O código desestruturou a matriz, ignorou o primeiro item, pegou o valor 4 do y da segunda lista e agrupou o restante em [[5,6]].

32) a → O código extraiu "tema="escuro"" de "preferencias" e agrupou o restante em "resto".

33) a → Criei um novo objeto "resto" com as propriedades restantes de "dados", então ter alterado "resto.c" não afetou "dados.c".

34) a → O código desestruturou "arr", pegando "x=1" e "resto=[2,3]. Mesmo como "y" não foi passado, assumiu o padrão "[...resto, x]=[2,3,1]".

35) c → O código fez um cópia apenas do objeto "opções", daí "config1" e "config2" tem objetos diferentes nessa chave.