Anwaar Ali*

Lecture for the position of teaching fellow at ITU, Lahore

***PhD candidate at University of Cambridge, UK. Awaiting result**

Stack

Abstract data type (ADT)

Stack ADT

Stack ADT

- Prerequisite:
  - Linked list ADT and C

Stack ADT – An introduction

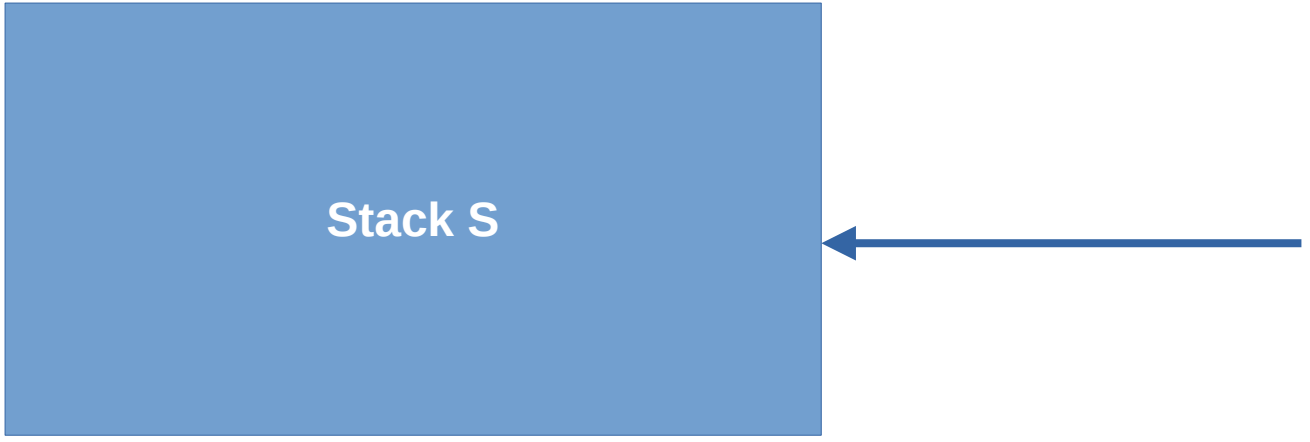
Stack ADT – An introduction

- A *last-in first-out* (LIFO) ADT

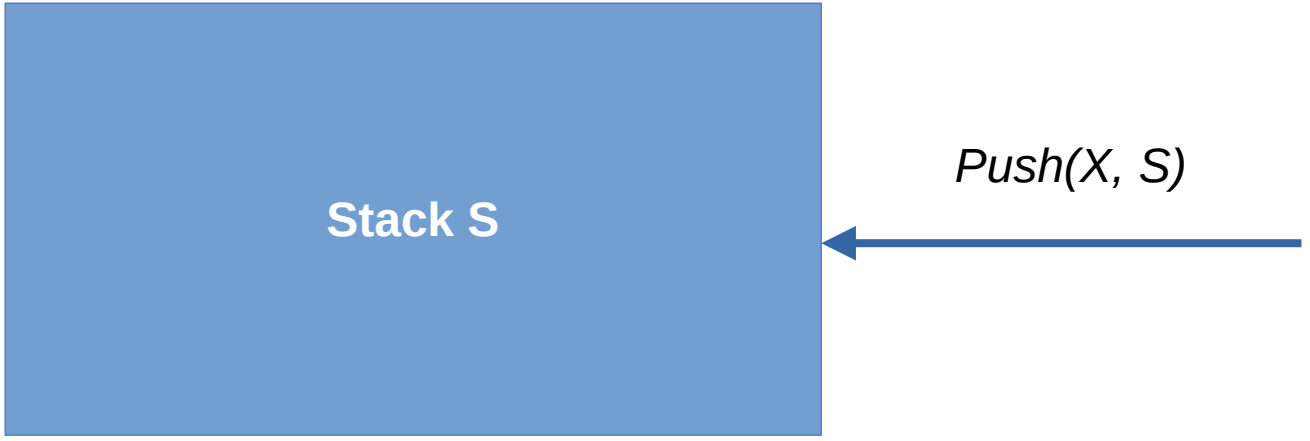
Stack ADT – Fundamental operations

Stack ADT – Fundamental operations

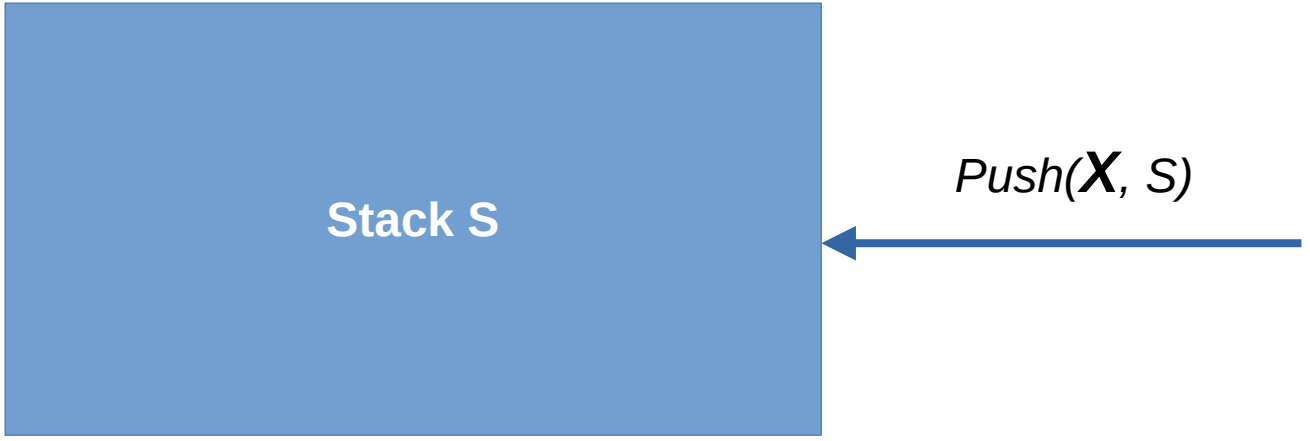
Stack ADT – Fundamental operations



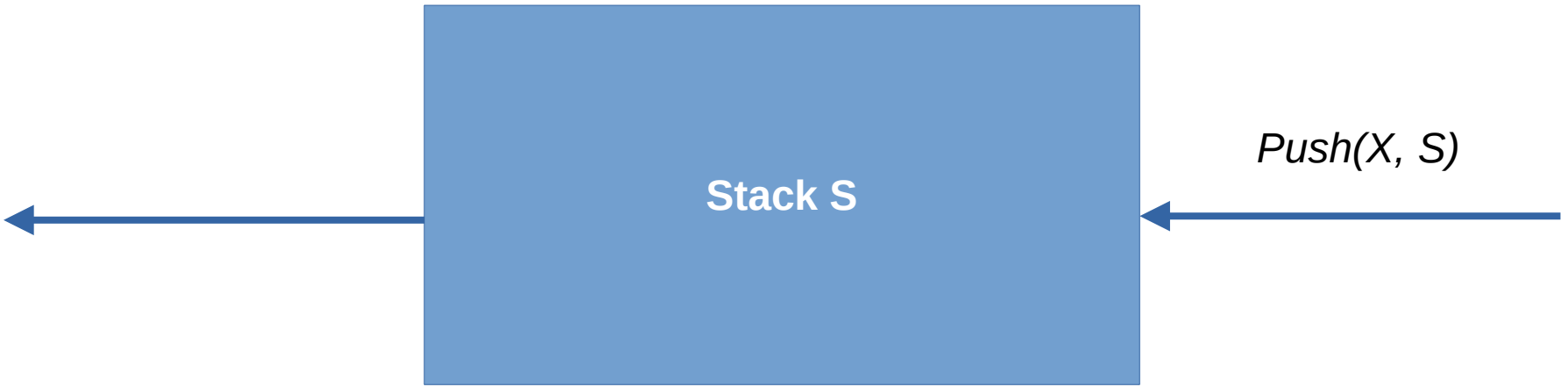
Stack ADT – Fundamental operations



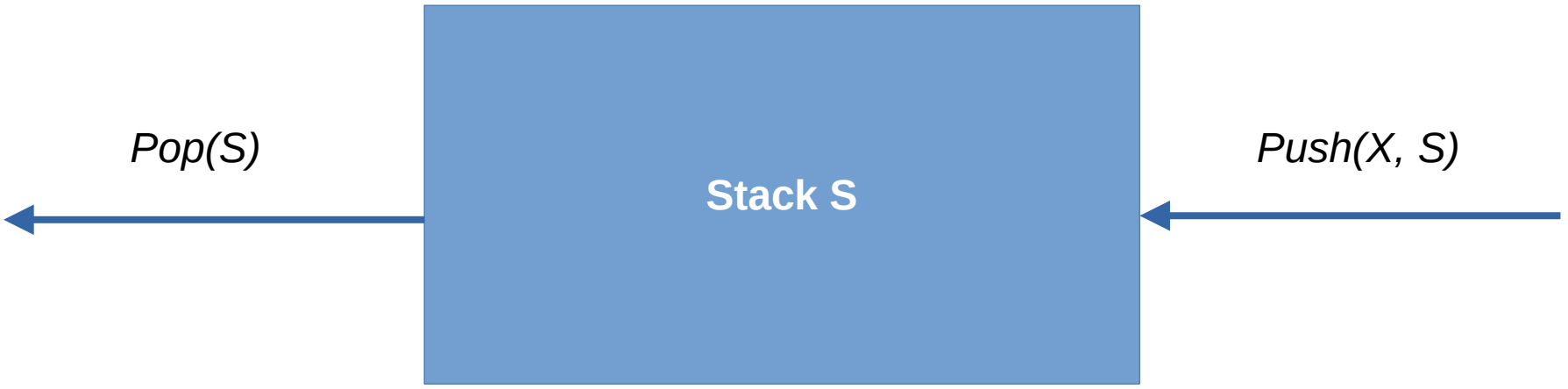
Stack ADT – Fundamental operations



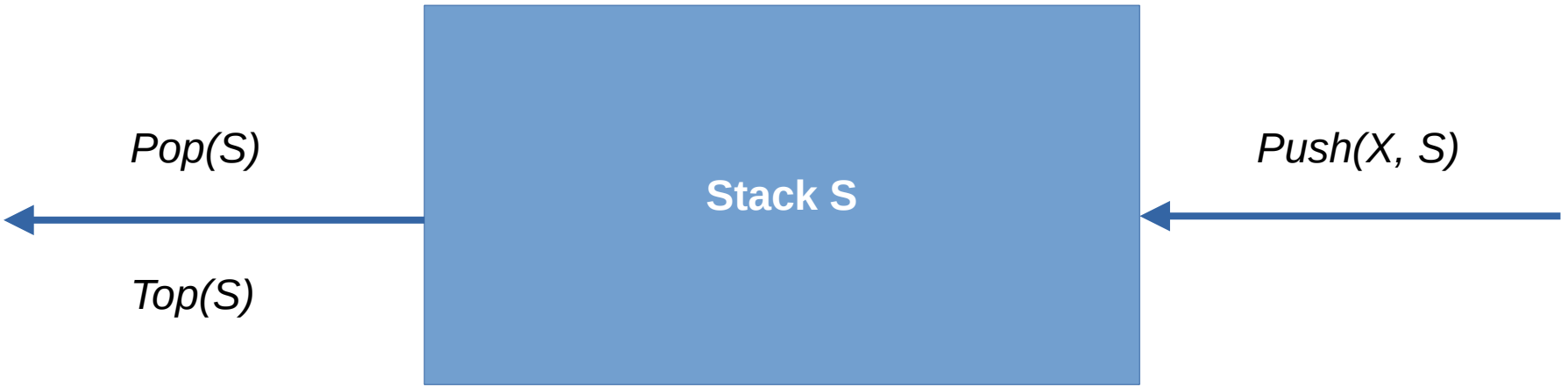
Stack ADT – Fundamental operations



