

INSTITUTO FEDERAL
RIO DE JANEIRO



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.

PADRÃO DE RESPOSTAS OFICIAL

EPF – 04
INFORMÁTICA
Programação

Nº DA QUESTÃO	Espera-se que o candidato(a) desenvolva os aspectos/conteúdos propostos a seguir.
1	<p>O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos:</p> <ul style="list-style-type: none">A) Apresentar o conceito de <i>Activity</i> como um paradigma não determinístico de execução de aplicações. (1 ponto)B) Apresentar o conceito de <i>Activity</i> implícita e explícita, ressaltando o seu funcionamento e o baixo acoplamento da <i>Activity</i> com a aplicação. (2 pontos)C) Discutir sobre as funções de <i>call-back</i> de uma <i>Activity</i> bem como a sua ordem de execução. (3 pontos)D) Apresentar as condições e eventos gerados no sistema operacional Android que causam uma mudança de estado de uma <i>Activity</i> bem como as funções de <i>call-back</i> que são invocadas durante a transição entre estados. (2 pontos)E) Comentar sobre como o consumo de memória e energia estão relacionados com o uso de <i>call-back</i> e <i>Activities</i> durante a execução de uma aplicação. (2 pontos)

Total previsto de linhas para a resposta final do(a) candidato(a): **40 linhas**.

A experiência em aplicativos para dispositivos móveis é diferente da versão para computador, porque a interação do usuário com a aplicação nem sempre começa no mesmo lugar. Em vez disso, a jornada do usuário geralmente começa de forma não determinista. Por exemplo, se você abrir um aplicativo de e-mails na tela inicial, poderá ver uma lista de e-mails. Por outro lado, se você estiver usando um aplicativo de mídia social que inicialize seu aplicativo de e-mails, poderá ir diretamente para a tela do aplicativo de e-mails para escrever uma mensagem.

Uma **Activity** é definida como uma tela em aplicações baseadas no sistema operacional Android, fornecendo uma área na qual o aplicativo desenha a própria IU. Dessa forma, uma **Activity** serve como ponto de entrada para a interação de um aplicativo com o usuário.

A maioria dos aplicativos contém várias telas, o que significa que elas abrangem várias **Activities**. Podem ser invocadas *explicitamente* por uma outra **Activity** na mesma aplicação ou de forma *implícita*. Neste último caso, a aplicação solicita ao sistema operacional uma **Activity** capaz de prover serviços e funcionalidades previamente registradas de forma compartilhada para outras aplicações.

Embora as **Activities** funcionem juntas para formar uma experiência do usuário coesa em um aplicativo, cada uma é vagamente vinculada às outras, reduzindo assim o acoplamento. Geralmente, há dependências mínimas entre as **Activities** em um aplicativo. Frequentemente elas iniciam **Activities** pertencentes a outros apps. Por exemplo, um aplicativo de navegação pode iniciar a **Activity** "Compartilhar" de um aplicativo de mídia social.

Ao longo da vida útil de uma **Activity**, ela passa por vários estados. Uma série de call-backs são usados para lidar com transições entre estados, conforme listado abaixo:

- **onCreate**: acionado quando o sistema cria uma **Activity**. Responsável por inicializar os componentes essenciais da **Activity**, alocando memória do sistema. Coloca a **Activity** no estado "Criada".
- **onStart**: invocado após o evento **onCreate**, colocando a **Activity** no estado "Visível" e servindo para a realização dos preparativos finais da atividade antes de ir para o primeiro plano e se tornar interativa.
- **onResume**: invocado imediatamente antes da **Activity** começar a interagir com o usuário. Neste ponto, a **Activity** fica na parte superior da pilha de atividades e captura toda a entrada do usuário, implementando a maior parte da funcionalidade principal de um aplicativo. Coloca a **Activity** no estado "Resumida".
- **onPause**: acionado quando a **Activity** perde o foco e entra no estado "Pausado". Ainda permite a atualização da UI da aplicação, uma vez que ainda está visível.

	<ul style="list-style-type: none"> • onStop: invocado quando a Activity não está mais visível para o usuário. Isso pode acontecer porque a atividade está sendo destruída (a fim de liberar memória e reduzir o processamento), uma nova Activity está sendo iniciada ou uma Activity existente está entrando em um estado "<i>Retomado</i>" e está cobrindo a Activity interrompida. • onRestart: invocado quando uma Activity no estado "<i>Interrompido</i>" está prestes a ser reiniciada. Restaura o estado da Activity a partir do momento em que ela foi interrompida. • onDestroy: O sistema invoca esse call-back antes de uma Activity ser destruída, colocando a Activity no estado "<i>Destruída</i>". Normalmente esse estado é invocado com o objetivo de liberar recursos computacionais (como a memória de sistema e consumo de energia).
2	<p>O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos:</p> <p>A) Item 2.1 (3 pontos)</p> <p>B) Explicar o sentido de acoplamento de código_____ (3 pontos)</p> <p>C) Explicar como diminuir o acoplamento utilizando interface_____ (4 pontos)</p> <p>Total previsto de linhas para a resposta final do(a) candidato(a): 32 linhas.</p> <p>Item 2.1</p> <pre>public class CCartao : ICartao { public string numeroCartao { get; set; } }</pre> <p>Item 2.2</p> <p>Explicar o sentido de acoplamento de código: é a relação de dependência entre códigos de diferentes classes que, quando é forte, dificulta o reaproveitamento destes, tornando-se difícil de compreender e, conseqüentemente, difícil de se implementar mudanças, sendo assim, sua redução é desejável.</p> <p>Explicar como diminuir o acoplamento utilizando interface: programando para interfaces e não para classes concretas. Na prática, deve-se substituir as declarações de classes concretas por declarações de interfaces, o que elimina a dependência de uma implementação específica, tornando o código dependente apenas da interface.</p>
3	<p>O candidato deverá desenvolver o(s) conteúdo(s) com base nos seguintes aspectos:</p> <p>A) Definição _____ (2 pontos)</p>

