ATIVIDADE PARA AVALIAÇÃO - SEMANA 3

A resposta correta da questão está identificada com a cor Vermelha.

1. **(1 ponto)**Sobre a linguagem C++, é correto afirmar que:
   1. É uma linguagem interpretada, então não existe a geração de um executável nativo para uma determinada plataforma.
   2. **Possui tipagem estática, então as variáveis não podem mudar de tipo após serem declaradas.**
   3. Programas escritos em C++ são mais lentos que programas escritos em Python, por esta última ser uma linguagem compilada.
   4. Não fornece suporte ao paradigma orientado a objetos.
   5. O escopo é delimitado pela indentação do código.

A resposta correta é: “Possui tipagem estática, então as variáveis não podem mudar de tipo após serem declaradas.”  
  
JUSTIFICATIVA  
  
É uma linguagem interpretada, então não existe a geração de um executável nativo para uma determinada plataforma.  
Incorreto, C++ é uma linguagem compilada.  
Correto.  
Programas escritos em C são mais lentos que programas escritos em Python, por esta última ser uma linguagem compilada.  
Incorreto, programas escritos em C++ são mais rápidos porque python é uma linguagem interpretada.  
Não fornece suporte ao paradigma orientado a objetos.  
Incorreto, o suporte à Programação Orientada a Objetos é fornecido.  
O escopo é delimitado pela indentação do código.  
Incorreto, o escopo é delimitado por chaves.

1. **(1 ponto)**Sobre os recursos e as regras de sintaxe da linguagem C++, assinale a opção correta.
   1. A sintaxe da linguagem C++ é semelhante à sintaxe da linguagem Python, ambas usam o operador ":" para delimitar o início do escopo.
   2. Tanto C++ quanto Python permitem manipulação explícita de ponteiros, o que torna ambas as linguagens igualmente capazes de acessar a memória principal do computador.
   3. Na passagem de parâmetro por valor, uma região de memória é passada para a função. Assim, qualquer mudança no valor dentro da função altera o valor da variável fora da função.
   4. A escolha da passagem de parâmetro por valor ou por referência importa por questões de desempenho, mas não altera o comportamento dos programas. Em outras palavras, se alguém trocar todas as passagens de parâmetro por valor para passagem de parâmetros por referência (e vice-versa), o programa executará da mesma maneira.
   5. **Para efetuar uma passagem de parâmetros por referência, colocamos o operador & antes do nome da variável.**

A resposta correta é: “Para efetuar uma passagem de parâmetros por referência, colocamos o operador & antes do nome da variável.”  
  
JUSTIFICATIVA  
  
A sintaxe linguagem C++ é semelhante à sintaxe da linguagem Python, ambas usam o operador ":" para delimitar o início do escopo.  
Incorreto, o operador ":" não inicia escopo em C++.  
Tanto C++ quanto Python permitem manipulação explícita de ponteiros, o que torna ambas as linguagens igualmente capazes de acessar a memória principal do computador.  
Incorreto, não há manipulação explícita de ponteiros em Python.  
Na passagem de parâmetro por valor, uma região de memória é passada para a função. Assim, qualquer mudança no valor dentro da função altera o valor da variável fora da função.  
Incorreto, o que foi mencionado na alternativa é a passagem de parâmetro por referência.  
A escolha de passagem de parâmetro por valor ou por referência importa por questões de desempenho, mas não altera o comportamento dos programas. Em outras palavras, se alguém trocar todas as passagens de parâmetro por valor para passagem de parâmetros por referência (e vice-versa), o programa executará da mesma maneira. Incorreto, trocar as passagens de parâmetro por valor para passagens por referência (e vice-versa) causará mudanças no comportamento dos programas.  
Para efetuar uma passagem de parâmetros por referência, colocamos o operador & antes do nome da variável. Correto.

