 INSTITUTO FEDERAL CEARÁ	CURSO ENGENHARIA DE COMPUTAÇÃO		DATA: 30 / 03 / 16
	Avaliação N1		2º semestre 1ª etapa
	DISCIPLINA: Estruturas de Dados.		Turno: Tarde
	Professor(a): Ernani Leite		Nota: 10,0
Aluno (a): JEFFERSON UCUOA PONTE			Matrícula:
Orientações Gerais: <ul style="list-style-type: none"> • Preencha legivelmente o cabeçalho e leia atentamente toda a avaliação antes de responder. Escreva com caneta azul ou preta, e utilize o verso ou folha adicional, caso necessário. • A prova deverá ser respondida individualmente e sem consulta, respeitadas as exceções previstas nas instruções específicas ou a critério do professor. Será atribuída nota zero ao aluno que utilizar meios ilícitos ou não autorizados pelo professor quando da realização de avaliações parciais. • O enunciado das questões contém todas as informações necessárias para respondê-las. A interpretação do enunciado faz parte da prova. Portanto, só em casos excepcionais poderão ser prestados esclarecimentos adicionais sobre as questões durante a realização da prova. • O aluno deverá entregar as folhas-rascunho juntamente com a prova. • O aluno que não comparecer às avaliações nas datas fixadas pode requerer uma prova substitutiva para cada disciplina, de acordo com o prazo fixado pelo calendário acadêmico, cabendo deferimento a Coordenação do Curso. • Pode ser concedida revisão de nota, por meio de requerimento, dirigido à Coordenação de Curso, no prazo de até 05 dias úteis após divulgação dos resultados. Não serão aceitos recursos em questões se respondidas a lápis. • Não será recebida prova antes de 30 minutos após o seu início. A permissão a submissão à prova por alunos retardatários será autorizada somente caso nenhum aluno houver entregado a prova. O aluno retardatário não gozará de tempo adicional para realização de sua prova. 			
INSTRUÇÕES ESPECÍFICAS			
1. Todos os códigos devem usar passagem de parâmetros.			
2. Os códigos devem ser declarados em pseudocódigo.			

Considere a estrutura abaixo para responder a questão abaixo:

```
struct nodo{
    int campo1;
    nodo *campo2;
};
Type strict nodo *NODOPTR;
NODOPTR p1,p2;
```

1. Em relação aos conteúdos ministrados em sala de aula em relação a disciplina Estruturas de Dados, marque V para proposições verdadeiras e F para as proposições falsas. No caso de proposições falsas, justifique sua resposta. (2,0 pontos)

(V) Um ponteiro é uma variável cujo valor é um endereço de memória do computador, e cujo valor está armazenado neste endereço.

(V) p1 = NULL;

(V) p2 = p1;

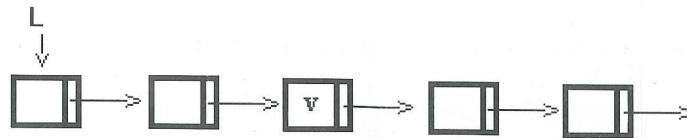
(F) p1=p2->campo1; //tentou atribuir um int a um ponteiro

(V) p1=p2->campo2.

(V) Passagem de parâmetros são argumentos usados para comunicação entre módulos. Existem dois tipos: **Por valor** e **Por referência**. Na passagem de parâmetro **por referência** as alterações efetuadas nos parâmetros formais alteram os parâmetros reais;

- ✓ (V) O operador & devolve o endereço na memória do seu operando.
 ✓ (F) O operador & é o complemento de *, ele devolve o valor da variável localizada no endereço que o segue. // & devolve o endereço da variável, não o valor dela;

2. Considere a estrutura de dados conforme exibido na figura abaixo. Use a figura para responder as questões a seguir:



- ✓ a. Considerando a figura acima como uma lista simplesmente encadeada, qual o problema o uso da seguinte instrução pode causar? Dado: $L = L \rightarrow \text{prox}$; (2,0 pontos);
- ✓ b. Agora suponha que exista um nó neutro no início de uma lista ligada. Esse nó não tem nenhum dado útil. Ele não é o primeiro nó e trata-se de um nó vazio apontado por L. Escreva um trecho de algoritmo que exclua o primeiro nó (o nó depois do nó neutro) (2,0 pontos);
- ✓ c. Considerando-se o nó apontado por L como neutro, escreva as instruções em pseudocódigo para excluir o nó V. Compare esta resposta com a resposta do item a. Elas são as mesmas? O que você pode concluir com isso? O uso de um nó neutro simplifica a operação em uma lista ligada? De que forma? (2,0 pontos);
- Ⓢ d. Considerando a figura acima como uma estrutura tipo fila, onde o final da fila está representado pelo nó apontado por L. Elabore uma função para enfileirar um elemento X na referida fila. (2,0 pontos);

"Não é o desafio com que nos deparamos que determina quem somos e o que estamos nos tornando, mas a maneira com que respondemos ao desafio. Somos combatentes, idealistas, mas plenamente conscientes, porque o ter consciência não nos obriga a ter teoria sobre as coisas: só nos obriga a sermos conscientes. Problemas para vencer, liberdade para provar. E, enquanto acreditarmos no nosso sonho, nada é por acaso."
 Henfil.

2a)

O comando $L = L \rightarrow \text{prox};$ fará com que a lista comece a partir do segundo elemento. A referência do primeiro será perdida, mas ela continuará alocada em memória. Se fosse anteriormente atribuída a uma variável auxiliar, poderia ser liberada após o comando.

b)

```

Void excluirPrimeiro (Lista *l) {
    //Assumindo lista preenchida, como no enunciado
    Lista *Auxiliar;
    Auxiliar = l->prox;
    l->prox = l->prox->prox;
    free(Auxiliar);
}


```

c)

```

Void excluirSegundo (Lista *l) {
    Lista *Auxiliar;
    Auxiliar = l->prox->prox;
    l->prox->prox = l->prox->prox->prox;
    free(Auxiliar);
}

```

Os algoritmos são semelhantes. Utilizando um nó neutro não houve necessidade de modificar a variável ponteiro externa à função ao excluir o primeiro nó. 

O que possibilitou a utilização de funções com retorno void. É aí desnecessário um retorno, se não tivéssemos esse nó neutro, ao excluir o primeiro teríamos que retornar o elemento que é o novo primeiro para atribuir à lista original.