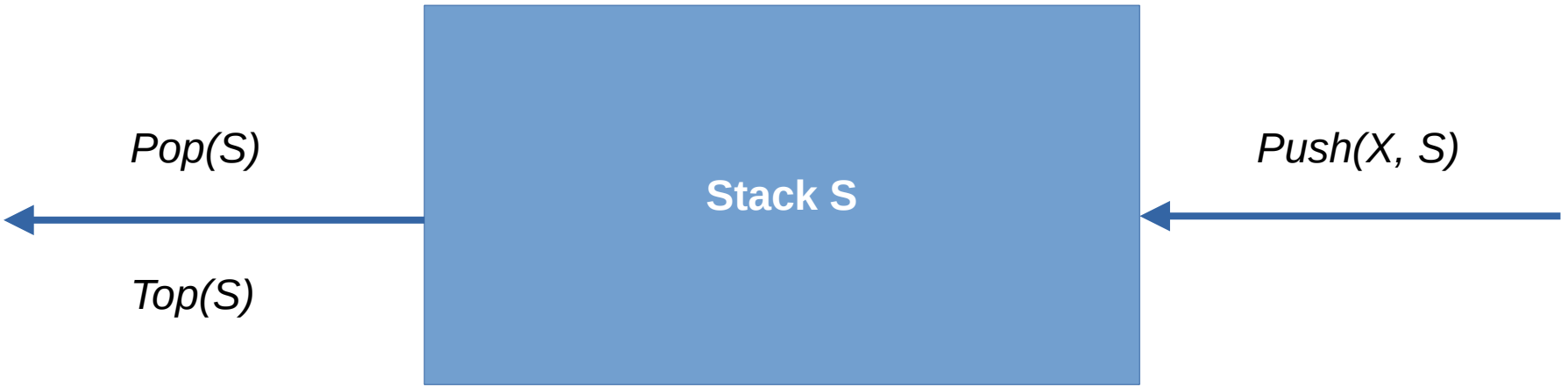
Stack ADT – Fundamental operations



Stack ADT – Fundamental operations



Stack ADT – Fundamental operations



Important: *All fundamental operations are performed only at the top of the stack ADT.*

Stack ADT – Implementation approaches

Stack ADT – Implementation approaches

- (Singly-)Linked-list

Stack ADT – Implementation approaches

- (Singly-)Linked-list
- Array

Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
- Array

Stack ADT – Implementation approaches

- **(Singly-)Linked-list**

Stack ADT – Implementation approaches

- **(Singly-)Linked-list**

Important: *All fundamental operations are performed only at the top of the stack ADT.*

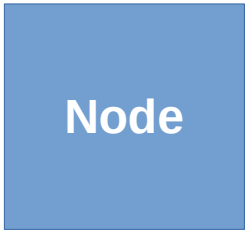
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**

Stack ADT – Implementation approaches

