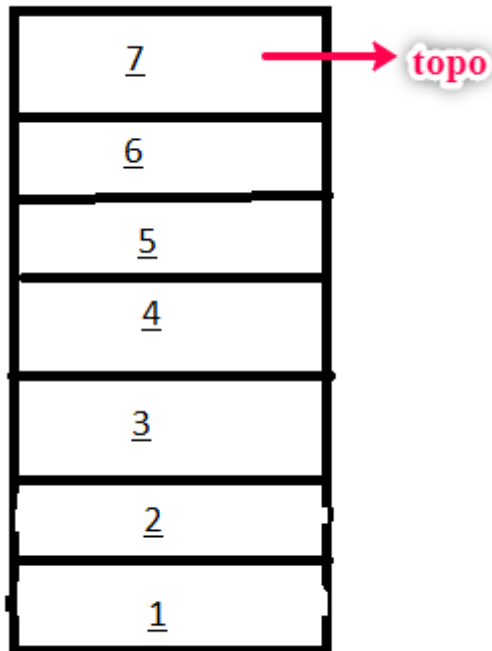


## 2-PILHA

-PILHA = STACK

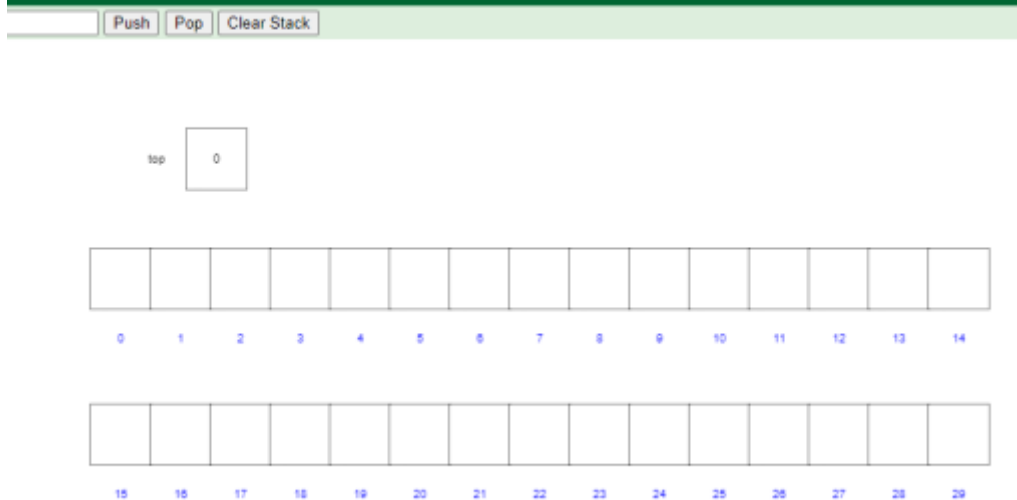
-FILO = FIRST IN/LAST OUT

- LIFO = LAST IN FIRST OUT



- Os elementos são inseridos no topo, ou seja, sempre que um novo elemento é inserido ele ocupa o topo da pilha.
- So temos acesso ao elemento que esta no topo.
- O processo de inserir um elemento é chamado de push.
- O processo de remover um elemento é chamado de pop.

### Stack (Array Implementaion)



- Se o topo for igual ao tamanho da pilha, a pilha esta cheia. Ou seja, antes de adicionar um elemento temos que verificar se o topo eh menor do que o tamanho da pilha.

- Para podermos remover, tambem precisamos fazer uma verificação. Para a gente poder remover o topo precisar ser  $\geq 0$ , pois 0 é nosso ultimo elemento da pilha, não conseguimos remover o que esta vazio.

## FUNÇÕES DA PILHA

```

/*****FUNÇÕES*****/
//LISTANDO ELEMENTOS
void lista_elementos(){
    printf("\n=====PILHA ATUAL=====\\n");
    for(int i = 0; i < TAMPILHA; i++){
        printf("-");
        printf("|%d|", pilha[i]);
        printf("-");
    }
    printf("\\n|Topo| = %d", topo);
    printf("\\n=====\\n");
}

//fim_lista_elementos

//INSERINDO NA PILHA
void push(){

    int val;
    printf("Informe o valor:\\n");
    scanf("%d", &val);

    if(topo < TAMPILHA){
        pilha[topo] = val;
        topo++;
        lista_elementos();
    }else{
        printf("Pilha Cheia!\\n");
    }

}

// fim_ push

//REMOVENDO DA PILHA

void pop(){
    if(topo >= 0 ){
        pilha[topo-1] = 0;
        topo--;
        lista_elementos();
    }else{
        printf("Pilha Vazia! Favor limpe a pilha[4] ou [-1] para sair.");
    }
}

//LIMPANDO PILHA
void clear(){
    for(int i = 0; i < TAMPILHA; i++){
        pilha[i] = 0;
    }
    topo = 0;
}

