

Questão 1)

Um dos passos mais importantes de qualquer projeto é a instalação de sua infraestrutura, dito isso, qual o comando usamos para subir o servidor local?

Dica: nosso projeto já deve ter sido criado!

Selecione uma alternativa

A

`npm install -g @angular/cli`

B

`ng new alurapic`

C

`ng --version`

D

`ng serve --open`

Questão 2)

Utilizando o Angular, temos acesso à sua poderosa capacidade de *Data binding*. Mas existem algumas diferenças em relação ao seu uso!

Marque abaixo a opção correta.

Selecione uma alternativa

A

Utilizamos [] dentro de tags e {{ }} para atributos!

B

Utilizamos {{ }} dentro de tags e [] para atributos

C

Utilizamos -- -- dentro de tags e () para atributos!

Questão 3

Vimos que existe uma convenção para a nomenclatura dos arquivos e componentes do Angular. Dito isso, qual o padrão que devemos seguir?

Selecione uma alternativa

A

Para arquivos usamos `menubarcomponent.ts`, no nome usamos `menubarcomponent`

B

Para arquivos usamos `menubarComponent.ts`, no nome usamos `Menubar.component`

C

Para arquivos usamos `menubar.component.ts`, no nome usamos `MenubarComponent`

D

Para arquivos usamos `MenubarComponent`, no nome usamos `menubar.component.ts`

Questão 4)

Vejamos a declaração da seguinte classe:

```
class Abc { }
```

Ela é apenas uma classe do ES6 e não caracteriza um componente em Angular!

Qual das opções abaixo a torna efetivamente um componente?

Selecione uma alternativa

A

```
import {Component} from '@angular/core';

@Component({
  selector: './abc.component.html',
  templateUrl: 'my-abc'
})
class AbcComponent { }
```

B

```
import {Component} from '@angular/core';

@Component({
  selector: 'my-abc' ,
  templateUrl: './abc.component.html'
})
class AbcComponent { }
```

C

```
import {Component} from '@angular/core';

class AbcComponent { }
```

Questão 5)

Marque a alternativa verdadeira sobre a propriedade @Input:

Selecione uma alternativa

A

Permite que o componente receba valores externos quando usado na forma declarativa no template de outros componentes.

B

Todo componente precisa ter, obrigatoriamente, todas as propriedades da classes decoradas com o decorator `Input`.

C

Todo componente já permite, por padrão, receber dados através da sua forma declarativa. Quando usamos o decorator `Input` estamos apenas explicitando nossa intenção.

Questão 6)

Marque as opções verdadeiras à respeito da criação de componentes:

Selecione 2 alternativas

A

Componentes declarados no array `declarations` de um módulo são visíveis para os componentes também declarados no array.

B

Um componente obrigatoriamente precisa pertencer a um módulo.

C

Componentes declarados no array `declarations` são automaticamente visíveis para componentes pertencentes à outros módulos.

Questão 7)

Como comumente é chamado o servidor que tem a responsabilidade de fornecer os dados para a aplicação Angular em uma *Single Page Application*?

Selecione uma alternativa

- A** Angular Server
- B** Web Server
- C** Single Page Server
- D** Web API

Questão 8)

Qual das alternativas abaixo ilustra como deve ser feito o *import* do `HttpClient` em nosso código?

Selecione uma alternativa

- A** Devemos fazer: `import {HttpClient} from '@angular/http';`
- B** Após adicionarmos o `import` em nosso `app.module.ts`, precisamos importar em nosso `component` o seguinte: `import {HttpClient} from '@angular/common/http';`
- C** Basta importar em nosso `component` o seguinte: `import {HttpClient} from '@angular/common/http';`

Questão 9)

A Ana está trabalhando com Angular e precisa criar um novo componente. Ela está usando o Angular CLI para essa tarefa e digitou o comando:

```
ng generate component users/user-list
```

O comando executou com sucesso, mas em qual pasta o componente será criado?

Selecione uma alternativa

A

`src/app/user-list`

B

`src/app/users/user-list`

C

`src/users/user-list`

D

`src/app/users`

Questão 10)

Por quais motivos criamos um componente próprio de listagem de imagens?