- **(Singly-)Linked-list**

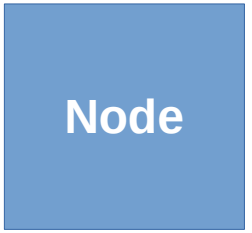
struct



Stack ADT – Implementation approaches

- **(Singly-)Linked-list**

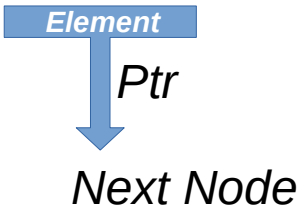
struct



malloc()

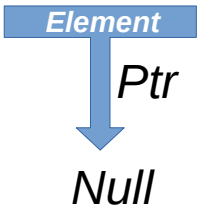
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



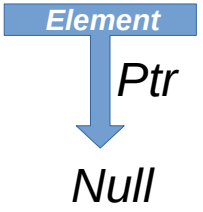
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



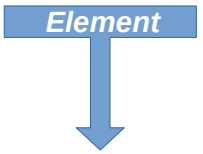
Stack ADT – Implementation approaches

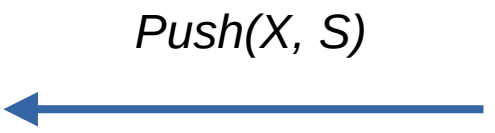
- **(Singly-)Linked-list**
  - Empty stack



Stack ADT – Implementation approaches

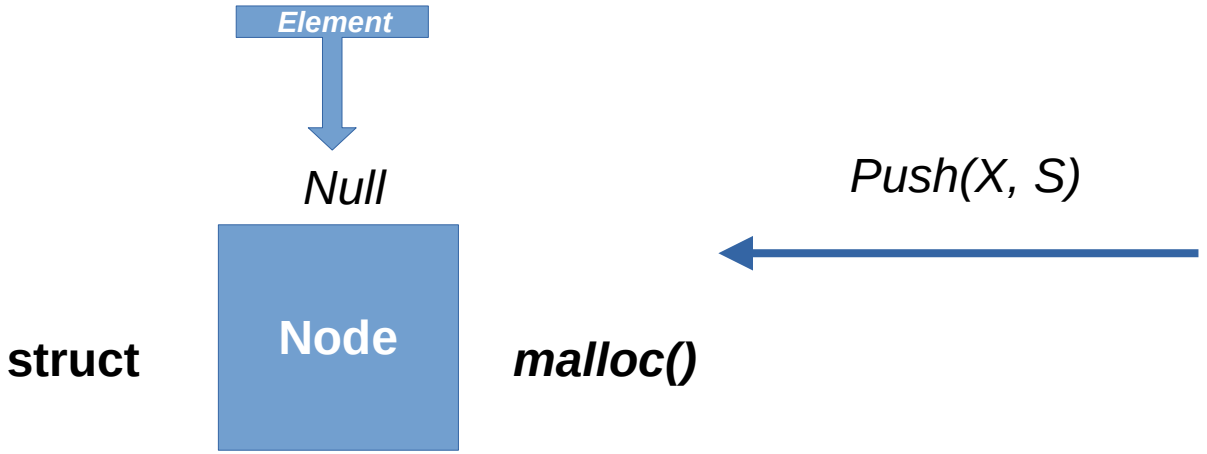
- **(Singly-)Linked-list**
  - Empty stack





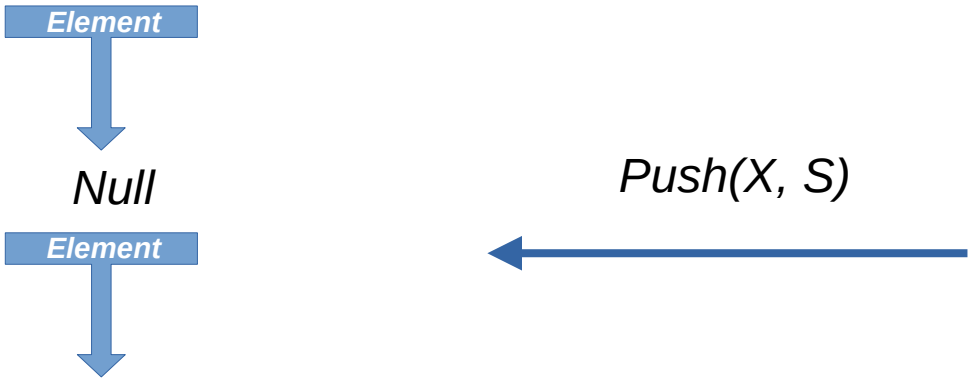
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack



