• Definição da estrutura do 'Nó' da Pilha:

```
typedef struct Node{
int data;
struct Node* next;
}Node;
```

• Definição da estrutura da Pilha:

```
typedef struct{
Node* top;
}Stack;
```

• Função para criar uma nova Pilha Vazia:

```
Stack* createStack(){
Stack* stack =
(Stack*)malloc(sizeof(Stack));
IF (stack==NULL){
Printf("Erro: Falha na alocação
de memoria para a Pilha\n");
exit(EXIT_FAILURE);}
stack->top=NULL;
Return stack}
```

• Função para verificar se a Pilha esta Vazia:

```
int isEmpty(stack* stack){
    return stack->top==NULL;
}
```

• Função para empilhar um elemento na pilha:

```
Void Push(Stack *stack, int data){
  Node* newNode =
(Node*)malloc(sizeof(Node));
  IF(newNode == NULL){
    Printf("Erro: Falha na alocação de memória
          para o nó da pilha.\n");
    exit(EXIT_FAILURE);
  } newNode -> data = data;
    newNode -> next = stack -> Top;
    stack -> Top = newNode;}
```

• Função para desempilhar um elemento da pilha:

```
int Pop(Stack *stack){
  IF(isEmpty(stack)){
    Printf("Erro: A Pilha está vazia.\n");
    exit(EXIT_FAILURE);}
  Node * Temp = stack -> Top;
  int data = Temp -> Data;
  stack -> Top = Temp -> next;
  return data;}
```

• Função para busca um elemento na Pilha de forma recursiva:

```
int searchRecursive(Node * current, int target){
  IF(current == NULL){
    return 0; // Elemento não encontrado}
```

```
IF(current -> data == target){
    Return 1; //Elemento encontrado}
return searchRecursive(current ->
    next, target);}
```

• Função para buscar um elemento na Pilha:

```
int search(stack *stack, int target){
    return searchRecursive(stack -> top, target);}
```

• Função para exibir os elementos da Pilha:

```
Void display(Stack *stack){
    IF(isEmpty(stack)){
    Printf("A Pilha esta vazia\n");}

Node *current = stack -> top;
Printf("Elementos da Pilha:");
While(current != NULL){
Printf("%d", current -> data);
current = current -> next;}
Printf("\n");}
```

• Função Principal:

```
int main(){
    stack *stack = createStack();

    Push(stack, 10);
    Push(stack, 20);
    Push(stack, 30);
    Push(stack, 40);
```

```
DisPlay(&stek);
    int target=20;
if(search(&stek,target)){
Printf("O elemento %d esta na Pilha\n", target
);}else{
    Printf("O elemento %d não esta na Pilha
    .\n", target);}

Printf("Desempilhando um elemento: %d\n",
Pop(&stek));

DisPlay(&stek);

Free(&stek); // Liberando a memoria alocada
                para a Pilha
return 0;}
```