

## CURSO ENGENHARIA DE COMPUTAÇÃO Avaliação N1

DATA:	05 /	' 05 <i>/</i>	/ 21
-------	------	---------------	------

1º semestre | 1º etapa

Turno: Tarde

DISCIPLINA: Estruturas de Dados. Nota:

Professor(a): Ernani Leite

Aluno (a): Jardel Pimentel Brasiliano

Matrícula:

## Orientações Gerais:

- **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.
  - 1. Em relação aos conteúdos ministrados em sala de aula em relação à disciplina Estruturas de Dados, marque V para proposições verdadeiras e F para as proposições falsas. No caso de proposições falsas, justifique sua resposta. (0,5 pontos)

Considere a estrutura abaixo para responder a questão abaixo:

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

( F) Na alocação dinâmica, os ponteiros permitem, durante a <u>execução</u> do programa, criar e desativar outras variáveis, denominadas *estáticas*.

Falso pois o ponteiro apenas vai servir para usar/alterar o valor alocado na variável.

```
(F)p1 = NULL.
```

Falso pois quando declarado sem passar um valor ele pega um valor na memoria.

(F)p1 = p2.

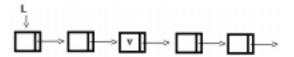
Falso pois são estâncias diferentes , logo possuem valores e local de memoria diferente. (F) P1 = p2 -> campo2.

Falso pois p1 e p2 não têm estâncias diferentes que não tem ligação.

( F) O operador & é o complemento de \*, ele devolve o valor da variável localizada no endereço que o segue.

Falso pois o & vai devolver o endereço na memoria.

**2.** Considere a estrutura de dados conforme exibido na figura abaixo. Use a figura para responder às 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->prox; (0,5 pontos);
  - **R** Se nenhum outro ponteiro seja referenciado L, o valor do 1] item será perdido e L vai ser o 2º item da lista.
- 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) (1,0 ponto);

```
R -

celula *lixo;

lixo = L->prox;

L->prox = lixo ->prox;

free(lixo);
```

c. Considerando-se o nó apontado por L como neutro, escreva as instruções em pseudocódigo para EXCLUIR o primeiro nó após o nó V. (IMPORTANTE: Você deve percorrer a lista até encontrar o no V) (1,0 ponto);

**R** -

{

onde Y é o valor a ser buscado.

```
busca_e_remove( int y, celula *le)

celula *p, *q;
p = le;
q = le->prox;
while( q != NULL && q->conteudo !=y ) {
    p = q;
    q = q->prox;
    }

if( q != NULL ) {
    celula *lixo;
    lixo = le->prox;
    le->prox;
}
```

```
free(lixo);
}
```

d. Agora, considerando a figura acima como uma estrutura tipo **FILA**, onde o final da **FILA** está representado por **L**. Elabore uma função para **DESENFILEIRAR** um elemento **X** na referida **FILA**. (1,0 ponto)

```
R -
    RegistroFila{
        Item item[maxTam]
        int frent, fim, tam;
}
Int Desempileira(Item *X, fila *fila){
        *X = pila->item[fila->frente]
        fila->frente = fila->frente + 1;
        return *X;
}
main(){
        int x = (valor);
        Fila fila;
        Desempileira(&x, &fila);
}
```

e. Agora, considerando a figura acima como uma estrutura tipo **PILHA**, onde o topo da pilha está representado pelo no apontado por **L**. Elabore uma função para **EMPILHAR** um elemento **X** na referida fila. **(1,0 ponto)**;

**R** -

"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."