Tempo restante 0:57:27

Questão 1

Correto

Vale 10,00 ponto(s).

Programa para verificar o balanceamento de expressões aritméticas. Função Empilha.

Esta atividade visa desenvolver habilidades em estruturas de dados, especificamente em pilhas, e sua aplicação em problemas práticos. A tarefa é completar a implementação de uma função crucial para um programa que identifica se os delimitadores em expressões aritméticas, parênteses "()", chaves "{}" e colchetes "[]", estão corretamente balanceados.
O programa fornecido já possui a maior parte da lógica necessária para verificar o balanceamento de delimitadores em uma expressão. No entanto, ele está incompleto porque falta a implementação da função Empilha. Esta função é responsável por empilhar um delimitador à pilha cada vez que um delimitador aberto é encontrado em uma expressão aritmética fornecida pelo usuário na ***main.c***.

A pilha utilizada pelo programa está implementada em uma lista simplesmente encadeada cujo nó está definido abaixo. O topo da pilha está na cabeça desta lista:

```
// Tipo pilha
typedef struct no{
    char caracter;
    struct no *proximo;
} No;
```

O analista de testes da equipe montou um programa (função ***main()*** abaixo) para testar expressões aritméticas fornecidas pelo usuário.

```
// Função Main
int main(){
    char exp[50];
    int retorno;
    scanf("%49[^\n]", exp);
    retorno = identifica_formacao(exp);
    if (retorno == 1)
        printf("BALANCEADA\n");
    else
        printf("DESBALANCEADA\n");
    return 0;
}
```

Você está encarregado de desenvolver a função ***Empilha*** conforme abaixo.

```
No *Empilha(No *pilha, char x){
    SEU CÓDIGO VAI AQUI!
}
```

Em suma, sua tarefa é postar o código da função ***No *Empilha(No *pilha, char x)*** . As outras funções do programa estão prontas e uma vez que você concluir esta função, o programa de testes será executado corretamente e gerará saídas conforme os exemplos apresentados.

POSTE APENAS O CÓDIGO DA FUNÇÃO REQUERIDA. NÃO POSTE NADA A MAIS (main, includes ou outro código qualquer).

For example:

Input	Result
`(a + b) * [8 + (c - d)]`	BALANCEADA
`((a + b)`	DESBALANCEADA
`(a * {b + c}) / [d - e]`	BALANCEADA

Answer: (penalty regime: 0 %)

```
No *Empilha(No *pilha, char x) {
    No *novo = (No *)malloc(sizeof(No));
    if (novo == NULL) {
        printf("Erro de memória\n");
        exit(1);
    }
    novo->caracter = x;
```

VERIFICAR

	Input	Expected	Got	
✔	`(a + b) * [8 + (c - d)]`	BALANCEADA	BALANCEADA	✔
✔	`((a + b)`	DESBALANCEADA	DESBALANCEADA	✔
✔	`(a * {b + c}) / [d - e]`	BALANCEADA	BALANCEADA	✔

Passou em todos os teste! ✔

Questão 2

Correto

Vale 10,00 ponto(s).

Programa para verificar o balanceamento de expressões aritméticas. Função Desempilha.

Esta atividade visa desenvolver habilidades em estruturas de dados, especificamente em pilhas, e sua aplicação em problemas práticos. A tarefa é completar a implementação de uma função crucial para um programa que identifica se os delimitadores em expressões aritméticas, parênteses "()", chaves "{}" e colchetes "[]", estão corretamente balanceados.
O programa fornecido já possui a maior parte da lógica necessária para verificar o balanceamento de delimitadores em uma expressão. No entanto, ele está incompleto porque falta a implementação da função Desempilha. Esta função é responsável por desempilhar a pilha cada vez que um delimitador fechado é encontrado em uma expressão aritmética fornecida pelo usuário na ***main.c***.

A pilha utilizada pelo programa está implementada em uma lista simplesmente encadeada cujo nó está definido abaixo. O topo da pilha está na cabeça desta lista:

```
// Tipo pilha
typedef struct no{
    char caracter;
    struct no *proximo;
} No;
```

O analista de testes da equipe montou um programa (função ***main()*** abaixo) para testar expressões aritméticas fornecidas pelo usuário.

```
// Função Main
int main(){
    char exp[50];
    int retorno;
    scanf("%49[^\n]", exp);
    retorno = identifica_formacao(exp);
    if (retorno == 1)
        printf("BALANCEADA\n");
    else
        printf("DESBALANCEADA\n");
    return 0;
}
```

Você está encarregado de desenvolver a função ***Desempilha*** conforme abaixo.

```
No *Desempilha(No *pilha){
    SEU CÓDIGO VAI AQUI!
}
```

Em suma, sua tarefa é postar o código da função ***No *Desempilha(No *pilha)*** . As outras funções do programa estão prontas e uma vez que você concluir esta função, o programa de testes será executado corretamente e gerará saídas conforme os exemplos apresentados.

POSTE APENAS O CÓDIGO DA FUNÇÃO REQUERIDA. NÃO POSTE NADA A MAIS (main, includes ou outro código qualquer).

