Questão 01

Qual é a principal vantagem dos dados assíncronos?

a) Eles são mais rápidos do que dados síncronos.
b) Eles retornam um valor imediatamente após a chamada de função.
c) Eles não param a execução da thread principal para obter uma resposta.
d) Eles chamam uma função de callback quando o dado está disponível.
e) Eles permitem a criação de aplicações desacopladas.

Questão 02

Qual é o padrão de projeto utilizado para a programação reativa no Angular?

a) Callback.
b) Promise.
c) Async.
d) Observable.
e) Não há padrão reativo em angular.

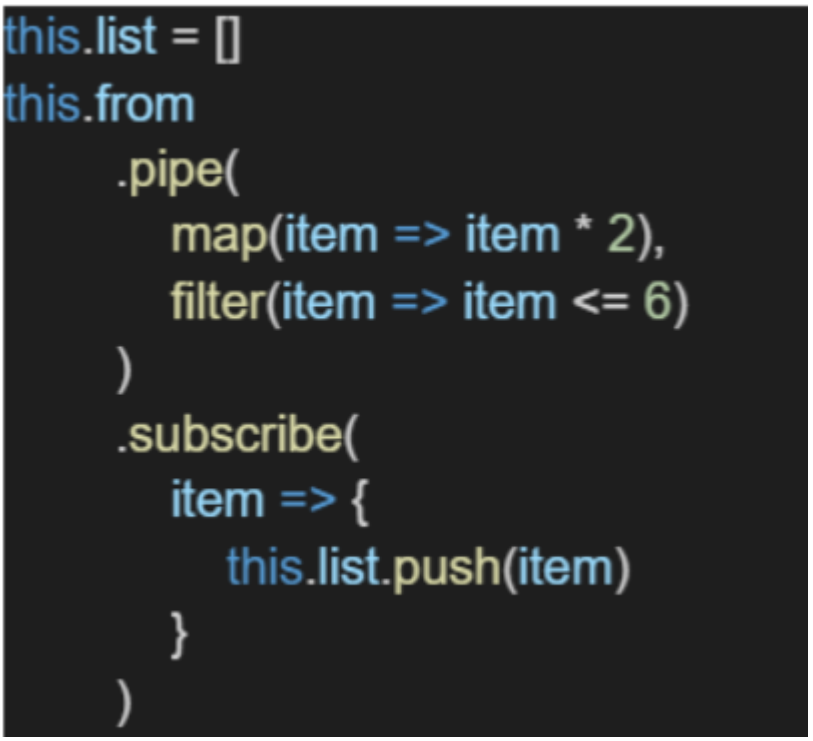
Questão 03

Podemos citar como valores agregados da utilização do RXJS com angular, exceto:

a) Redução do código fonte.
b) Facilidade de programação assíncrona.
c) Reaproveitamento de conhecimento em diversas linguagens.
d) Melhoria da performance da aplicação.
e) Permite a manipulação do stream de dados usando diversas técnicas com o uso dos pipes.

Questão 04

Observe o código fonte abaixo:



```
this.list = []
this.from
      .pipe(
         map(item => item * 2),
         filter(item => item <= 6)
      )
      .subscribe(
         item => {
            this.list.push(item)
         }
      )
```

Qual alternativa melhor descreve o comportamento deste código?

a) O código empilha os itens da função map e filter e retorna um subscribe.
b) O código adiciona duas funções ao stream de dados, a primeira de transformação do valor para o dobro do número, a segunda filtrando valores menores que 6, em seguida o método escuta os valores emitidos pelo stream e para cada item adiciona na lista.
c) O código encadeia as funções de mapeamento de um dado a outro, dobrando seu valor, e faz um filtro para obter os itens menores que 6, tudo de forma síncrona e retorna um subscribe.
d) O código executa a função pipe recebendo como parâmetro duas funções que serão executadas em paralelo para cada item do stream, em seguida adiciona os itens na lista.
e) O método executa a função pipe no stream de dados from, passando como argumentos duas funções de mapeamento, em seguida escuta cada elemento com o subscribe e adiciona na lista.

Questão 05

Qual é o objetivo do construtor "interval" do RXJS?

a) Criar um stream de inteiros.
b) Emitir eventos indefinidamente com intervalos de tempo fixos.
c) Criar um observável a partir de eventos emitidos em elementos.
d) Combinar itens dois a dois e acumular o resultado.
e) Retornar o último item omitido.

Questão 06

Leia o texto a seguir:

“Ao criar observáveis que emitem mais de um evento, o framework não possui recursos para saber de antemão quando o stream será fechado e quando não há mais classes interessadas no item, ele irá continuar emitindo eventos, o que pode ocasionar em vazamentos de memória e impactar na performance da aplicação.”

Dado o contexto acima, qual a importância do gerenciamento de memória através do unsubscribe ou uso do asyn pipe em aplicações Angular nos casos em que o stream não é fechado automaticamente?

a) O unsubscribe do rxjs é desnecessário em aplicações Angular, pois a memória é automaticamente liberada ao navegar entre componentes e páginas.
b) É uma preocupação válida e relevante para os sistema em angular, mas não é relevante em outros tipos de aplicações.
c) O unsubscribe é inútil em aplicações Angular pois o garbage collector já é o suficiente para liberar a memória.
d) O unsubscribe do rxjs pode ser feito de qualquer forma, desde que o objeto não seja mais referenciado na aplicação.
e) Permite que a aplicação funcione de forma contínua com performance ótima, pois o angular é construído para aplicação single page onde a pagina ou aplicativo raramente é recarregada, liberando espaços em memória.