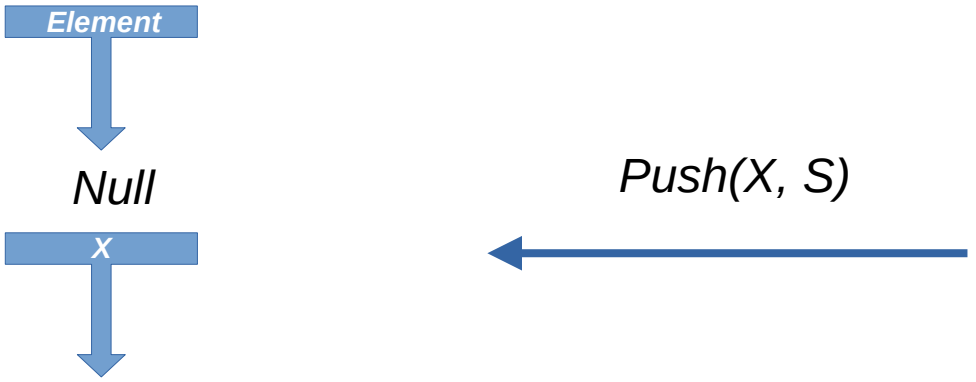
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack



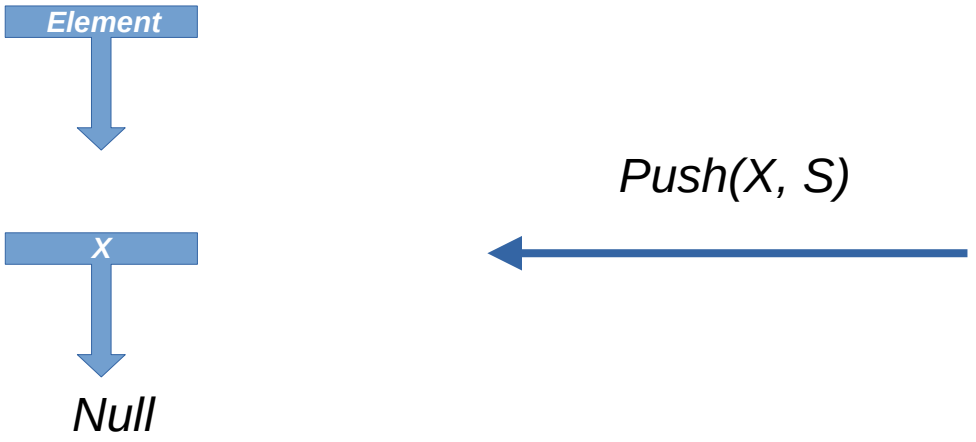
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack



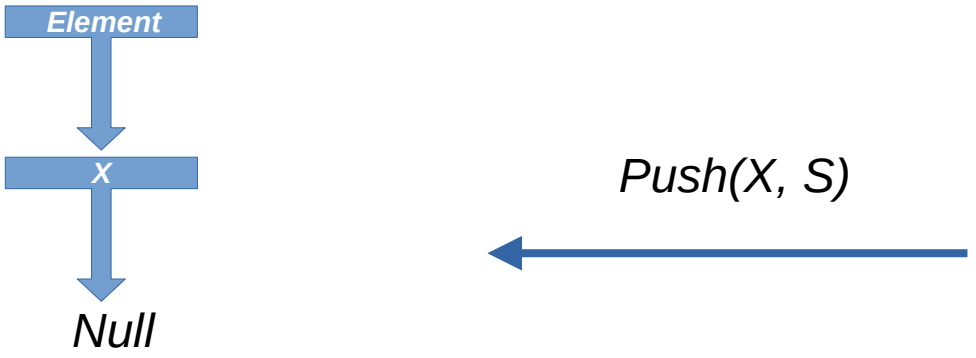
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack



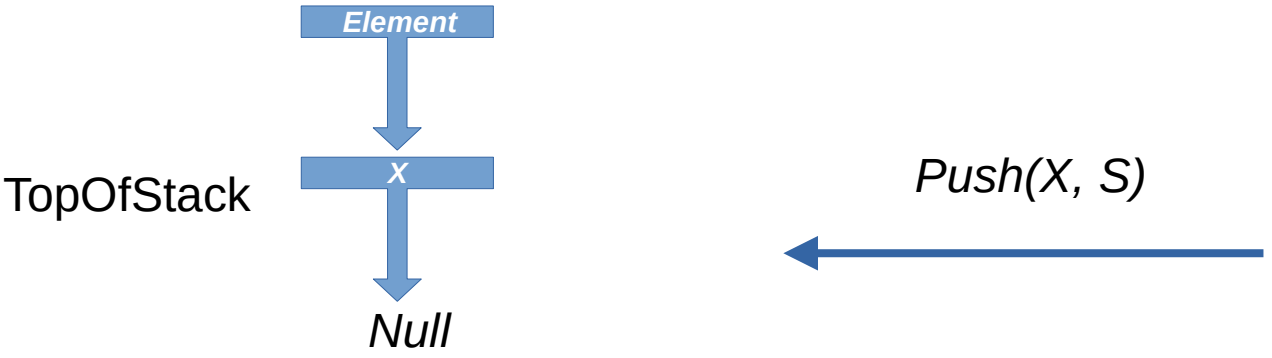
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



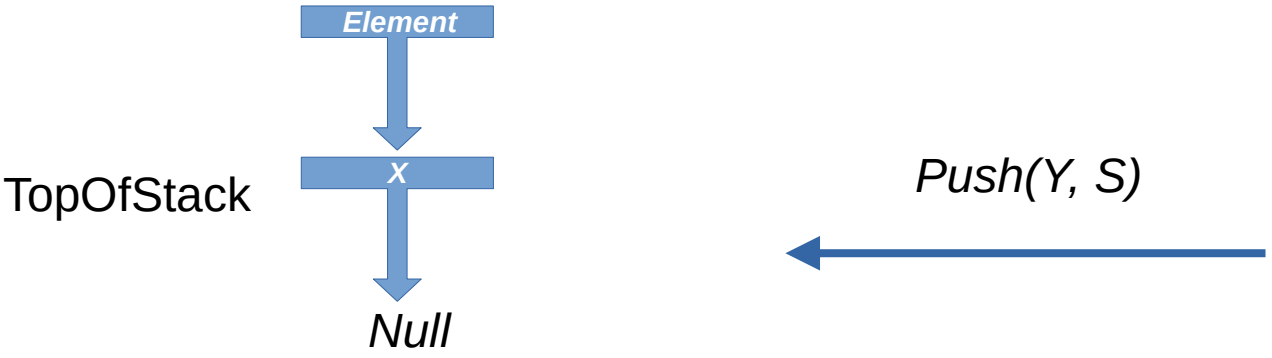
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



