Estrutura de Dados 1

Atividade Avaliativa 1

Pilha em C

A pilha é uma das estruturas mais simples e mais versáteis dentre as utilizadas na computação. A pilha éuma estrutura em que a entrada e a saída de dados se dão pela mesma extremidade, chamada de topo dapilha. São estruturas conhecidas como Last In, First Out (LIFO), que pode ser traduzido por Último a Entrar,Primeiro a Sair.

Considere a implementação em linguagem C da estrutura de dados pilha com tamanho fixo e a funçãodesempilhar:

In [ ]:
```c
#define tam 10

struct pilha {
    int dados[tam];
    int primeiro;
    int ultimo;
};

pilha p;

void desempilhar() {
    p.dados[p.ultimo-1] = 0;
    p.ultimo--;
}
```

Em determinado instante da execução do programa, que utiliza a estrutura pilha e o procedimento paradesempilhar, o usuário selecionou a opção "desempilhar", porém a estrutura pilha estava vazia.

Diante da situação especificada, informe a estrutura utilizada pelo desenvolvedor para representar apilha e descreva o comportamento da pilha mediante a execução da função desempilhar.

In [ ]:

A estrutura utilizada é struct, um tipo de estrutura de dado que permite combinar diferentes tipos de variáveis em uma única estrutura. Ela é usada para agrupar elementos relacionados, de forma que possam ser tratados como uma unidade lógica.

No código acima, utilizado para representar uma pilha de tamanho fixo, foi definido uma struct chamada "pilha" que contém três variáveis: "dados", um array de tamanho 10 e do tipo primitivo int; "primeiro" do tipo primitivo int e "ultimo" também do tipo primitivo int, porém, a struct pode abrigar variáveis de todos os tipos primitivos em C, a depender da necessidade do programador.

As structs são muito úteis quando há necessidade de agrupar várias variáveis relacionadas em uma única entidade, facilitando o armazenamento e manipulação dessas informações. A pilha é implementada como um array de inteiros com tamanho definido pela constante "tam". A struct possui dois membros adicionais: "primeiro" e "ultimo", que indicam a posição do primeiro elemento e do último elemento adicionado na pilha, respectivamente.

Em seguida, é declarada uma variável global chamada "p" do tipo "pilha", que será utilizada para manipular a pilha.

Além disso, o código inclui uma função chamada "desempilhar()", que remove o elemento do topo da pilha. Essa função acessa o array de dados da pilha usando o membro "ultimo" como índice e atribui o valor 0 para remover o elemento. Em seguida, o membro "ultimo" é decrementado para indicar a nova posição do último elemento na pilha.

Vale ressaltar que o código fornecido está incompleto e não inclui a implementação de outras operações básicas de uma pilha, como empilhar (adicionar elemento ao topo) e verificar se a pilha está vazia ou cheia. Essas funcionalidades podem ser adicionadas ao código conforme necessário para tornar a implementação completa de uma pilha em C, a exemplo do código a seguir:

In [ ]:
```c
#include <stdio.h>

#define tam 10



struct pilha {
    int dados[tam];
    int topo;
};

struct pilha p;

void inicializar_pilha() {
    p.topo = -1; // Inicializa o topo da pilha como -1 (indicando pilha vazia)
}

int pilha_vazia() {
    return (p.topo == -1); // Retorna 1 se a pilha estiver vazia, 0 caso contrário
}

int pilha_cheia() {
    return (p.topo == tam - 1); // Retorna 1 se a pilha estiver cheia, 0 caso contrário
}

void empilhar(int valor) {
    if (pilha_cheia()) {
        printf("Erro: Pilha cheia\n");
        return;
    }

    p.topo++;
    p.dados[p.topo] = valor;
}

void desempilhar() {
    if (pilha_vazia()) {
        printf("Erro: Pilha vazia\n");
        return;
    }

    p.dados[p.topo] = 0; // Limpa o valor do elemento no topo da pilha
    p.topo--;
}

void imprimir_pilha() {
    if (pilha_vazia()) {
        printf("Pilha vazia\n");
        return;
    }

    printf("Elementos da pilha:\n");
    int i;
    for (i = p.topo; i >= 0; i--) {
        printf("%d\n", p.dados[i]);
    }
}

int main() {
    inicializar_pilha();

    empilhar(5);
    empilhar(10);
    empilhar(3);
    empilhar(7);

    imprimir_pilha(); // Saída esperada: 7 3 10 5

    desempilhar();
    desempilhar();

    imprimir_pilha(); // Saída esperada: 10 5

    return 0;
}
```

Nesse exemplo, a função `inicializar_pilha()` é utilizada para definir o topo da pilha como -1, indicando que a pilha está vazia. A função `pilha_vazia()` verifica se a pilha está vazia, retornando 1 em caso positivo e 0 caso contrário. A função `pilha_cheia()` verifica se a pilha está cheia, retornando 1 se estiver cheia e 0 caso contrário.

As funções `empilhar()` e `desempilhar()` são responsáveis por adicionar elementos à pilha e remover o elemento do topo, respectivamente. Elas verificam se a pilha está cheia ou vazia antes de realizar a operação.

A função `imprimir_pilha()` percorre a pilha a partir do topo e imprime os elementos na ordem em que foram empilhados.

No exemplo do `main()`, são realizadas algumas operações de empilhar e desempilhar, seguidas da impressão dos elementos da pilha para verificar o resultado.

Lembrando que esse é apenas um exemplo básico de implementação de uma pilha em C. Existem outras variações e melhorias que podem ser feitas dependendo dos requisitos específicos.

In [ ]: