Exercício de Pilha

27 de setembro de 2022

Considere o código abaixo. Neste código o *push* e o *pop* não verificam se existe espaço ou se a pilha está vazia. Isto está a cargo do usuário da pilha.

```
typedef struct Pilha{
    int *v;
    int max, top;
}Pilha;
Pilha * newPilha(int max){
    Pilha * aux = (Pilha *)malloc(sizeof(Pilha));
    aux->v = (int *)malloc(sizeof(int)*max);
    aux->max=max;
    aux->top=0;
    return aux;
}
void push(Pilha *p,int x){
    p \rightarrow v[p \rightarrow top ++] = x;
}
int pop(Pilha *p){
    return p->v[--p->top];
}
```

Neste código, o elemento é inserido sempre na primeira posição livre, que é inicialmente a posição zero. Os elementos são inseridos da esquerda para a direita, conforme mostra a Figura 1.

Já o pop remove o elemento que está na posição top-1 e em seguida faz um top--.

Altere o código para que o push insira um elemento na posição livre mais à direita conforme mostrado na Figura 2. Note que o pop vai remover seguindo a regra FILA ($first\ in\ las\ out$).

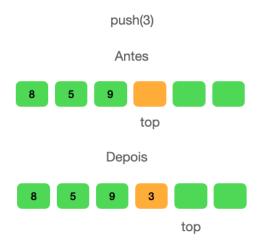


Figura 1: Push Tradicional

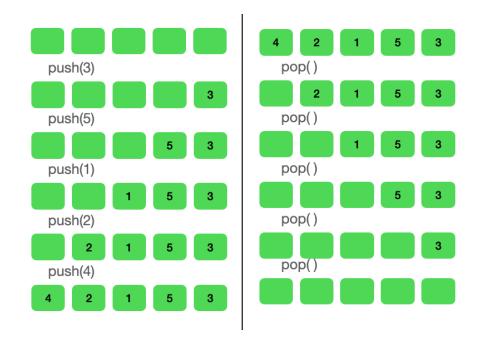


Figura 2: Nesta figura temos o *push* e o *pop* modificados. Note que, neste caso, o elemento é inserido na posição livre mais a direita. Os elementos são inseridos, então, da direita para a esquerda. O *pop* precisa ser alterado para que a pilha continue funcionando como *first in last out*.