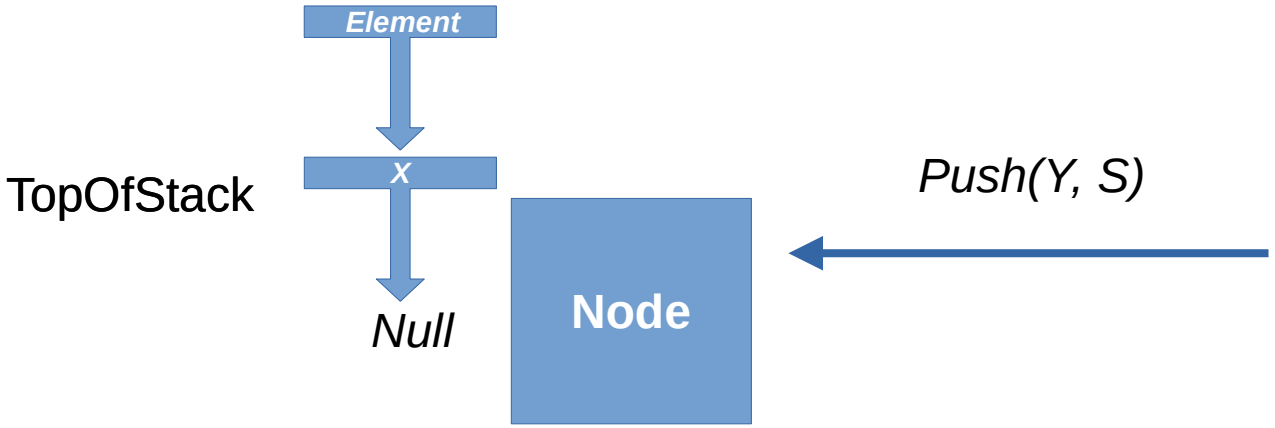
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



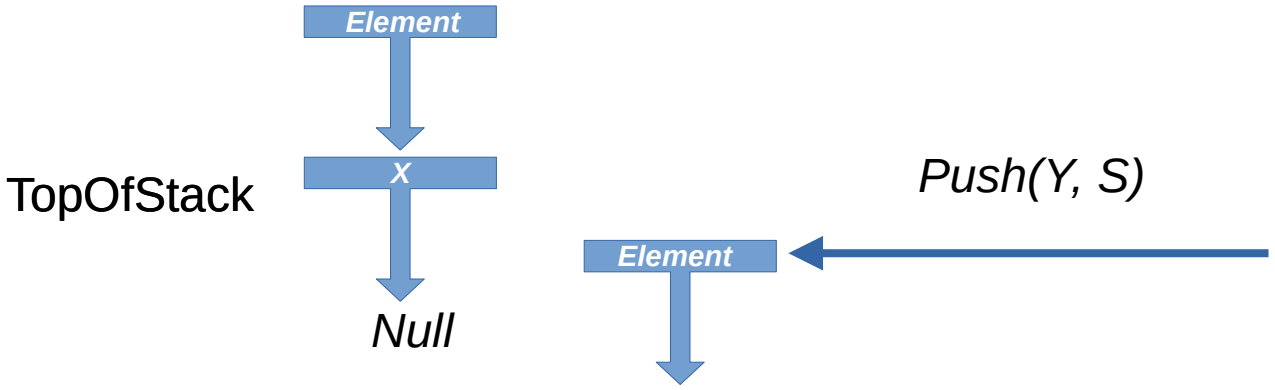
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



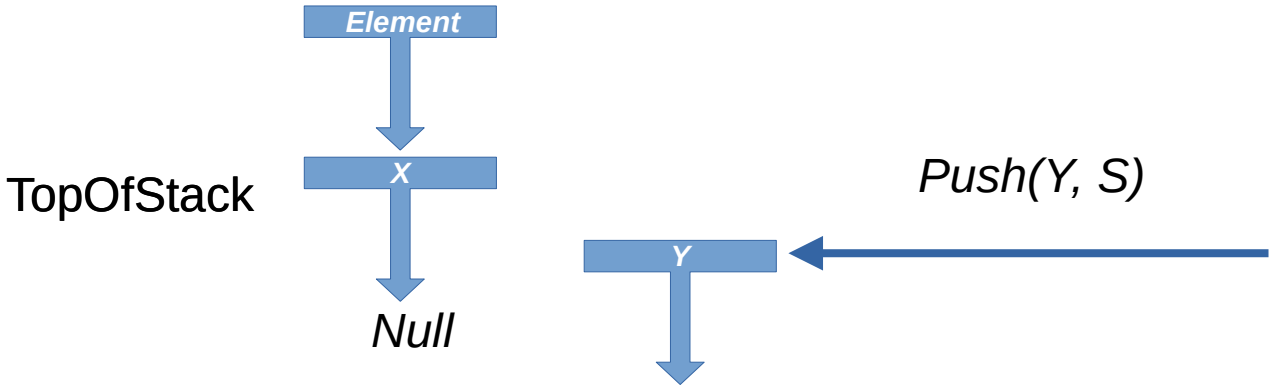
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



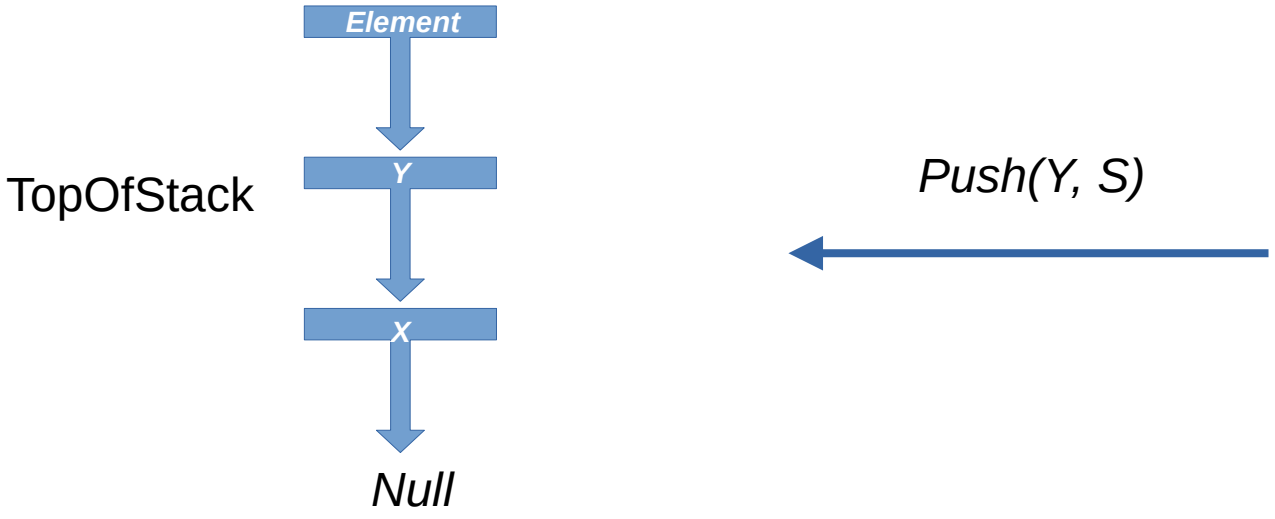
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