- B) Algoritmo de Busca _____ (2 pontos)
C) Operação de Inserção _____ (2 pontos)
D) Operação de Exclusão _____ (2 pontos)
E) ABB depois da inserção _____ (1 ponto)
F) ABB depois das Exclusões _____ (1 ponto)

Definição: As árvores binárias de busca apresentam uma relação de ordem entre os nós. Essa ordem é definida pela chave. Uma árvore binária T é uma árvore binária de busca se:

- chaves da sub-árvore esquerda de T são menores do que chave da raiz de T;
- chaves da sub-árvore da direita de T são maiores do que a chave da raiz de T;
- e sub-árvores da esquerda e da direita de T são árvores binárias de busca.

Busca:

```
busca (arvore r, int k) {  
    if (r == NULL || r->chave == k)  
        return r;  
    if (r->chave > k)  
        return busca (r->esq, k);  
    else  
        return busca (r->dir, k);  
}
```

Inserção: Se a árvore for vazia, instala o novo nó na raiz. Se não for vazia, compara a chave com a chave da raiz:

- se for menor, instala o nó na sub-árvore da esquerda
- caso contrário, instala o nó na sub-árvore da direita

A ordem em que as chaves são inseridas numa árvore de busca binária pode fazer com que uma árvore binária se deteriore, ficando com altura muito grande. Sabendo disso, é possível reordenar as chaves de entrada de forma a obter uma árvore o mais balanceada possível. Algoritmo:

Seja v um vetor ORDENADO contendo as chaves a serem inseridas
Inserir a chave do meio
Chamar recursivamente para os dois pedaços que sobraram.

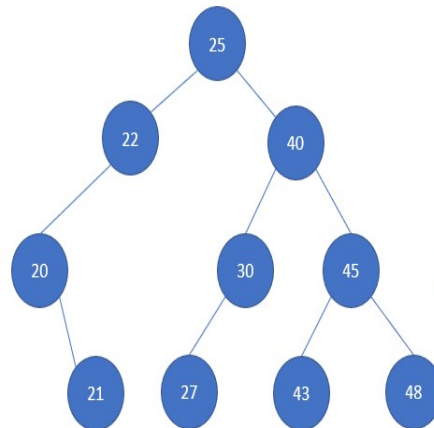
Exclusão: Exclusão Física possui 3 casos:

Caso 1 - Nó é folha: Quando o nó a ser excluído é uma folha, basta removê-lo. **Caso 2** - Nó não folha com uma sub-árvore: Raiz da sub-árvore passa a ocupar o lugar do nó excluído. **Caso 3** - nó não folha com duas sub-árvores: Reestruturar a árvore. Utiliza-se uma estratégia Recursiva:

- 1) Trocar o valor do nó a ser removido com:
 - a) valor do nó que tenha a maior chave da sua sub-árvore à esquerda
 - b) OU valor do nó que tenha menor chave da sua sub-árvore à direita
- 2) Ir à sub-árvore onde foi feita a troca e remover o nó

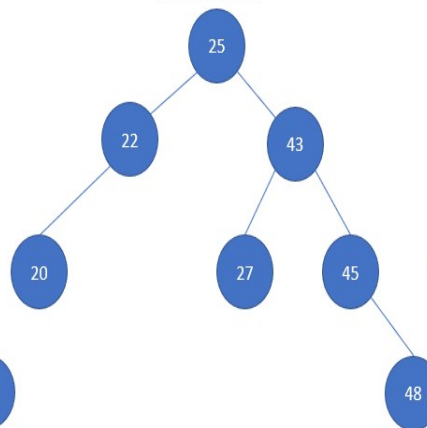
Desenho das duas ABBs solicitadas no enunciado:

Inserir em uma ABB inicialmente vazia,
os seguintes valores:
25, 22, 40, 30, 45, 27, 43, 20, 21, 48

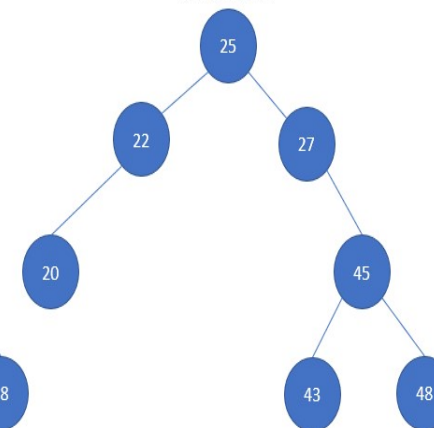


Excluir os nós 21, 30 e 40

Método 1



Método 2



Total previsto de linhas para a resposta final do(a) candidato(a): **32 linhas.**