Selecione 2 alternativas

A

Para separar melhor a responsabilidade (e assim facilidade de manutenção)

B

Melhor testabilidade

C

Melhorar o desempenho

D

Melhorar o carregamento assíncrono das photos

E

Escrever menos código e assim menos bugs

Questão 11)

Todas as afirmações sobre `BrowserModule` e `CommonModule` são verdadeiras exceto:

Selecione uma alternativa

- A** Ao usar o `BrowserModule` não temos acesso ao `CommonModule`.
- B** O `CommonModule` possui todas as diretivas básicas como `NgIf`, `NgFor`, `NgForOf` etc
- C** O `BrowserModule` possui funcionalidades essenciais para rodar e iniciar a aplicação.
- D** O `BrowserModule` só deve ser importado no modulo principal da aplicação.

Questão 12)

O que acontece na nossa aplicação ao acessarmos uma URL como `/user/flavio` ?

Selecione uma alternativa

- A** É chamado automaticamente o back end pelo navegador.
- B** O framework Angular interpreta essa URL e automaticamente a delega para o back end.
- C** O framework Angular interpreta essa URL e verifica se há um roteamento associado.
- D** É chamado o módulo pelo navegador.

Questão 13)

Você encontrou a seguinte declaração de rotas no projeto:

```
const appRoutes: Routes = [  
  { path: 'produtos', component: ProdutosListComponent },  
  { path: 'p/:id', component: ProdutoDetalheComponent },  
  { path: 'admin', component: AdminComponent },  
  { path: '**', component: PageNotFoundComponent }  
]
```

Ao acessar a rota `/produtos/34` qual componente será chamado?

Selecione uma alternativa

A

`ProdutosListComponent`

B

`ProdutoDetalheComponent`

C

`PageNotFoundComponent`

D

Nenhuma.

Questão 14)

Thereza criou um componente container, aquele que pode ter elementos filhos. Todavia, ao ser renderizado, nenhum dos elementos filhos foi exibido, apenas o título do elemento pai.

Marque a opção que mais se aproxima da causa do problema enfrentado por Thereza.

Selecione uma alternativa

A

O componente pai não pertence a um módulo.

B

O `templateUrl` do componente pai não existe.

C

No template do componente não foi utilizado `<ng-content>`.

Questão 15)

Marque todas as alternativas verdadeiras à respeito de uma output property:

Selecione 3 alternativas

A

O nome da output property é o mesmo nome do evento utilizado por aqueles que desejam interagir com o componente.

B

São propriedades decoradas com o decorator `Output`.

C

Não basta aplicar um decorator específico, é necessário que a propriedade seja uma instância de `EventEmitter`.

D

Dizemos que através de uma output property estamos realizando um data binding.

Questão 16)

Marque as opções verdadeiras sobre a criação de diretivas:

Selecione 3 alternativas

A

Todo componente é uma diretiva que possui template. No entanto, uma diretiva em seu estado bruto não possui templates.

B

Podemos usar uma diretiva como atributo envolvendo o valor do seu `seletor` entre colchetes.

C

Podemos injetar no constructor da diretiva uma referência para o elemento no qual ela foi associada. Angular nos dá acesso ao elemento através do wrapper `ElementRef`.

D

A diretiva, através do decorator `HostBinding`, pode ouvir os eventos disparados no elemento host, isto é, aquele no qual a diretiva foi associada.

Questão 17)

Qual módulo Ana deve importar em seu código para que possa ter acesso ao `FormBuilder`, a instância utilizada para criar validações diretamente no componente?

Selecione uma alternativa

A

`ReactiveInputModule`

B

`ValidatorFormsModule`

C

`ReactiveFormsModule`

Questão 18)

Luiz Antônio descobriu que possui dois meios para redirecionar um usuário para sua página de perfil depois de realizar o login, quais alternativas abaixo realizam o redirecionamento conforme Luiz Antônio quer?

Selecione 2 alternativas

A

```
this.router.navigateByUrl('user/' + userName)
```

B

```
this.router.redirectTo('user/' + userName)
```

C

```
this.router.navigate(['user', userName])
```

D

```
this.router.redirectTo(['user', userName])
```

Questão 19)

Aprendemos que podemos dar o `focus` automaticamente para um campo em caso de erro de validação. Qual o processo que devemos seguir para que isso seja feito?

Selecione uma alternativa

A

Não há mistério, podemos simplesmente usar o `focus` do HTML5 que o problema será resolvido.

B

Primeiramente precisamos injetar uma variável de template referente ao campo desejado. Esta variável ao ser injetada será do tipo `ElementRef`. Após isso teremos acesso ao método `focus()`.

