**Universidade Federal de Uberlândia**
Faculdade de Computação
Av. João Naves de Ávila, 2121 – Bloco 1A – Campus Santa Mônica
38400-902 – Uberlândia, MG

**Resposta ao recurso apresentado pelo discente Claudionei Pereira da Cunha Filho, referente ao resultado do Exame de Suficiência na disciplina GAC060 – Estrutura de Dados 1, conforme Edital FACOM nº 18/2023, Processo SEI 23117.067977/2023-08.**

O discente apresentou recurso quanto ao resultado proferido pela Banca Examinadora, requisitando: 1) cópia da avaliação respondida pelo discente; 2) gabarito oficial considerado pela banca examinadora; e 3) considerações sobre as questões respondidas. Nesse sentido, seguem as considerações da Banca Examinadora em relação às questões respondidas pelo discente frente às respostas esperadas em cada questão. As respostas apresentadas pelo discente estão anexadas ao final deste documento.

**Questão 1.**

Nessa questão, pede-se que o discente apresente o resultado da execução de uma função escrita em linguagem C, denominada função **f**. A função **f** recebe como parâmetro uma lista de inteiros e retorna o maior elemento dessa lista. Assim, para a lista apresentada, a resposta esperada é o número inteiro 67. O discente apresentou como resposta uma lista de números. Portanto, a resposta apresentada pelo discente está **incorreta**.

**Questão 2.**

Nessa questão, o discente deveria indicar o erro existente na função apresentada. A função deveria contabilizar o número de ocorrências de um elemento na função. Nesse sentido, observa-se que existe um erro na lógica dessa função apresentada, notadamente na sexta linha da função, ou seja, no comando

```
while ((noLista->prox != NULL))
```

Isso ocorre pois o último nó de uma lista encadeada tem como característica principal o campo "prox" igual a NULL, já que não há nenhum outro nó depois dele. Como essa é a condição de parada para a contagem de ocorrências, quando chega no último nó, a função interrompe o comando **while** antes de executar a verificação do seu valor. Consequentemente, caso o último elemento da lista seja igual ao valor do qual se quer contar as ocorrências, ele não será contabilizado. Uma forma bem simples de corrigir essa função é substituir o comando citado anteriormente por

```
while ((noLista != NULL))
```

Assim, essa função não interromperá o comando **while** quando chegar no último nó, mas apenas quando chegar no nó seguinte ao último nó, cujo valor é NULL. Portanto, a resposta apresentada pelo discente está **incorreta**.

**Universidade Federal de Uberlândia**
Faculdade de Computação
Av. João Naves de Ávila, 2121 – Bloco 1A – Campus Santa Mônica
38400-902 – Uberlândia, MG

**Questão 3.**

A resposta apresentada pelo discente está **correta**, pois a função apresentada, de fato, retorna a lista revertida.

**Questão 4.**

Essa questão traz uma função que recebe uma sequência de elementos e realiza inserções e remoções em uma pilha, dependendo do elemento dado como entrada. Observa-se que todos os elementos diferentes de 3 serão empilhados e, somente quando o valor de entrada for igual a 3, é que um elemento da pilha será desempilhado (removido da pilha). Com isto, o programa exibirá na tela, ao final de sua execução, o valor 5. Destaca-se, ainda, que a ordem dos elementos restantes na pilha resultante, da base ao topo, será: 1, 4 e 6. Portanto, a resposta apresentada pelo discente está **incorreta**.

Considerando as correções apresentadas acima, registra-se que da correção da aludida prova teórica resultou em 25 pontos dos 100 possíveis. Portanto, conforme se preconiza o § Único, do Art. 200, das Normas Gerais de Graduação da Universidade Federal de Uberlândia, o candidato deve ser considerado **REPROVADO**.

Uberlândia, 13 de novembro de 2023.

Prof. Dr. Paulo Henrique Ribeiro Gabriel
Prof. Dr. Daniel Stefany Duarte Caetano
Prof. Dr. Jose Gustavo de Souza Paiva

**FACULDADE DE ENGENHARIA DE AGRIMENSURA E CARTOGRÁFICA**

**Disciplina:** Estrutura de Dados 1 **Valor:** 100 pt

**Aluno (a):** Claudionei Pereira da Cunha Filho **Data:** 31/10/23 **Nota:**

**ORIENTAÇÕES PARA A PROVA:** IDENTIFICADA QUALQUER TENTATIVA DE FRAUDE NA REALIZAÇÃO DA PROVA, ESTA SERÁ ZERADA. A INTERPRETAÇÃO DAS QUESTÕES FAZ PARTE DO PROCESSO AVALIATIVO. A PROVA É INDIVIDUAL E SEM CONSULTA. AS RESPOSTAS FINAIS DEVERÃO SER FEITAS À CANETA AZUL OU PRETA.

