Campus Cachoeiro de Itapemirim	
Curso Técnico em Informática	
Disciplina: Programação 1	**Professor**: Rafael Vargas Mesquita
Lista 24 de Exercícios – Pilhas	

LISTA 24

1. Comentar a função **push**. Explique o que acontece em cada linha de código:

```
1. void push (Pilha *p, int v){
2.    No *novo = (No*) malloc(sizeof(No));
3.    novo->info = v;
4.    novo->prox = p->prim;
5.    p->prim = novo;
6. }
```

1. Assinatura da função `push` (função para inserção de um novo elemento na pilha). Esta função recebe como parâmetros o ponteiro (`*p`) para uma pilha e o valor (`v`) a ser inserido na pilha;
2. Declaração de um ponteiro para um elemento da pilha, bem como, por meio da função `malloc` a alocação de espaço na memória para armazenamento deste elemento;
3. Atribuição do valor (`v`) como a informação (`novo->info`) armazenada pelo novo elemento da pilha (`novo`). Como novo é um ponteiro para o registro No, então deve ser utilizado o operador '`->`' para acessar as variáveis de novo.
4. Atribuição do atual primeiro elemento da pilha (`p->prim`) como o próximo (`novo->prox`) armazenado pelo novo elemento da pilha (`novo`). Como novo é um ponteiro para o registro No, então deve ser utilizado o operador '`->`' para acessar as variáveis de novo.
5. Atualização do primeiro elemento da pilha (`p->prim`) como o novo elemento inserido (`novo`);
6. Final da função.

Campus Cachoeiro de Itapemirim	
Curso Técnico em Informática	
Disciplina: Programação 1	**Professor**: Rafael Vargas Mesquita
Lista 24 de Exercícios – Pilhas	

2. Comentar a função **pop**. Explique o que acontece em cada linha de código:

```
1. int pop (Pilha *p){
2.    No* novo_prim;
3.    No* removido;
4.    int valor;

5.    removido = p->prim;
6.    novo_prim = p->prim->prox;

7.    valor = p->prim->info;
8.    p->prim = novo_prim;
9.    free(removido);
10.   return valor;
11.}
```

1. Assinatura da função `pop` (função para remoção do elemento do topo da pilha). Esta função recebe como parâmetro o ponteiro (`*p`) para uma pilha. Não é preciso receber o valor do elemento a ser removido, pois em uma pilha o elemento removido é sempre o primeiro, ou o elemento do topo. É importante notar que esta função retorna o valor da informação armazenada pelo elemento removido;
2. Declaração de uma variável auxiliar (`novo_prim`) ponteiro para o novo primeiro elemento da pilha;
3. Declaração de uma variável auxiliar (`removido`) ponteiro para o elemento a ser removido da pilha;
4. Declaração de uma variável do tipo int para guardar a informação do elemento removido da pilha;
5. Atribuição do primeiro elemento da pilha (`p->prim`) como sendo a variável auxiliar ponteiro (`removido`), criada para apontar para o elemento a ser removido da pilha;
6. Atribuição do elemento correspondente ao próximo do primeiro elemento da pilha (`p->prim->prox`) como sendo a variável auxiliar ponteiro (`novo_prim`), criada para apontar para o elemento a ser o novo primeiro elemento da pilha;
7. Atribuição da informação (`p->prim->info`) armazenada pelo antigo primeiro elemento como a informação (`valor`) do elemento que será removido.
8. Atualização do primeiro elemento da pilha (`p->prim`) como o novo primeiro elemento (`novo_prim`);
9. A função `free` libera o espaço da memória no qual estava sendo armazenado o ponteiro removido, ou seja, o antigo primeiro elemento da pilha (`p->prim`).
10. Retornando o valor do elemento removido da pilha;
11. Final da função.

Campus Cachoeiro de Itapemirim	
Curso Técnico em Informática	
Disciplina: Programação 1	**Professor**: Rafael Vargas Mesquita
Lista 24 de Exercícios – Pilhas	

3. Ilustrar como fica uma pilha após a execução de cada passo da seqüência de funções:
   inicializa(pilha)
   push(pilha, 45)
   push(pilha, 22)
   pop(pilha)
   push(pilha, 12)
   pop(pilha)

		22		12	
	45	45	45	45	45
inicializa(pilha)	push(pilha, 45)	push(pilha, 22)	pop(pilha)	push(pilha, 12)	pop(pilha)