CONCURSO PÚBLICO
MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO DE JANEIRO

EDITAL Nº 006/2022

PADRÃO DE RESPOSTAS DA PROVA DISCURSIVA REALIZADA DOMINGO, 15 DE MAIO DE 2022.
PRAZO PARA RECURSO CONTRA O PADRÃO DE RESPOSTAS: 16 E 17 DE MAIO DE 2022, NO ENDEREÇO ELETRÔNICO:
http://www.selecon.org.br

PADRÃO DE RESPOSTAS PRELIMINAR

NIT – 02

Desenvolvimento de Aplicativos Multiplataforma e Jogos Digitais com *Engines*

Nº DA QUESTÃO	Espera-se que o candidato(a) desenvolva os aspectos/conteúdos propostos a seguir.
1	O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos: a) A resposta deve dissertar que no Unity, os *game objects* vão sendo compostos e construídos através de inclusão de componentes com diferentes características, fazendo com que os game objects fiquem cada vez mais específicos. Pode-se dar um exemplo concreto, embora não é obrigatório. O player tem a característica de ser controlado por um usuário, portanto deve ter o Character Controller ou algo semelhante. Além disso, por ser 3D, ele deve ter componentes de renderização (Mesh Renderer) e de física (Rigid Body e algum *collider,* embora caso mencione que há o componente Character Controller, estes podem ser suprimidos) para este tipo de cenário. Finamente, como ele tem diferentes animações, deve haver um componente que controle o estado destas (animator). **(4,0 pontos)**

	b) O candidato deverá desenhar e/ou explicar como é a máquina de estados para as animações mencionadas acima, portanto deve conter todas estas ações: andar, pular, correr, rastejar, agachar, atirar. Seria ideal que ele também coloque as funções de transição de um estado para outro, porém não é obrigatório. **(3,0 pontos)** c) Deseja-se aqui que seja explicado por alto o funcionamento do algoritmo A*. O ideal seria que o candidato coloque um pseudocódigo, mas basta que ele explique que o A* é um algoritmo baseado em árvore de decisão e que estima heuristicamente a direção para onde o objeto deve ir (fazendo um vetor de direção simples, por exemplo), escolhendo o galho da árvore com menor custo. O Unity implementa o *pathfinding* através dos componentes de *navigation*. **(3,0 pontos)** Total previsto de linhas para a resposta final do(a) candidato(a): **60 linhas**
2	O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos: a) Espera-se que o aluno disserte sobre este tipo de estrutura de dados, mencionando que o grafo de cena contém objetos de cena nos nós e que existe um relacionamento hierárquico entre os nós. O grafo de cena assemelha-se a uma estrutura de árvore. **(5,0 pontos)** b) O problema é resolvido escalonando a transformação em questão através do tempo transcorrido no game loop. No caso do Unity, utiliza-se o deltaTime ou a função FixedUpdate () dos game objects. **(5,0 pontos)** Total previsto de linhas para a resposta final do(a) candidato(a): **60 linhas**
3	O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos: a) Espera-se que o candidato explique cada um dos frameworks JS abordando suas características principais: • NodeJS*: framework* que permite a execução de código Javascript (JS) no servidor (server-side). Dessa forma, *o framework* habilita o uso de JS para o desenvolvimento de aplicações *backend*, como o desenvolvimento de APIs. • ReactJS: criado para resolver problemas geralmente não resolvidos no JS, como a criação de componentes e gerenciamento de estados. A estrutura de componentes permite a criação de blocos de código reutilizáveis em diversas partes da aplicação. O React resolve problemas de performance através do Virtual DOM. Usado principalmente para desenvolvimento interfaces Web com processamento no cliente (client-side) inclusive Progressive Web Apps (PWA). • React Native: criado a partir do ReacJS, o objetivo deste *framework* é gerar código nativo para ser executado em sistemas operacionais móveis iOS e Android. Devido à geração de código nativo, os apps criados com React Native são mais fluidos e possuem integrações mais completas com recursos do sistema operacional móvel. Diferente do React, no React Native não são utilizadas *tags* HTML, mas sim *tags* nativas referente aos componentes providos pela plataforma. **(3,0 pontos)** b) Promise: uma promessa é um objeto que representa o resultado de uma computação assíncrona. Esse resultado pode ou não estar pronto ainda, você só pode pedir ao Promise para chamar uma função de retorno de call-back apenas quando o valor estiver pronto.

Portanto, no nível mais simples, as promessas são apenas uma maneira diferente de trabalhar com retornos de *callback*. No entanto, existem benefícios práticos em usá-los. Um problema real com a programação assíncrona baseada em retorno de *callback* é que é comum acabar com retornos de *callback* dentro de retornos de *callback*, com linhas de código tão recuadas que é difícil de ler. As promessas permitem que esse tipo de retorno de *callback* aninhado sejam reescritos como uma cadeia de promessas mais linear que tende a ser mais fácil de ler e raciocinar.

- Async/await: O ES2017 apresenta duas novas palavras-chave — async e await — que representam uma mudança de paradigma na programação JavaScript assíncrona. Essas novas palavras-chave simplificam drasticamente o uso de Promises e nos permitem escrever código assíncrono baseado em Promise que se parece com código síncrono que bloqueia enquanto aguarda respostas de rede ou outros eventos assíncronos. Embora ainda seja importante entender como as Promises funcionam, muito de sua complexidade de código desaparece quando você as usa com async e await. Como qualquer código que usa await é assíncrono, há uma regra crítica: você só pode usar a palavra-chave await dentro de funções que foram declaradas com a palavra-chave async**. (4,0 pontos)**

c)

```
async function getAluno() {
  try {
    let response = await fetch("/api/aluno/matriculas.json");
    let data = await response.json();
    console.log(data);
  } catch (err) {
    console.log(err);
  }
}

getAluno();
```

(3,0 pontos)

Total previsto de linhas para a resposta final do(a) candidato(a): **60 linhas**