1. **(1 ponto)** Sobre o trecho de código mostrado a seguir:

Assinale a alternativa verdadeira:

* 1. Ocorrerá um erro em tempo de compilação, porque intPointer não pode receber o valor retornado por &alpha.
  2. Não ocorrerá erro em tempo de compilação, mas o programa gerará erro em tempo de execução, porque intPoint não pode receber o valor retornado por &alpha.
  3. **A variável intPointer aponta para alpha. Nesse caso, alterações podem ser feitas diretamente usando alpha ou indiretamente com intPointer.**
  4. A variável alpha terá um valor diferente daquele conteúdo que está sendo apontado por intPointer na memória.
  5. A variável alpha é um ponteiro para uma região de memória, podendo acessar essa memória indiretamente.

A resposta correta é: “A variável intPointer aponta para alpha. Nesse caso, alterações podem ser feitas diretamente usando alpha ou indiretamente com intPointer.”  
  
JUSTIFICATIVA  
  
Ocorrerá um erro em tempo de compilação, porque intPointer não pode receber o valor retornado por &alpha.  
Incorreto, não será gerado erro em tempo de compilação neste código.  
Não ocorrerá erro em tempo de compilação, mas o programa gerará erro em tempo de execução, porque intPoint não pode receber o valor retornado por &alpha.  
Incorreto, não será gerado erro em tempo de execução.  
A variável alpha terá um valor diferente daquele que está sendo apontado por intPointer na memória.  
Incorreto, o valor de alpha será igual ao valor de \*intPointer.  
A variável intPointer aponta para alpha. Nesse caso, alterações podem ser feitas diretamente usando alpha ou indiretamente com intPointer.  
Correto.  
A variável alpha é um ponteiro para uma região de memória, podendo acessar essa memória indiretamente.  
Incorreto, alpha é uma variável convencional, não um ponteiro.

1. **(1 ponto)**Sobre alocação dinâmica, é correto afirmar que:
   1. **Alocação dinâmica permite alocar memória sem a necessidade de definir em tempo de compilação o tamanho a ser utilizado, dado que a alocação ocorre em tempo de execução.**
   2. O operador "new" para fazer alocação dinâmica em C++ permite gerar alocação de objetos, mas não permite fazer alocação de tipos pré-definidos, como "int" e "float".
   3. O operador "delete" permite desalocar qualquer espaço de memória, inclusive as regiões de memória alocadas estaticamente em tempo de compilação.
   4. Em C++, não há a necessidade de desalocar a memória com "delete", dado que a máquina virtual efetua essa tarefa periodicamente. A prática de desalocar memória é aconselhável, mas não essencial.
   5. Ao alocar espaço em uma região de memória com "new", temos a opção de acessar essa região com um ponteiro ou com uma variável convencional. Neste último caso, basta atribuir o retorno obtido com o comando "new" para uma variável que não é ponteiro.

A resposta correta é: “Alocação dinâmica permite alocar memória sem a necessidade de definir em tempo de compilação o tamanho a ser utilizado, dado que a alocação ocorre em tempo de execução.”  
  
JUSTIFICATIVA  
  
Alocação dinâmica permite alocar memória sem a necessidade de definir em tempo de compilação o tamanho a ser utilizado, dado que a alocação ocorre em tempo de execução.  
Correto.  
O operador "new" para fazer alocação dinâmica em C++ permite gerar alocação de objetos, mas não permite fazer alocação de tipos pré-definidos, como "int" e "float".  
Incorreto, a alocação de tipos pré-definidos é permitida.  
O operador "delete" permite desalocar qualquer espaço de memória, inclusive as regiões de memória alocadas estaticamente em tempo de compilação.  
Incorreto, é possível somente desalocar espaço de memória que foi alocado dinamicamente.  
Em C++, não há a necessidade de desalocar a memória com "delete", dado que a máquina virtual efetua essa tarefa periodicamente. A prática de desalocar memória é aconselhável, mas não essencial.  
Incorreto, não temos máquina virtual em C++ que faça algum tipo de coletor de lixo.  
Ao alocar espaço em uma região de memória com "new", temos a opção de acessar essa região com um ponteiro ou com uma variável convencional. Neste último caso, basta atribuir o retorno obtido com o comando "new" para uma variável que não é ponteiro.  
Incorreto, não é possível atribuir o retorno de "new" para uma variável que não é ponteiro.