**QUESTÃO 01 (25PT):** Considere a seguinte representação de lista encadeada implementada na linguagem C, bem como a função f abaixo, e responda à questão abaixo:

**LISTA ENCADEADA**

```
typedef struct no {
    int valor;
    struct no *prox;
}No;

typedef struct lista {
    No *inicio;
}Lista;
```

```
int f(Lista *l) {
    if (l == NULL)
        return -2;
    No *noLista = l->inicio;
    if (noLista == NULL)
        return -1;
    int x = noLista->valor;
    while ((noLista != NULL)) {
        if (noLista->valor > x)
            x = noLista->valor;
        noLista = noLista->prox;
    }
    return x;
}
```

Considerando a lista `l = [ 2 55 67 3 5 1 12 6 12 ]`, recebida como parâmetro, responda o que a função f retornará.

A função f retornará como resposta a seguinte lista:

l = [2 55 3 3 1 1 6 12]

**QUESTÃO 02 (25PT):** Considere a função abaixo que, dado uma lista encadeada de números inteiros, conta todas as ocorrências de um determinado número, retornando o valor dessa contagem. O código compila normalmente, mas não funciona para todas as situações. Identifique qual é o erro, e escreva o código de uma proposta para corrigi-lo.

```
int contaOcorrencias(Lista *l,int ch) {
    if (l != NULL) {
        int ct = 0;
        No *noLista = l->inicio;
            if (noLista != NULL) {
            while ((noLista->prox != NULL)) {
                if (noLista->valor == ch)
                    ct++;
                noLista = noLista->prox;
            }
        return ct;
    }else
        return -1;
}
```

O código não armazena a contagem de repetições para cada valor da lista encadeada. A variável int ch não assume valor da lista antes de verificar possíveis repetições; Deveria haver um código com implementação da lista encadeada.

2.

Uma forma de se corrigir o código apresentado, é mostrada abaixo:

Lista encadeada:

```
typedef struct no{
        int valor;
        struct no *prox;
}
NO;

typedef struct lista{
        No *inicio;
}
Lista;
int contaOcorrencias(Lista *l, int ch){
    if (l != NULL){
        int ct = 0;
        int ocor[];
        No *noLista = l -> inicio;
            if (noLista != NULL){
                while ((noLista -> prox != NULL)){
                    if (noLista -> valor == ch)
                        ct++;
                    noLista = noLista -> prox;
                }
            }
            return ct;
            ocor[] = ct;
            else
                return -1;
    }
```

**QUESTÃO 03 (25PT):** A seguinte função recebe uma fila F como argumento e utiliza uma pilha P durante o processamento.

```
void fun(Fila *F) {// cria função
    Pilha *P;  // cria pilha P
    P = criarPilha();

    while (filaVazia(F) != false) {// repetidor enquanto fila tiver conteúdo
        push(P, dequeue(F)); // aloca valor da fila F na pilha P
    }
    while (pilhaVazia(P) != false) {// repetidor enquanto pilha tiver conteúdo
        enqueue(F, pop(P)); // aloca valor da pilha P na fila F
    }
}
```

Considere a fila F = {3,1,4,15,9,2,6,5}. Mostre como ficará a fila F após a execução dessa função.

Após executada a função fun, a fila F assumirá valores:
F = {5, 6, 2, 9, 15, 4, 1, 3}.

**QUESTÃO 04 (25PT):** Considere o programa abaixo, que utiliza uma estrutura de dados Pilha:

```
int main() {
    Pilha *p = criar();
    char opcao;// cria variável caractere
    int num; // cria variável inteiro
    int x; // cria variável inteiro
    scanf("%c",&opcao); // captura variável caracter opção
    while (opcao != 'S') { // comando repetidor enquanto o caracter na variável
                           opção for diferente de 'S' executa comandos abaixo
        scanf("%d",&num); // captura variável num
        if (num != 3) // comando se variável num é diferente de 3.
          push(p,num); // aloca o valor na pilha
        else // comando se variável num é igual a 3
          pop(p,&x); // aloca o valor na variável x.
        scanf("%c",&opcao); // captura variável caracter opção.
    }
    printf("O valor de x eh %d",x); // printa em tela o texto.
    return 0;
}
```

Dado a sequência de entrada:

```
'C', 1, 'C', 2, 'C', 3, 'C', 4, 'C', 5, 'C', 3, 'C', 6, 'S'
```

Escreva o que o programa exibirá na tela, e quais elementos estarão na pilha ao final da execução, da base até o topo (NESTA ORDEM).

O programa exibirá na tela: "O valor de x eh 3"
Os elementos que estarão na pilha são, em ordem da base para o topo: 1, 2, 'C', 4, 5, 'C', 6.