C

Infelizmente não há um jeito de se fazer isso, mesmo utilizando o Angular.

D

Primeiramente precisamos injetar uma variável de template referente ao campo desejado. Esta variável ao ser injetada será do tipo `TemplateField`. Após isso devemos fazer um *casting* do tipo `TemplateField` para `ElementRef`, e, por fim, teremos acesso ao método `focus()`.

Questão 20)

Análise o código abaixo:

```
return this.http
  .post(
    API_URL + '/user/login',
    { userName, password }
  )
  .pipe(tap(res => {
    const authToken = res.headers.get('x-access-token')
    console.log(`User ${userName} authenticated with token : ${authToken}`);
  }));
```

Ele compila?

Selecione uma alternativa

A

Não, pois para acessar o `headers` na resposta, é preciso expô-lo na hora que fizermos o `post`, para que possamos manipulá-lo.

B

Sim, o código compila.

C

Não, pois para acessar o `headers` na resposta, é preciso expô-lo na hora que fizermos o `pipe`, para que possamos manipulá-lo.

D

Não, pois para acessar o `headers` na resposta, é preciso expô-lo na hora que fizermos o `tap`, para que possamos manipulá-lo.

Questão 21)

Sobre o token, julgue as afirmativas abaixo:

- 1) O token é gerado no padrão **JWT** (*J*son *Web Token*)
- 2) Um dos algoritmos de criptografia usado em sua assinatura é o **HMAC SHA256** (**HS256**)
- 3) O token pode ser decodificado
- 4) Ao ser decodificado, o token pode ser alterado, codificado novamente e enviado para o back-end, gerando assim um acesso indevido

Assinale a alternativa correta:

Selecione uma alternativa

A	A afirmativa 3 é falsa
B	A afirmativa 2 é falsa
C	A afirmativa 1 é falsa
D	A afirmativa 4 é falsa

Questão 22)

Renata estava estudando Angular e não conseguiu entender muito bem qual a função do *Async pipe*.

Qual das explicações abaixo deve ser utilizada para explicá-la?

Selecione uma alternativa

A

Com o *Async pipe* conseguimos capturar a emissão do `Observable` diretamente do nosso template.

B

O *Async pipe* serve para termos um *placeholder* para as informações contidas no nosso template.

C

O *Async pipe* nada mais é do que uma forma de concatenação de informações.

D

Mesmo com o *Async pipe*, precisaremos da propriedade `user` para acessar no template, já que o *pipe* nos ajuda apenas em performance.

Questão 23)

Rafael começou um projeto próprio e reparou que seu usuário, depois de logado, consegue acessar rotas indevidas! Por que isto acontece?

Selecione uma alternativa

A

O guarda de rotas serve para bloquearmos todas as rotas para nosso usuário, garantindo que ele só tenha acesso àquela em que ele está no momento.

B

O guarda de rotas serve para darmos consistência para nossa aplicação, liberando acesso apenas para as rotas que fazem sentido para nosso usuário.

C

Não existe forma de bloquear tais acessos, temos que apenas esconder os links que levam para as rotas indevidas.

D

O guarda de rotas nos garante segurança contra ataques e deixa nossa aplicação segura.

Questão 24)

A classe `FormGroup` é importantíssima no Angular e nos disponibiliza uma série de funcionalidades!

Dentre as opções abaixo, marque apenas a que **não representa** uma funcionalidade possível de ser executada com essa classe!

Dica: Caso seja necessário, utilize a [documentação](#) dessa classe!

Selecione uma alternativa

A

Setar os valores do formulário através de um objeto.

B

Verificar se o formulário tem um determinado campo através do respectivo `controlName`.

C

Obter um objeto com os valores do formulário.

D

Remover todo o formulário do HTML.

Você encontrou a seguinte declaração de rotas no projeto:

```
const appRoutes: Routes = [  
  { path: 'produtos',  
    component: ProdutosHomeComponent,  
    children: [  
      {  
        path: '',  
        component: ProdutoListComponent  
      },  
      {  
        path: ' :id',  
        component: ProdutoDetalheComponent  
      }  
    ]  
  }  
]
```

Quais componentes são executados ao acessar `/produtos/4` ?