1. **(1 ponto)**Sobre alocação de vetores em C++, é correto afirmar que:
   1. Em C++, um vetor alocado estaticamente ocupa regiões contíguas de memória, enquanto que um vetor alocado dinamicamente fica espalhado na memória de maneira não contígua.
   2. **Todos os elementos alocados em um vetor devem ser do mesmo tipo, e o tamanho do vetor deve ser informado antes de fazer a utilização do mesmo, não podendo ser aumentado ou diminuído posteriormente.**
   3. Em C++, é possível inicializar elementos no momento da declaração de um vetor estático usando uma lista inicializadora (lista de elementos entre chaves). A quantidade de elementos na lista não pode ser menor do que o tamanho declarado para o vetor, pois isso geraria erro em tempo de compilação.
   4. Em C++, é possível acessar um elemento usando a sintaxe de colchetes se quisermos acessar um único elemento, ou a sintaxe de chaves para acessar vários elementos ao mesmo tempo.
   5. Em C++, se o código tentar acessar uma posição que está fora dos limites do vetor, receberemos um erro em tempo de compilação dado que a informação do tamanho do vetor é conhecida antes de executar o programa.

A resposta correta é: “Todos os elementos alocados em um vetor devem ser do mesmo tipo, e o tamanho do vetor deve ser informado antes de fazer a utilização do mesmo, não podendo ser aumentado ou diminuído posteriormente."  
  
JUSTIFICATIVA  
  
Coloque aqui um comentário fundamentado ou a resolução passo a passo.  
Todos os elementos alocados em um vetor devem ser do mesmo tipo, e o tamanho do vetor deve ser informado antes de fazer a utilização do mesmo, não podendo ser aumentado ou diminuído posteriormente.  
Correto.  
Em C++, um vetor alocado estaticamente ocupa regiões contíguas de memória, enquanto que um vetor alocado dinamicamente fica espalhado na memória de maneira não contígua.  
Incorreto, um vetor alocado dinamicamente também ocupa posições contíguas de memória.  
Em C++, é possível inicializar elementos no momento da declaração de um vetor estático usando uma lista inicializadora (lista de elementos entre chaves). A quantidade de elementos na lista não pode ser menor do que o tamanho declarado para o vetor, pois isso geraria erro em tempo de compilação.  
Incorreto, é possível colocar menos elementos na lista.  
Em C++, é possível acessar um elemento usando a sintaxe de colchetes se quisermos acessar um único elemento, ou a sintaxe de chaves para acessar vários elementos ao mesmo tempo.  
Incorreto, não existe a referida sintaxe de chaves para acessar vários elementos ao mesmo tempo.  
Em C++, se o código tentar acessar uma posição que está fora dos limites do vetor, receberemos um erro em tempo de compilação dado que a informação do tamanho do vetor é conhecida antes de executar o programa.  
Incorreto, é função do programador gerenciar os limites do vetor, não havendo erro em tempo de compilação se o limite for violado.

1. **(1 ponto)** Sobre a estrutura de dados pilha com implementação com um vetor, é possível afirmar que:
   1. Na implementação de uma pilha, precisamos sempre de duas variáveis distintas, uma para apontar para o local onde ocorrem as inclusões e outra para apontar para o local onde ocorrem as exclusões.
   2. **O comportamento de uma pilha é semelhante ao comportamento do botão "desfazer" de editores de texto. Ao acionar o desfazer, o último comando inserido é o primeiro a ser desfeito.**
   3. Para implementar uma pilha usando vetor, é obrigatório que a alocação do vetor seja dinâmica, pois precisamos fazer com que o vetor aumente e diminua de tamanho em tempo de execução.
   4. Nas implementações de pilhas usando vetor, as inserções ocorrem em tempo constante, enquanto que as remoções dependem do tamanho do vetor.
   5. Nas implementações de pilha usando vetor, a variável que indica o topo da pilha precisa ser uma variável do tipo ponteiro para inteiro, dado que um vetor é, na verdade, um ponteiro para uma região de memória.