For example:

Input	Result
`(a + b) * [8 + (c - d)]`	BALANCEADA
`((a + b)`	DESBALANCEADA
`(a * {b + c}) / [d - e]`	BALANCEADA

Answer: (penalty regime: 0 %)

```
No *Desempilha(No *pilha) {
    if (pilha) {
        No *temp = pilha;
        pilha = pilha->proximo;
        free(temp);
    }
    return pilha;
```

VERIFICAR

	Input	Expected	Got	
✔	`(a + b) * [8 + (c - d)]`	BALANCEADA	BALANCEADA	✔
✔	`((a + b)`	DESBALANCEADA	DESBALANCEADA	✔
✔	`(a * {b + c}) / [d - e]`	BALANCEADA	BALANCEADA	✔

Passou em todos os teste! ✔

Questão 3

Correto

Vale 80,00 ponto(s).

Programa para verificar o balanceamento de expressões aritméticas: Função Avalia Expressão.

Esta atividade visa desenvolver habilidades em estruturas de dados, especificamente em pilhas, e sua aplicação em problemas práticos. A tarefa é completar a implementação de uma função crucial para um programa que identifica se os delimitadores em expressões aritméticas, parênteses "()", chaves "{}" e colchetes "[]", estão corretamente balanceados.
O programa fornecido já possui a maior parte da lógica necessária para verificar o balanceamento de delimitadores em uma expressão. No entanto, ele está incompleto porque falta a implementação da função identifica_formacao. Esta função é responsável por avaliar se os parênteses, colchetes e chaves de uma expressão aritmética estão balanceados. A expressão aritmética é fornecida como argumento de entrada como verificado na função ***main.c*** abaixo.

A pilha utilizada pelo programa está implementada em uma lista simplesmente encadeada cujo nó está definido abaixo. O topo da pilha está na cabeça desta lista:

```
// Tipo pilha
typedef struct no{
    char caracter;
    struct no *proximo;
} No;
```

O analista de testes da equipe montou um programa (função ***main()*** abaixo) para testar expressões aritméticas fornecidas pelo usuário.

```
// Função Main
int main(){
    char exp[50];
    int retorno;
    scanf("%49[^\n]", exp);
    retorno = identifica_formacao(exp);
    if (retorno == 1)
        printf("BALANCEADA\n");
    else
        printf("DESBALANCEADA\n");
    return 0;
}
```

Você está encarregado de desenvolver a função ***identifica_formacao*** conforme abaixo. Esta função deve retornar 0, caso haja algum parêntese, colchete ou chave desbalanceado ou 1, no caso de todos estarem balanceados.

```
int identifica_formacao(char x[]){
    SEU CÓDIGO VAI AQUI!
}
```

O programa já possui as seguintes funções que você poderá utilizar em seu código:

1. Função para desempilhar:

```
// Função Desempilha
// Assinatura:
    No *Desempilha(No *pilha);
//Retorno
    // Topo da pilha
```

2. Função para empilhar:

```
// Função Empilha
// Assinatura:
    No *Empilha(No *pilha, char x);
//Retorno
    // Topo da pilha
```

3. Função para verificar o casamento de padrão de aberturas e fechamentos de parênteses ("()"), colchetes ("[]") e chaves ("{}").

// Função identifica_formacao
// Assinatura:
int forma_par(char f, char d);
//Retorno
// 0 - Caso f não seja o fechamento de d
// 1 - Caso f seja o fechamento de d

Em suma, sua tarefa é postar o código da função ***int identifica_formacao(char x[])*** . As outras funções do programa estão prontas e uma vez que você concluir esta função, o programa de testes será executado corretamente e gerará saídas conforme os exemplos apresentados.

POSTE APENAS O CÓDIGO DA FUNÇÃO REQUERIDA. NÃO POSTE NADA A MAIS (main, includes ou outro código qualquer).

For example:

Input	Result
`(a + b) * [8 + (c - d)]`	BALANCEADA
`((a + b)`	DESBALANCEADA
`(a * {b + c}) / [d - e]`	BALANCEADA

Answer: (penalty regime: 0 %)

```
int identifica_formacao(char x[]) {
    No *pilha = NULL;
    for (int i = 0; x[i] != '\0'; i++) {
        if (x[i] == '(' || x[i] == '[' || x[i] == '{') {
            pilha = Empilha(pilha, x[i]);
        } else if (x[i] == ')' || x[i] == ']' || x[i] == '}') {
            if (pilha == NULL || !forma_par(x[i], pilha->caracter)) {
                return 0; // Desbalanceada
            }
            pilha = Desempilha(pilha);
        }
    }
    if (pilha == NULL) {
        return 1; // Balanceada
    } else {
        return 0; // Desbalanceada
    }
}
```

VERIFICAR

	Input	Expected	Got	
✔	`(a + b) * [8 + (c - d)]`	BALANCEADA	BALANCEADA	✔
✔	`((a + b)`	DESBALANCEADA	DESBALANCEADA	✔
✔	`(a * {b + c}) / [d - e]`	BALANCEADA	BALANCEADA	✔

Passou em todos os teste! ✔