FATEC IPIRANGA

PASTOR ENÉAS TOGNINI

ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO ESTRUTURADA E MODULAR

PROFESSOR CARLOS HENRIQUE VERISSIMO PEREIRA

MILENA MITIE AOKI

SÃO PAULO, SP
2024

SUMÁRIO

N1 - DESAFIO PEM - BUGS DA HP12C

1. CÓDIGO REFATORADO

```
#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

#include <string.h>


#define TAMANHO_PILHA 4


// Função para exibir a pilha

void exibirPilha(int pilha[]) {

    printf("Pilha: [T: %d] [Z: %d] [Y: %d] [X: %d]\n",

           pilha[3], pilha[2], pilha[1], pilha[0]);

}


// Função para empurrar valores na pilha

void empurrar(int pilha[], int valor) {

    for (int i = TAMANHO_PILHA - 1; i > 0; i--) {

        pilha[i] = pilha[i - 1];

    }

    pilha[0] = valor;

}


// Função para executar a operação entre os dois primeiros operandos da pilha
```

```
int executarOperacao(int pilha[], char operador) {
    int resultado;

    // Verifica se há operandos suficientes para a operação
    if (pilha[1] == 0 && (operador == '+' || operador == '-' || operador == '*' || operador
== '/')) {
        printf("Erro: Operação inválida. Não há operandos suficientes.\n");
        return -1;
    }

    // Opera entre o penúltimo (pilha[1]) e o último (pilha[0]) valor
    switch (operador) {
        case '+':
            resultado = pilha[1] + pilha[0];
            break;
        case '-':
            resultado = pilha[1] - pilha[0];
            break;
        case '*':
            resultado = pilha[1] * pilha[0];
            break;
        case '/':
            if (pilha[0] == 0) {
                printf("Erro: Divisão por zero não permitida.\n");
```

```
            return -1;
        }
        resultado = pilha[1] / pilha[0];
        break;
    default:
        printf("Erro: Operador inválido.\n");
        return -1;
}

// Atualiza a pilha com o resultado da operação
pilha[0] = resultado;
for (int i = 1; i < TAMANHO_PILHA - 1; i++) {
    pilha[i] = pilha[i + 1]; // Faz o shift nos valores para liberar espaço
}
pilha[TAMANHO_PILHA - 1] = 0; // Limpa o topo da pilha

return 0; // Retorna sucesso
}

int main() {
    int pilha[TAMANHO_PILHA] = {0, 0, 0, 0}; // Inicializando a pilha com zeros
    char entrada[100]; // Para armazenar a entrada do usuário
    char continuar;
```

```
    printf("Bem-vindo à Calculadora Fatec-HP12c!\n");


    do {
        printf("\nDigite a expressão em formato RPN (ex: 5 1 2 + 4 * + 3) ou 'sair' para encerrar: ");
        fgets(entrada, sizeof(entrada), stdin); // Lê a entrada do usuário


        // Verificar se o usuário deseja sair
        if (strncmp(entrada, "sair", 4) == 0) {
            break;
        }


        // Dividir a entrada em tokens (números e operadores)
        char *token = strtok(entrada, " ");
        while (token != NULL) {
            // Verifica se o token é um número (operando)
            if (isdigit(token[0]) || (token[0] == '-' && isdigit(token[1]))) {
                int valor = atoi(token); // Converte o token para número inteiro
                empurrar(pilha, valor); // Empurra o número para a pilha
                exibirPilha(pilha); // Exibe o estado da pilha
            }
            // Caso contrário, trata-se de um operador
            else {
```

```
            if (executarOperacao(pilha, token[0]) == -1) {
                // Se houver erro na operação, encerrar o loop atual
                break;
            }
            exibirPilha(pilha); // Exibe o estado da pilha
        }
        token = strtok(NULL, " "); // Avança para o próximo token
    }

    printf("\nResultado final: %d\n", pilha[0]); // Exibe o resultado final

    // Pergunta ao usuário se deseja realizar outra operação
    printf("\nDeseja realizar outra operação? (s/n): ");
    scanf(" %c", &continuar);
    getchar(); // Limpa o buffer

} while (continuar == 's' || continuar == 'S');

// Mensagem de encerramento
printf("Obrigado por usar nossa Calculadora Fatec-HP12c!\n");

return 0;
}
```

2. ERROS E CORREÇÃO

Erro 1: Aceitação de entradas inválidas (strings ou fórmulas RPN incorretas)

Descrição: O programa aceita qualquer string como entrada, incluindo operadores sem operandos suficientes e operações inválidas. Isso resulta em erros de execução ou resultados incorretos.

Correção: Foi adicionada uma validação que verifica se o número de operandos na pilha é suficiente antes de processar o operador. Caso contrário, o programa exibe uma mensagem de erro e não realiza a operação. Isso garante que a entrada siga as regras da notação polonesa reversa (RPN).

Exemplo:

Entrada inválida: 5 +

Nova resposta: "Erro: Operação inválida. Não há operandos suficientes."

Erro 2: Manipulação incorreta da pilha após operações

Descrição: Após realizar uma operação, a pilha não estava sendo corretamente "shiftada", resultando em inconsistências quando outras operações eram executadas.

Correção: Foi corrigido o processo de ajuste da pilha após cada operação. O valor resultante da operação é colocado na posição X (base da pilha), e os demais valores são deslocados corretamente, mantendo o estado esperado da pilha.

Exemplo:

Entrada anterior: 5 3 +

Estado da pilha esperado: Pilha: [T: 0] [Z: 0] [Y: 0] [X: 8]

Erro 3: Ausência de tratamento para divisão por zero

Descrição: O programa realizava a operação de divisão sem verificar se o divisor era zero, o que resultava em falha no cálculo.

Correção: Foi adicionada uma verificação específica para a divisão. Caso o divisor seja zero, o programa exibe uma mensagem de erro e não executa a operação.

Exemplo:

Entrada: 8 0 /

Nova resposta: "Erro: Divisão por zero não permitida."

Erro 4: Falta de feedback claro para operadores inválidos

Descrição: O programa não fornecia mensagens claras quando operadores inválidos eram inseridos, o que poderia confundir o usuário.

Correção: Foi implementada uma verificação para operadores não reconhecidos. Agora, o programa exibe uma mensagem de erro quando um operador inválido é detectado, sem tentar realizar uma operação.

Exemplo:

Entrada: 5 3 $

Nova resposta: "Erro: Operador inválido."

Erro 5: Resultado final incorreto devido à manipulação de strings erradas

Descrição: Ao lidar com strings de entrada mal formatadas, o programa executava operações indevidas ou exibia resultados inconsistentes.

Correção: Foi ajustada a função "strtok" para garantir que as entradas numéricas e os operadores sejam devidamente processados. Operações inválidas são descartadas, e a pilha é exibida corretamente a cada passo.

Exemplo:

Entrada anterior: 5 2 + 3

Resposta anterior: Incorreta, pois a entrada "3" era tratada erroneamente após a soma.

Nova resposta: Corrigida com o fluxo esperado da operação.

Erro 6: Não exibição do estado correto da pilha ao longo das operações

Descrição: O estado da pilha nem sempre refletia corretamente o que estava acontecendo com os operandos e operadores processados.

Correção: O estado da pilha é agora exibido de forma consistente após cada operação e inserção de operandos, garantindo que o usuário possa acompanhar as mudanças.

Exemplo:

Entrada anterior: 5 1 2 + 4 *

Exibição correta: Pilha: [T: 0] [Z: 0] [Y: 28] [X: 0]