A resposta correta é: “O comportamento de uma pilha é semelhante ao comportamento do botão "desfazer" de editores de texto. Ao acionar o desfazer, o último comando inserido é o primeiro a ser desfeito."  
  
JUSTIFICATIVA  
  
Em “Incremental, Cascata, Recursivo e Imperativo”: Recursivo não é um tipo de ciclo de vida, é um método de implementação.  
Em “Funcional, Imperativo, Interativo e Orientado a Objetos.”: são paradigmas de desenvolvimento de Na implementação de uma pilha, precisamos sempre de duas variáveis distintas, uma para apontar para o local onde ocorrem as inclusões e outra para apontar para o local onde ocorrem as exclusões.  
Incorreto, as inclusões e exclusões ocorrem todas no topo da pilha, então basta uma variável para fazer a indicação de onde ocorrem.  
Para implementar uma pilha usando vetor, é obrigatório que a alocação do vetor seja dinâmica, pois precisamos fazer com que o vetor aumente e diminua de tamanho em tempo de execução.  
Incorreto, não é obrigatório que a alocação seja dinâmica. Além disso, mesmo com alocação dinâmica não podemos aumentar e diminuir o tamanho do vetor.  
O comportamento de uma pilha é semelhante ao comportamento do botão "desfazer" de editores de texto. Ao acionar o desfazer, o último comando inserido é o primeiro a ser desfeito.  
Correto  
Nas implementações de pilhas usando vetor, as inserções ocorrem em tempo constante, enquanto que as remoções dependem do tamanho do vetor.  
Incorreto, tanto as inserções quanto as remoções devem ocorrer em tempo constante.  
Nas implementações de pilha usando vetor, a variável que indica o topo da pilha precisa ser uma variável do tipo ponteiro para inteiro, dado que um vetor é, na verdade, um ponteiro para uma região de memória.  
Incorreto, a variável precisa ser do tipo inteiro, dado que o apontamento é feito usando o índice do vetor.

1. **(1 ponto)**Sobre a estrutura de dados fila com implementação com um vetor, é possível afirmar que:
   1. Uma fila é uma estrutura de dados tipicamente usada para resolver problemas de alinhamento de estruturas. Por exemplo, em linguagens de programação, garantir que a abertura dos escopos está sendo adequadamente finalizada.
   2. **Filas são estruturas que garantem acesso justo a recursos compartilhados. Isso ocorre porque o primeiro elemento a entrar na fila (o que pode ser interpretado como pedir acesso a um recurso) será o primeiro a sair (ser atendido).**
   3. Uma fila é uma estrutura dita não linear, dado que não é facilmente interpretada como um vetor. Por conta disso, implementações de fila usando vetores não são adequadas.
   4. Na implementação de uma fila usando um vetor, utilizamos uma variável para guardar a posição da cabeça da fila, sendo isso suficiente para implementar as operações de inserção e remoção, dado que ambas as operações ocorrem na cabeça da fila.
   5. Um grande problema da implementação de filas com vetores é que inserções ocupam espaço no final do vetor e remoções liberam espaço no início do vetor, o que impede que o espaço liberado seja reaproveitado. Assim, a fila pode ficar cheia mesmo tendo muito espaço livre no início do vetor, um desperdício comum às implementações de fila com vetor.

A resposta falsa é: “Filas são estruturas que garantem acesso justo a recursos compartilhados. Isso ocorre porque o primeiro elemento a entrar na fila (o que pode ser interpretado como pedir acesso a um recurso) será o primeiro a sair (ser atendido)."  
  