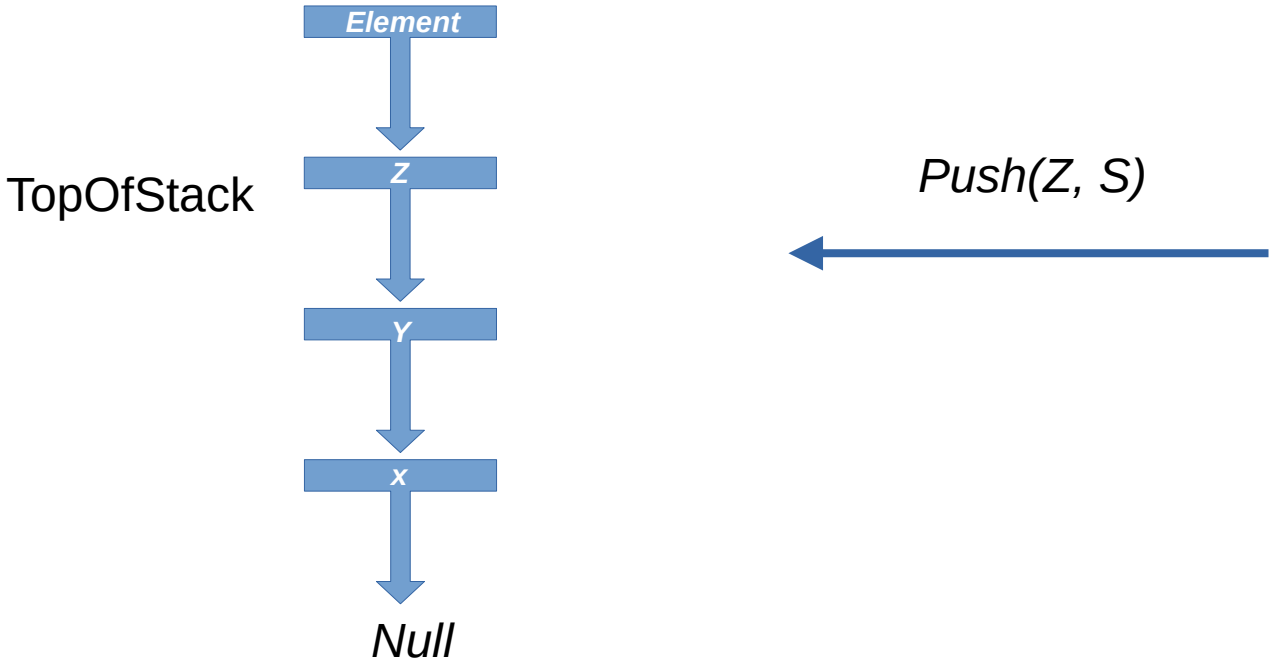
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



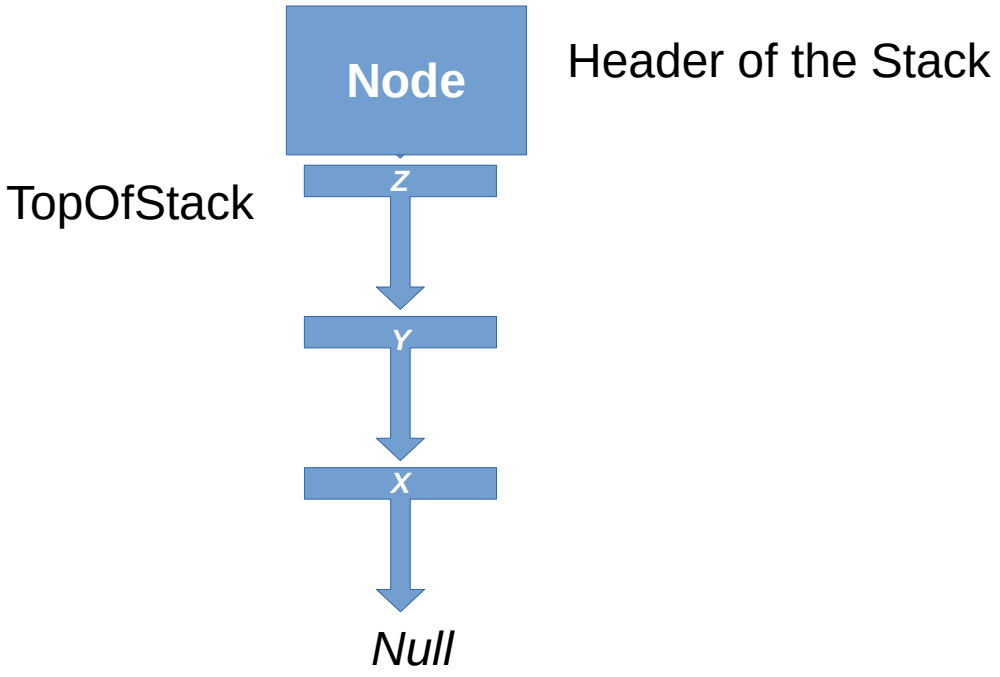
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



