file:phila01.c

```c
#include <stdio.h>
#include <stdlib.h>
#include "pilha01.h"

struct lista
{
	float info;
	struct lista* Link;
};

//Representada como uma lista siempre apuntado para o ultimo elemento
struct pilha
{
	Lista* LinkTopo;
};

Pilha* criar_pilha()
{
	Pilha* argPilha = (Pilha*)malloc(sizeof(Pilha));
	argPilha->LinkTopo = NULL;
	return argPilha;
}

void push(Pilha* argPilha, float Valor)
{
	Lista* novoElemento = (Lista*)malloc(sizeof(Lista));
	novoElemento->info = Valor;
	novoElemento->Link = argPilha-> LinkTopo;
	argPilha->LinkTopo = novoElemento;
}

float pop(Pilha* argPilha) //Retorna o elemento que foi removido
{
	Lista* t;
	float Valor;
	if(vazia(argPilha))
	{
		printf("Pilha vazia\n");
		exit(1);
	}
	t = argPilha->LinkTopo;
	Valor = t->info;
	argPilha->LinkTopo = t->Link;
	free(t);
	return Valor;
}

int vazia(Pilha* argPilha)
{
	return (argPilha->LinkTopo == NULL);
}

void liberar(Pilha* argPilha)
{
	Lista* q = argPilha->LinkTopo;
	while(q != NULL)
	{
		Lista* t = q->Link;
		free(q);
		q = t;
	}
	free(argPilha);
}

float topo(Pilha* argPilha)
{
	if(vazia(argPilha))
```

```
        {
                printf("Pilha vazia\n");
                exit(1);
        }
        return argPilha->LinkTopo->info;
}
```

file: phila01.h

```
#include <stdio.h>
#include <stdlib.h>
#include "pilha01.h"

struct lista
{
        float info;
        struct lista* Link;
};

//Representada como uma lista siempre apuntado para o ultimo elemento
struct pilha
{
        Lista* LinkTopo;
};

Pilha* criar_pilha()
{
        Pilha* argPilha = (Pilha*)malloc(sizeof(Pilha));
        argPilha->LinkTopo = NULL;
        return argPilha;
}

void push(Pilha* argPilha, float Valor)
{
        Lista* novoElemento = (Lista*)malloc(sizeof(Lista));
        novoElemento->info = Valor;
        novoElemento->Link = argPilha-> LinkTopo;
        argPilha->LinkTopo = novoElemento;
}

float pop(Pilha* argPilha) //Retorna o elemento que foi removido
{
        Lista* t;
        float Valor;
        if(vazia(argPilha))
        {
                printf("Pilha vazia\n");
                exit(1);
        }
        t = argPilha->LinkTopo;
        Valor = t->info;
        argPilha->LinkTopo = t->Link;
        free(t);
        return Valor;
}

int vazia(Pilha* argPilha)
{
        return (argPilha->LinkTopo == NULL);
}

void liberar(Pilha* argPilha)
{
        Lista* q = argPilha->LinkTopo;
        while(q != NULL)
        {
                Lista* t = q->Link;
                free(q);
                q = t;
```

```
        }
        free(argPilha);
}

float topo(Pilha* argPilha)
{
        if(vazia(argPilha))
        {
                printf("Pilha vazia\n");
                exit(1);
        }
        return argPilha->LinkTopo->info;
}
```

file: main.c

```
#include <stdio.h>
#include "pilha01.h"

int main(int argc, char *argv[])
{
        Pilha* p;

        p = criar_pilha();
        push(p, 10);
        push(p, 20);
        push(p, 30);
        push(p, 1.5);
        push(p, 40);
        printf("Topo: %.2f\n", topo(p));
        pop(p);
        printf("Topo: %.2f\n", topo(p));
        pop(p);
        printf("Topo: %.2f\n", topo(p));
        pop(p);
        printf("Topo: %.2f\n", topo(p));
        pop(p);
        printf("Topo: %.2f\n", topo(p));
        pop(p);
        printf("Topo: %.2f\n", topo(p));
        return 0;
}
```