JUSTIFICATIVA  
  
Uma fila é uma estrutura de dados tipicamente usada para resolver problemas de alinhamento de estruturas. Por exemplo, em linguagens de programação, garantir que a abertura dos escopos está sendo adequadamente finalizada.  
Incorreto, a alternativa confunde fila com pilha.  
Uma fila é uma estrutura dita não linear, dado que não é facilmente interpretada como um vetor. Por conta disso, implementações de fila usando vetores não são adequadas.  
Incorreto, uma fila é uma estrutura dita linear.  
Filas são estruturas que garantem acesso justo a recursos compartilhados. Isso ocorre porque o primeiro elemento a entrar na fila (o que pode ser interpretado como pedir acesso a um recurso) será o primeiro a sair (ser atendido).  
Correto  
Na implementação de uma fila usando um vetor, utilizamos uma variável para guardar a posição da cabeça da fila, sendo isso suficiente para implementar as operações de inserção e remoção, dado que ambas as operações ocorrem na cabeça da fila.  
Incorreto, as operações de remoções e inserções ocorrem em pontos diferentes da fila.  
Um grande problema da implementação de filas com vetores é que inserções ocupam espaço no final do vetor e remoções liberam espaço no início do vetor, o que impede que o espaço liberado seja reaproveitado. Assim, a fila pode ficar cheia mesmo tendo muito espaço livre no início do vetor, um desperdício comum às implementações de fila com vetor.  
Incorreto. Este cenário da alternativa foi apresentado durante a aula. Entretanto, durante a própria aula foi criada uma solução para ele, que consiste em usar o resto da divisão pelo tamanho do vetor para encontrar a posição real do elemento no vetor. Fazendo, o vetor passa a ser imaginado como uma estrutura circular, ao invés de ser imaginado como uma estrutura linear.

1. **(1 ponto)**Sobre listas sequenciais e listas encadeadas, é correto afirmar que:
   1. Um vetor alocado estaticamente pode ser chamado de lista linear sequencial, pois a ordem lógica é igual à ordem física. O mesmo não ocorre com vetores alocados dinamicamente, pois estes podem ter seus elementos em lugares diversos na memória.
   2. **Para a busca binária funcionar com desempenho em tempo O(log n), precisamos da propriedade de que cada elemento possa ser acessado em tempo constante dado o índice. Nesse caso, é requerida uma lista linear sequencial, não sendo interessante usar lista encadeada.**
   3. Quando temos uma lista encadeada, o acesso a cada elemento, dado o índice, ocorre em tempo constante.
   4. Em uma lista encadeada, a ordem lógica dos elementos não corresponde ao posicionamento destes em memória. Por conta disso, é possível fazer acesso a qualquer elemento da lista em tempo constante.
   5. Em um vetor alocado dinamicamente, podemos aumentar ou diminuir o número de elementos. Nesse caso, é mais fácil alocar vetores com "new" do que estaticamente.

A resposta correta é: “Para a busca binária funcionar com desempenho em tempo O(log n), precisamos da propriedade de que cada elemento possa ser acessado em tempo constante dado o índice. Nesse caso, é requerida uma lista linear sequencial, não sendo interessante usar lista encadeada."  
  
JUSTIFICATIVA  
  
Um vetor alocado estaticamente pode ser chamado de lista linear sequencial, pois a ordem lógica é igual à ordem física. O mesmo não ocorre com vetores alocados dinamicamente, pois estes podem ter seus elementos em lugares diversos na memória.  
Incorreto, vetores alocados dinamicamente também são sequenciais.  
Quando temos uma lista encadeada, o acesso a cada elemento, dado o índice, ocorre em tempo constante.  
Incorreto, o acesso a cada elemento precisa seguir pelos elementos predecessores, não ocorrendo em tempo constante.  
Em uma lista encadeada, a ordem lógica dos elementos não corresponde ao posicionamento destes em memória. Por conta disso, é possível fazer acesso a qualquer elemento da lista em tempo constante.  
Incorreto, não é possível fazer acesso a qualquer elemento da lista em tempo constante.  
Para a busca binária funcionar com desempenho em tempo O(log n), precisamos da propriedade de que cada elemento possa ser acessado em tempo constante dado o índice. Nesse caso, é requerida uma lista linear sequencial, não sendo interessante usar lista encadeada.  
Correto.  
Em um vetor alocado dinamicamente, podemos aumentar ou diminuir o número de elementos. Nesse caso, é mais fácil alocar vetores com "new" do que estaticamente.  
Incorreto, não podemos aumentar ou diminuir o número de elementos em vetores alocados dinamicamente.