Stack ADT – Implementation approaches

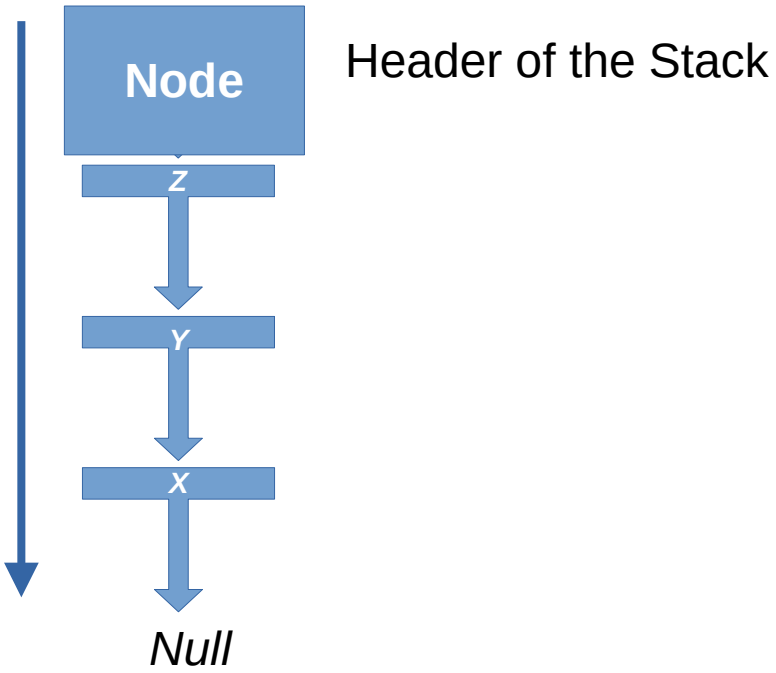
- **(Singly-)Linked-list**



Stack ADT – Implementation approaches

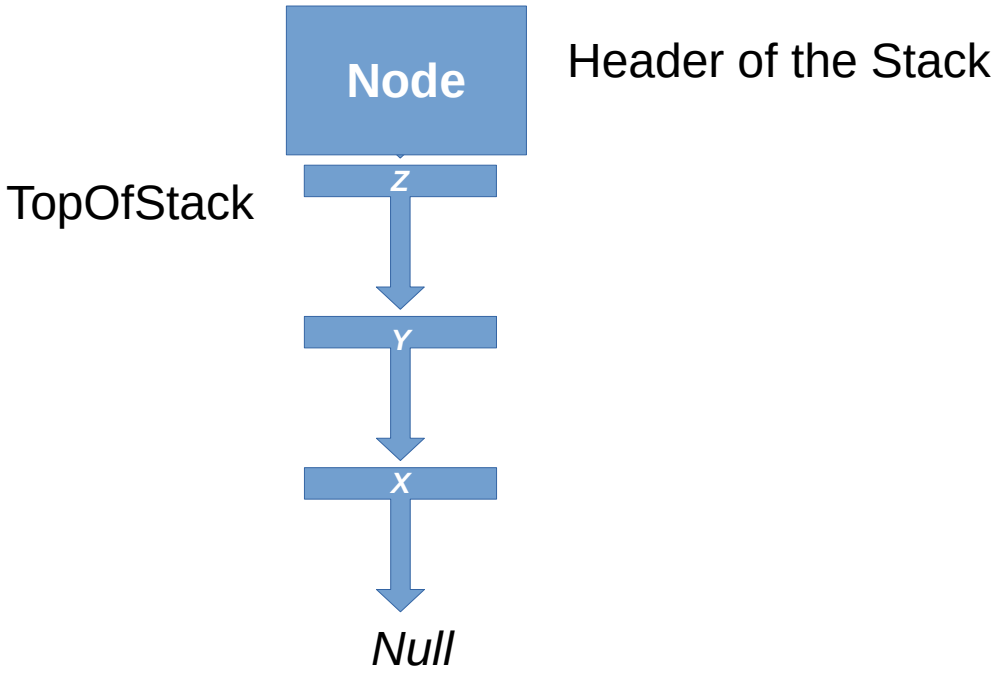
- **(Singly-)Linked-list**

Important: Grows downward with O(1)



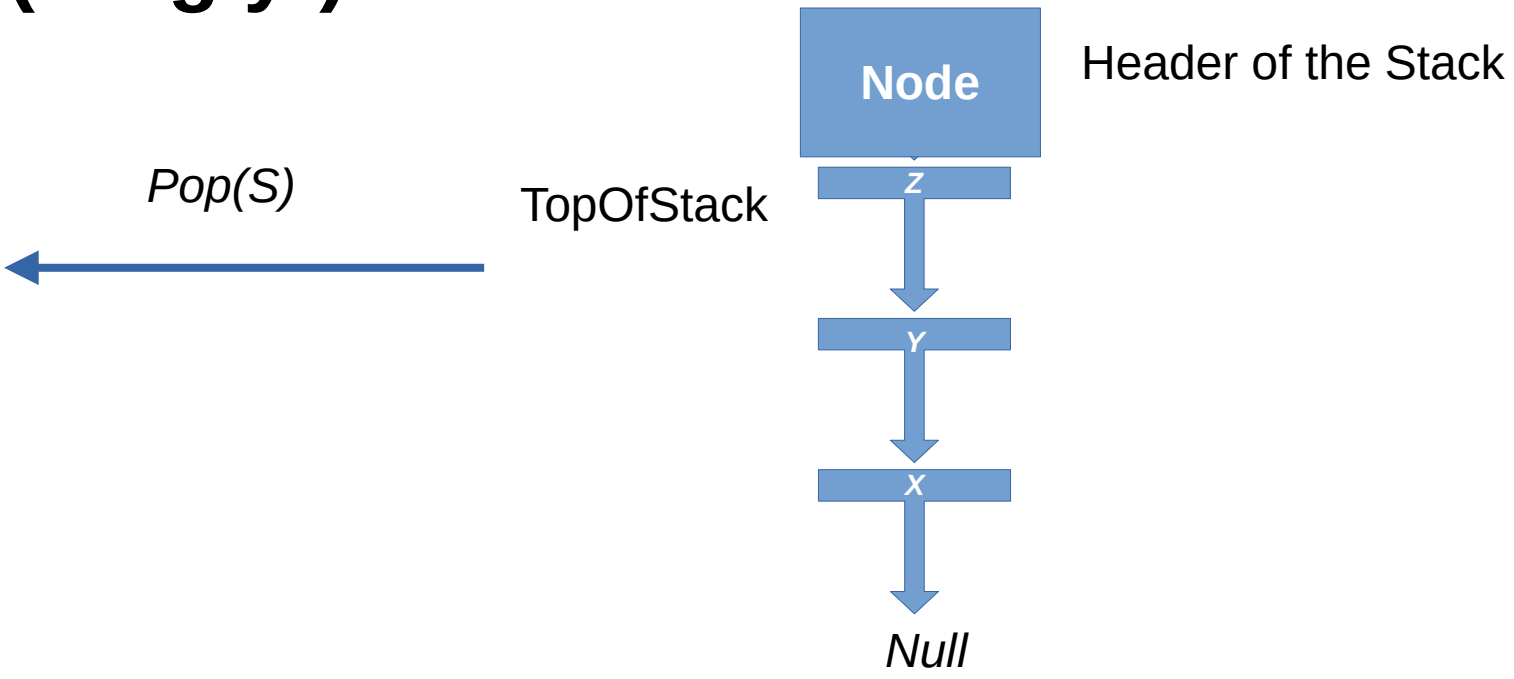
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