```

```

int main(){//inicio_main
printf("\n*****\n");

int opcao = 0;

do{
printf("Selecione a opcao:\n\n");
printf("[1] - inserir(push):\n");
printf("[2] - Remover(pop):\n");
printf("[3] - Listar Elementos:\n");
printf("[4] - Limpar a pilha:\n");
printf("[-1] - Sair:\n");
printf("OPCAO: ");
scanf("%d",&opcao);

switch (opcao)
{
case 1:
push();
break;
case 2:
pop();
break;
case 3:
lista_elementos();
break;
case 4:
clear();
break;
case -1:|
break;

default:
printf("Opcao invalida");
}

}while(opcao != -1);//condição de parada.

printf("\n*****\n");
printf("\n\n");
return 0;
} //fim_main

```

\*\*\*\*\*

Selecione a opcao:

- [1] - inserir(push):
- [2] - Remover(pop):
- [3] - Listar Elementos:
- [4] - Limpar a pilha:
- [-1] - Sair:

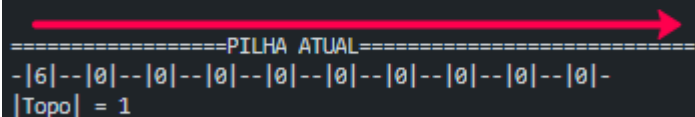
OPCAO: 1

Informe o valor:

6

SENTIDO PILHA

=====PILHA ATUAL=====



-|6|--|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|-

|Topo| = 1

Selecione a opcao:

- [1] - inserir(push):
- [2] - Remover(pop):
- [3] - Listar Elementos:
- [4] - Limpar a pilha:
- [-1] - Sair:

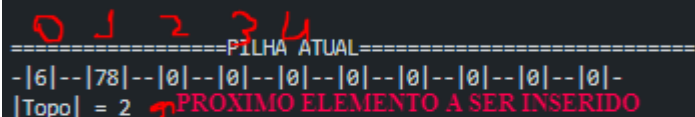
OPCAO: 1

Informe o valor:

78

0 1 2 3 4

=====PILHA ATUAL=====



-|6|--|78|--|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|-

|Topo| = 2

PROXIMO ELEMENTO A SER INSERIDO

Selecione a opcao:

- [1] - inserir(push):
- [2] - Remover(pop):
- [3] - Listar Elementos:
- [4] - Limpar a pilha:
- [-1] - Sair:

OPCAO: []

```
=====PILHA ATUAL=====
-|6|--|78|--|42|--|0|--|0|--|0|--|0|--|0|--|0|-
|Topo| = 3
=====

Selecione a opcao:

[1] - inserir(push):
[2] - Remover(pop):
[3] - Listar Elementos:
[4] - Limpar a pilha:
[-1] - Sair:
OPCAO: 2

=====PILHA ATUAL=====
-|6|--|78|--|0|--|0|--|0|--|0|--|0|--|0|--|0|-
|Topo| = 2
=====

Selecione a opcao:

[1] - inserir(push):
[2] - Remover(pop):
[3] - Listar Elementos:
[4] - Limpar a pilha:
[-1] - Sair:
OPCAO: 2

=====PILHA ATUAL=====
-|6|--|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|-
|Topo| = 1
=====

Selecione a opcao:

[1] - inserir(push):
[2] - Remover(pop):
[3] - Listar Elementos:
[4] - Limpar a pilha:
[-1] - Sair:
OPCAO: 2

=====PILHA ATUAL=====
-|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|--|0|-
|Topo| = 0
=====
```

Diagram illustrating stack operations (push and pop) using a stack structure. The stack is represented by an array of 10 slots, with the top element indicated by the `|Topo|` index.

**Initial State:** The stack contains elements 6, 78, and 42. `|Topo| = 3`.

**Operation 1 (Pop):** The user selects option 2 (Remover(pop)). The top element (42) is removed. The new top element is 78. `|Topo| = 2`.

**Operation 2 (Pop):** The user selects option 2 (Remover(pop)). The top element (78) is removed. The new top element is 6. `|Topo| = 1`.

**Operation 3 (Pop):** The user selects option 2 (Remover(pop)). The top element (6) is removed. The stack is now empty. `|Topo| = 0`.

**Final State:** The stack is empty. `|Topo| = 0`.