1. **(1 ponto)**Sobre as estruturas de dados pilha e fila, implementadas tanto com vetores quanto com listas encadeadas, é correto afirmar:
   1. Uma pilha implementada com vetores possui operações de push e pop em tempo constante. No caso da implementação com listas encadeadas, a operação de push possui tempo de execução determinado pelo número de elementos.
   2. **Tanto pilha ou fila implementada com vetores, quanto pilha ou fila implementada com listas encadeadas possuem operações de inserção e remoção de elementos em tempo constante.**
   3. Uma fila implementada com vetores possui operações de enqueue e dequeue em tempo constante. No caso da implementação com listas encadeadas, a operação de enqueue possui tempo de execução determinado pelo número de elementos.
   4. Uma pilha implementada com vetores possui operações de push e pop em tempo constante. No caso da implementação com listas encadeadas, a operação de pop possui tempo de execução determinado pelo número de elementos.
   5. Uma fila implementada com vetores possui operações de enqueue e dequeue em tempo constante. No caso da implementação com listas encadeadas, a operação de dequeue possui tempo de execução determinado pelo número de elementos.

A resposta correta é: “Tanto pilha ou fila implementada com vetores, quanto pilha ou fila implementada com listas encadeadas possuem operações de inserção e remoção de elementos em tempo constante."  
  
JUSTIFICATIVA  
  
Com a implementação de pilha em vetor, as operações de inserção e remoção simplesmente fazem operações locais no elemento marcado como topo. Isso envolve inserir ou remover um elemento do vetor e alterar o valor de uma variável do tipo inteiro, sendo que tudo pode ser feito independente do tamanho da pilha. Com a implementação em listas encadeadas, as operações de inserção e remoção envolvem alocar ou desalocar memória, respectivamente, seguida da atualização de um número constante de variáveis do tipo ponteiro.  
No caso de fila com a implementação em vetor, as operações de inserção e remoção envolvem inserir ou remover um único elemento que pode ser acessado em tempo constante com o índice, seguida da atualização dos valores de uma variável, que pode ser a que guarda o índice de quem está na frente, ou a que guarda o índice de quem está atrás, o que implica dizer que tudo ocorre independentemente do tamanho da fila. Com a implementação em listas encadeadas, as operações de inserção e remoção envolvem (do mesmo modo como ocorreu na pilha) alocar ou desalocar memória, respectivamente, seguida da atualização de um número constante de variáveis do tipo ponteiro.

1. **(1 ponto)**O código a seguir implementa um método em uma dada estrutura de dados, que pode ser uma pilha, implementada em aula na classe Stack, ou uma fila, implementada em aula na classe Queue. No código, a variável "structure" é um ponteiro para o início da estrutura de dados.

Sobre ele, é possível afirmar que:

* 1. O código implementa a função push da classe Stack, sendo que a estrutura interna é um vetor.
  2. **O código implementa a função push da classe Stack, sendo que a estrutura interna é uma lista encadeada.**
  3. O código implementa a função enqueue da classe Queue, sendo que a estrutura interna é um vetor.
  4. O código implementa a função enqueue da classe Queue, sendo que a estrutura interna é uma lista encadeada.
  5. O código implementa a função dequeue da classe Queue, sendo que a estrutura interna é uma lista encadeada.

A resposta correta é: “O código implementa a função push da classe Stack, sendo que a estrutura interna é uma lista encadeada."  
  
JUSTIFICATIVA  
  
O código possui alocação dinâmica no método de inserção de elementos, o que descarta o uso de um vetor como implementação interna. Além disso, o código está alocando o novo nó logo no início da estrutura com a linha "structure = location", algo que não seria feito se a estrutura de dados fosse. Lembrando que, se a estrutura de dados fosse uma fila, a inserção exigiria que um elemento que já está na estrutura apontasse para o elemento sendo inserido, algo que não vemos no trecho de código.