Selecione uma alternativa

A

`ProdutosHomeComponent` e `ProdutoDetalheComponent`

B

`ProdutosHomeComponent` e `ProdutoListComponent`

C

Apenas `ProdutoDetalheComponent`

D

Apenas `ProdutoListComponent`

Questão 26)

Marque as alternativas verdadeiras sobre rotas:

Selecione 2 alternativas

A

Para sabermos se um evento é do tipo `NavigationStart` fazemos `event.from(NavigationEnd)`.

B

Uma instância de `Router` possui a propriedade `events`, um Observable que nos permite saber a fase atual da rota acessada pela aplicação.

C

Um evento do tipo `NavigationEnd` é aquele disparado quando a rota termina com sucesso.

Questão 27)

Para obtermos informações sobre o progresso de upload ou download de arquivo precisamos passar um objeto especial para os métodos de `HttpClient` desta maneira:

Selecione uma alternativa

A

```
{
  reportProgress: true
}
```

B

```
{
  observe: 'events'
}
```

C

```
{
  observe: 'events',
  reportProgress: true
}
```

Temos o seguinte trecho de código:

```
this.photoService
  .upload(description, allowComments, this.file)
  .pipe(finally(() => {
    this.router.navigate(['/user', this.userService.getUser()]);
  }));
.subscribe(
  (event: HttpEvent<any>) => {
    if(event.type == EventType.UploadProgress) {
      this.percentDone = Math.round(100 * event.loaded / event.total);
    } else if(event instanceof HttpResponse) {
      this.alertService.success('Upload complete', true);
    }
  },
  err => {
    console.log(err);
    this.alertService.danger('Upload error!', true);
  });
```

Marque as alternativas verdadeiras sobre ele:

Selecione 2 alternativas

A

O usuário será direcionado a mesma rota tanto no sucesso quanto no fracasso da operação.

B

O usuário será direcionado para uma rota específica apenas quando a operação for realizada com sucesso.

C

Foi necessário o emprego de condicionais no callback passado para `subscribe` pois ele será chamado diversas vezes trazendo o percentual de upload já realizado no evento `HttpEvent.UploadProgress` e quando terminar através de um evento do tipo `Response`.

Questão 29)

Marque as alternativas verdadeiras a respeito do uso de interceptadores:

Selecione 2 alternativas

A

São classes que implementam a interface `HttpInterceptor`.

B

Não basta criarmos um interceptador, ele precisa ser registrado na aplicação para que esteja ativo.

C

Não podemos injetar artefatos em interceptadores.

Questão 30)

Vejamos o seguinte trecho de template:

```
<button tabindex="0" (keyup)="acao()" (click)="acao()"></button>
```

A função `acao()` será chamada quando o usuário clicar no botão ou pressionar qualquer tecla. Marque a opção na qual o evento `keyup` seja ativado apenas quando o usuário teclar espaço:

A) `<button tabindex="0" (keyup.0)="acao()" (click)="acao()"></button>`

B) `<button tabindex="0" (keyup.space)="acao()" (click)="acao()"></button>`

C) `<button tabindex="0" (keyup-space)="acao()" (click)="acao()"></button>`

Questão 31)

Para que um elemento de um template ganhe foco através do uso da tecla TAB ele precisa:

Selecione uma alternativa

A

Apenas teclar SHIFT + TAB.

B

Definir a propriedade `tabindex`, por exemplo, com o valor `tabindex = 0`.

C

Apenas teclar TAB.

Questão 32)

Temos o seguinte código de template

```
<div [ngClass]="{ 'x-y': xy }"></div>
```

Marque a alternativa verdadeira sobre ele:

Selecione uma alternativa

A

A classe `x-y` será aplicada no template apenas se a propriedade `xy` do componente no qual o template faz parte for `true`.

B

A classe `x-y` será aplicada no template apenas se a propriedade `xy` do componente no qual o template faz parte for a string `xy`.

C

A classe `xy` será aplicada no template apenas se a propriedade `x-y` do componente no qual o template faz parte for `true`.

Questão 33)

Podemos consultar se o formulário já foi submetido ou não com auxílio de uma variável de template que guarda como valor uma referência para:

Selecione uma alternativa

A

`ReactiveFormsModule`

B

`formGroup`

C

`ngForm`

No paradigma da orientação a objetos criamos a representação de algo do mundo real em nosso programa através de modelos e esses modelos são definidos através de classes.

