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

struct pilhaNo {
    int data;
    struct pilhaNo *proximoPonteiro;
};

typedef struct pilhaNo PilhaNo;
typedef PilhaNo *PilhaNoPonteiro;

void push (PilhaNoPonteiro *topoPonteiro, int info);
int pop (PilhaNoPonteiro *topoPonteiro);
int estaVazia (PilhaNoPonteiro topoPonteiro);
void imprimirPilha (PilhaNoPonteiro atualPonteiro);
void instrucoes (void);

int main (void) {
    PilhaNoPonteiro pilhaPonteiro = NULL; //aponta para o topo da pilha
    int escolha; //escolha do menu do usuário
    int valor; //entrada int dada pelo usuário

    instrucoes (); //exibe o menu
    printf ("?\n");
    scanf ("%d", &escolha);

    //enquanto o usuário não digitar 4
    while (escolha != 4) {
        switch (escolha) {
            case 1: //coloca valor na pilha
                printf("Digite um valor: ");
                scanf ("%d", &valor);
                push (&pilhaPonteiro, valor);
                break;
            case 2: //remove valor da pilha
                //se a pilha não está vazia
                if (!estaVazia(pilhaPonteiro)) {
                    printf("O valor %d foi retirado.\n", pop(&pilhaPonteiro));
                }
                break;
            case 3: //imprime a pilha
                imprimirPilha (pilhaPonteiro);
                break;
            default:
                printf ("Escolha invalida.\n\n");
                instrucoes();
                break;
        }
        printf ("?\n");
        scanf ("%d", &escolha);
    }
    printf ("Fim da execucao!\n\n");
    return 0;
}

void instrucoes (void) {
    printf ("Digite sua escolha:\n"
        "   1 para colocar um valor na pilha.\n"
        "   2 para retirar um valor da pilha.\n"
        "   3 para imprimir a pilha.\n"
        "   4 para terminar o programa\n");
}
```

```c
void push (PilhaNoPonteiro *topoPonteiro, int info) {
    PilhaNoPonteiro novoPonteiro; //ponteiro para novo nó

    novoPonteiro = malloc (sizeof (PilhaNo));

    //insere o nó no topo da pilha
    if (novoPonteiro != NULL) {
        novoPonteiro -> data = info;
        novoPonteiro -> proximoPonteiro = *topoPonteiro;
        *topoPonteiro = novoPonteiro;
    }
    else {
        printf ("%d nao inserido. Sem memoria disponivel!\n", info);
    }
}

int pop (PilhaNoPonteiro *topoPonteiro) {
    PilhaNoPonteiro tempPtr; //ponteiro de nó temporário
    int popValue;

    tempPtr = *topoPonteiro;
    popValue = (*topoPonteiro) -> data;
    *topoPonteiro = (*topoPonteiro) -> proximoPonteiro;
    free (tempPtr);

    return popValue;
}

void imprimirPilha (PilhaNoPonteiro atualPonteiro) {
    //se a pilha está vazia
    if (atualPonteiro == NULL) {
        printf ("A pilha esta vazia!\n\n");
    }
    else {
        printf ("A pilha e:\n");
        //enquanto não chegar no final da pilha
        while (atualPonteiro != NULL) {
            printf ("%d --> ", atualPonteiro->data);
            atualPonteiro = atualPonteiro->proximoPonteiro;
        }
        printf ("NULL\n\n");
    }
}

int estaVazia (PilhaNoPonteiro topoPonteiro) {
    return topoPonteiro == NULL;
}

```