Marque a alternativa que cria corretamente uma classe usando o ECMASCRIPT 2015:

Selecione uma alternativa

A

```
class Pessoa {
    constructor(nome, idade) {
        this._nome = nome;
        this._idade = idade;
    }
}
```

B

```
class Pessoa {
    Pessoa(nome, idade) {
        this._nome = nome;
        this._idade = idade;
    }
}
```

C

```
class Pessoa {
    nome;
    idade;
}
```

Questão 35)

Temos as seguintes afirmações a respeito do compilador do TypeScript:

- a) O uso de Node.js é opcional, mas altamente recomendável.
- b) Ele traduz um código escrito em JavaScript para um código compatível com o TypeScript, necessário para que o navegador o compreenda.
- c) Podemos passar configurações especiais para o compilador através do arquivo tsconfig.json
- d) É instalado através do npm

Sobre as afirmativas anteriores, podemos dizer que:

Selecione uma alternativa

A

D e C são falsas

B

C e A são verdadeiras

C

A e B são falsas.

Marque apenas as opções verdadeiras sobre o tipo **any**.

Selecione 2 alternativas

A

Uma variável do tipo `any` só pode receber dados do tipo `any` .

B

Por padrão, é assumido automaticamente pelo TypeScript quando não definimos o tipo das nossas variáveis.

C

É possível desativar o tipo implícito `any` passando uma configuração especial para o compilador no arquivo `tsconfig.json` . Isso fará com que o compilador emita um erro em todos os locais que o tipo `any` foi adotado implicitamente.

D

Favorece o compilador, inclusive IDE's a realizarem o autocomplete e a inferirem todos os métodos da variável.

Questão 37)

Marque apenas as opções que declaram um array corretamente em TypeScript, assumindo que a configuração `noImplicitAny` está definida com o valor `true` :

Selecione 2 alternativas

A

```
let nomes: string[] = [];
```

B

```
let idades: Array<Number> = [1, 2, 3];
```

C

```
let salarios: Array = [5400.00, 2400.00, 7100.00];
```

Questão 38)

Utilizando a API do DOM, podemos criar elementos dinamicamente através de `document.createElement` ou:

Selecione uma alternativa

A

Através da propriedade `innerHTML` que aceita receber elementos do DOM.

B

Através da propriedade `innerHTML` que recebe uma string que é convertida para elementos do DOM.

C

`textContent`, que aceita receber uma string que é convertida em elementos do DOM.

Questão 39)

Mônica decidiu criar um jogo em JavaScript, mas optou por utilizar TypeScript devido aos recursos extras da linguagem. Ela criou três classes:

- Humanoide
- Humano
- Alienigena

Em termos de design, tanto Humano quanto Alienigena são humanóides, por isso herdam dessa classe:

```
class Humanoide {  
  
    private _energia: number = 100;  
    private _nome: string = '';  
  
    get energia() {  
  
        return this._energia;  
    }  
  
    get nome() {  
  
        return this._nome;  
    }  
  
    set nome(nome) {  
  
        this._nome = nome;  
    }  
}  
  
class Humano extends Humanoide {  
  
    private _idade: number = 0;  
  
    get idade() {  
  
        return this._idade;  
    }  
  
    set idade(idade) {  
  
        this._idade = idade;  
    }  
}  
  
class Alienigena extends Humanoide {  
  
    private _energiaExtra: number = 100;  
  
    get energia() {  
  
        return this._energia + this._energiaExtra;  
    }  
}
```

Marque a alternativa verdadeira:

Selecione uma alternativa

- A) A classe `Humanoide` não compila.
- B) A classe `Alienígena` não compila.
- C) A classe `Humano` não compila.

Questão 40)

Elizabete usa a biblioteca `underscore` (<http://underscorejs.org/>) durante muitos anos. Contudo, ao utilizar TypeScript, recebe um erro de compilação todas as vezes que o `_`, o alias do `underscore` é utilizado.

```
let numeros = [1, 2, 3];  
_.map(numeros, num => num * 3);
```

Marque a única afirmativa verdadeira que explica o motivo do erro de compilação do código.

Selecione uma alternativa

A

O `_` é uma variável que vive no escopo global. O compilador TypeScript não sabe disso e a considera como não declarada, por isso o erro de compilação. Faz sentido, porque a ideia do TypeScript é nos blindar de possíveis erros em nosso código e utilizar variáveis que não foram declaradas é uma deles.

B

O código não compila porque não podemos usar `_` como nome de variáveis.

C

A mensagem de erro recebida é apenas um warning e não impedirá que o código seja compilado. Inclusive, `_` é uma variável no escopo global e o TypeScript automaticamente já detecta essa questão.

Questão 41)

Vimos que podemos criar novos objetos através de um spread, e que caso alteremos o valor desse novo objeto, ele não será alterado no objeto original. Essa afirmação é verdadeira? Por que?

Selecione uma alternativa

A

Sim, ele mantém o mesmo nome e a mesma referência, porém ao alterar, ele cria um novo espaço na memória com o novo valor.

B

Não, pois ele cria uma referência ao objeto original, apontando para o mesmo espaço da memória no qual o valor original se encontra.

C

Não, porém ele cria um novo espaço na memória referenciando o objeto original, sendo assim, dois objetos iguais alocados em espaços na memória diferente.

D

Sim, pois a criação do novo objeto não é feita por referência, ou seja, não aponta para o mesmo espaço de memória;

Questão 42)

Vimos que a criação de uma *classe* pode ser feita através da palavra-chave `class` e com isso definimos quais atributos queremos que essa classe possua.

Sabendo que classes são apenas um dos recursos da linguagem Javascript, quando é que devemos criar ou não uma classe ?

Selecione uma alternativa

A

Devemos evitar o uso de classes, uma vez que elas tornam nosso programa mais lento e difícil de ser compreendido.

B

Classes definem uma forma de organizarmos uma série de informações repetidas no nosso código e por isso devemos usar elas quando temos código que se repete e que faz parte de um contexto coeso.

C

Só podemos usar classes onde tivermos código repetido. Não devemos usar elas em mais nenhum lugar.

Questão 43)

Criamos métodos para proteger atributos e informações sensíveis de nossas classes. Porém se não fizermos nada essas informações ainda estão expostas e podem ser alteradas manualmente.

Sobre a proteção de atributos, marque as alternativas corretas:

Selecione 2 alternativas

A

Não precisamos nos preocupar com proteção de atributos se tivermos os métodos necessários para alterar aquele atributo.

B

Por padrão no JS utilizamos o "_" para indicar que um atributo é privado e não deveria ser alterado.

C

Atualmente no Js nenhum atributo ou método é realmente privado

Questão 44)

A organização de um projeto de programação é algo muito importante para que conforme o sistema crescer encontrarmos mais facilmente as classes e lugares que queremos alterar.

Para que serve a criação de módulos no JavaScript?

Selecione uma alternativa

A

Só devemos usar módulos quando temos código muito grande e que não conseguimos fazer caber em uma única tela.

B

Módulos servem apenas para conseguirmos exportar código de bibliotecas externas ao nosso projeto.

C

Criamos módulos para compartilhar código entre os diferentes arquivos do meu sistema, ajudando na organização dele.

Questão 45)

Sobre os assessores do tipo `get` e `set` marque as alternativas corretas:

Selecione 2 alternativas

A

Usando assessores do tipo `set` podemos alterar a regra de como um atributo pode ou não ser modificado sem precisar alterar isso em diversos pontos do código

B

Podemos usar assessores do tipo `get` para atribuir novos valores para um atributo.

C

Usar assessores do tipo `set` é uma boa prática para garantirmos que a atribuição de propriedades está sempre segura

Questão 46)

Abaixo, temos algumas afirmações a respeito da utilização de construtores.

Qual delas é verdadeira?

Selecione uma alternativa

A

Construtores são utilizados para inicializar os atributos.

B

Construtores não têm utilidade real, podemos deixar os atributos públicos e defini-los manualmente.

C

Construtores não podem receber parâmetros.

Questão 47)

Qual a sintaxe do JavaScript para herdarmos de uma classe?

Selecione uma alternativa

A

```
class Carro extends Veiculo
```

B

```
class Carro(Veiculo)
```

C

```
class Carro : Veiculo
```

D

```
class Carro < veiculo
```

Aprendemos nessa aula sobre o conceito de Métodos e Classes abstratas.

Selecione as afirmativas corretas sobre esses temas:

Selecione 2 alternativas

A

Classes abstratas são classes que possuem classes filhas, mas ainda podem ser instanciadas diretamente.

B

Classes abstratas são pensadas de forma que elas apenas possam ser herdadas por outras.

C

Métodos abstratos devem ser sobrescritos pelas classes filhas.

D

Um método abstrato é qualquer método que lance um erro quando for chamado.

E

Nosso programa pode ser composto apenas de classes abstrata.