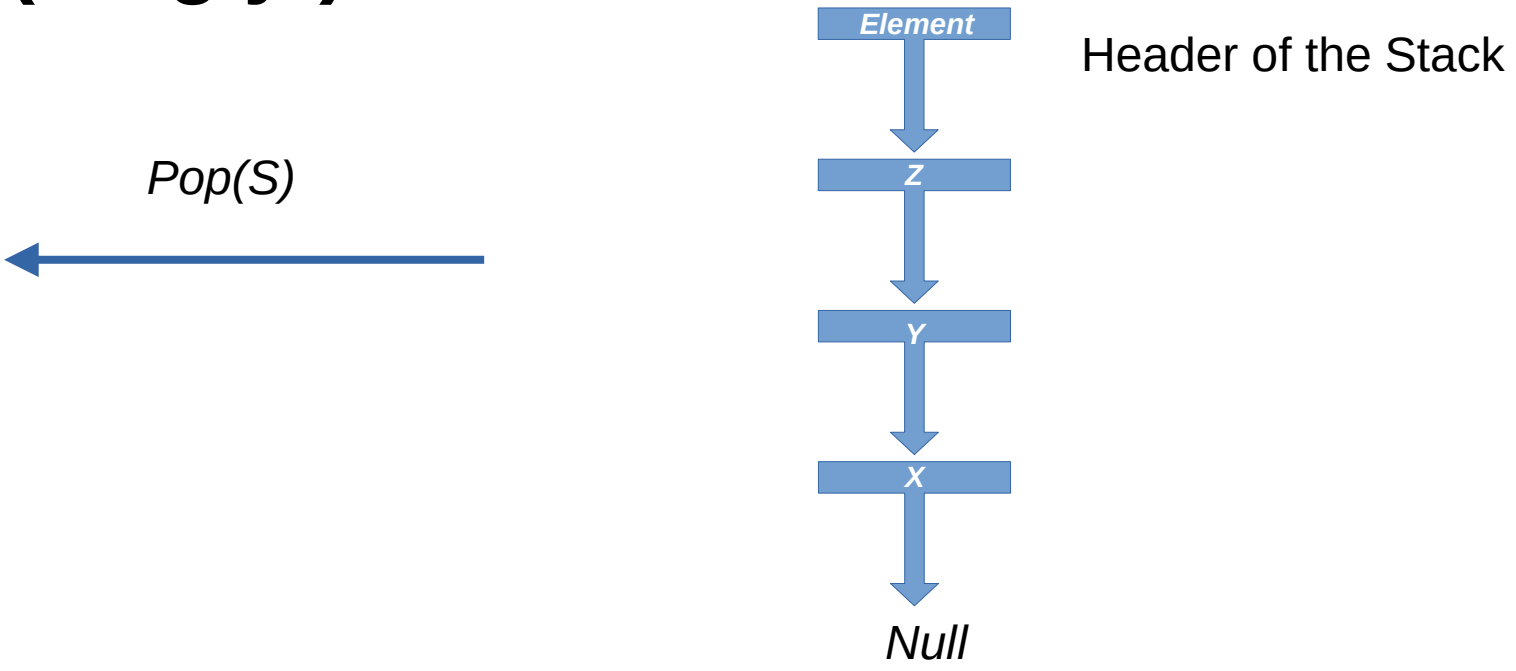
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



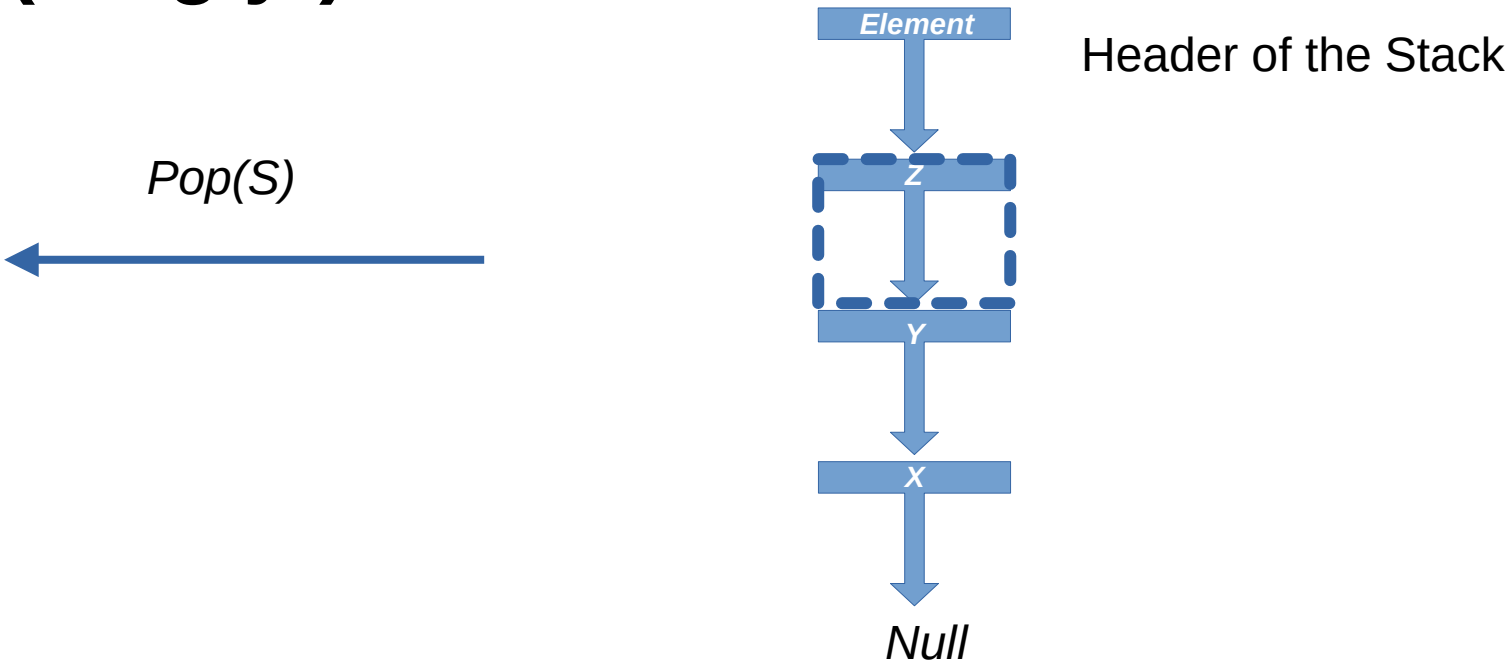
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



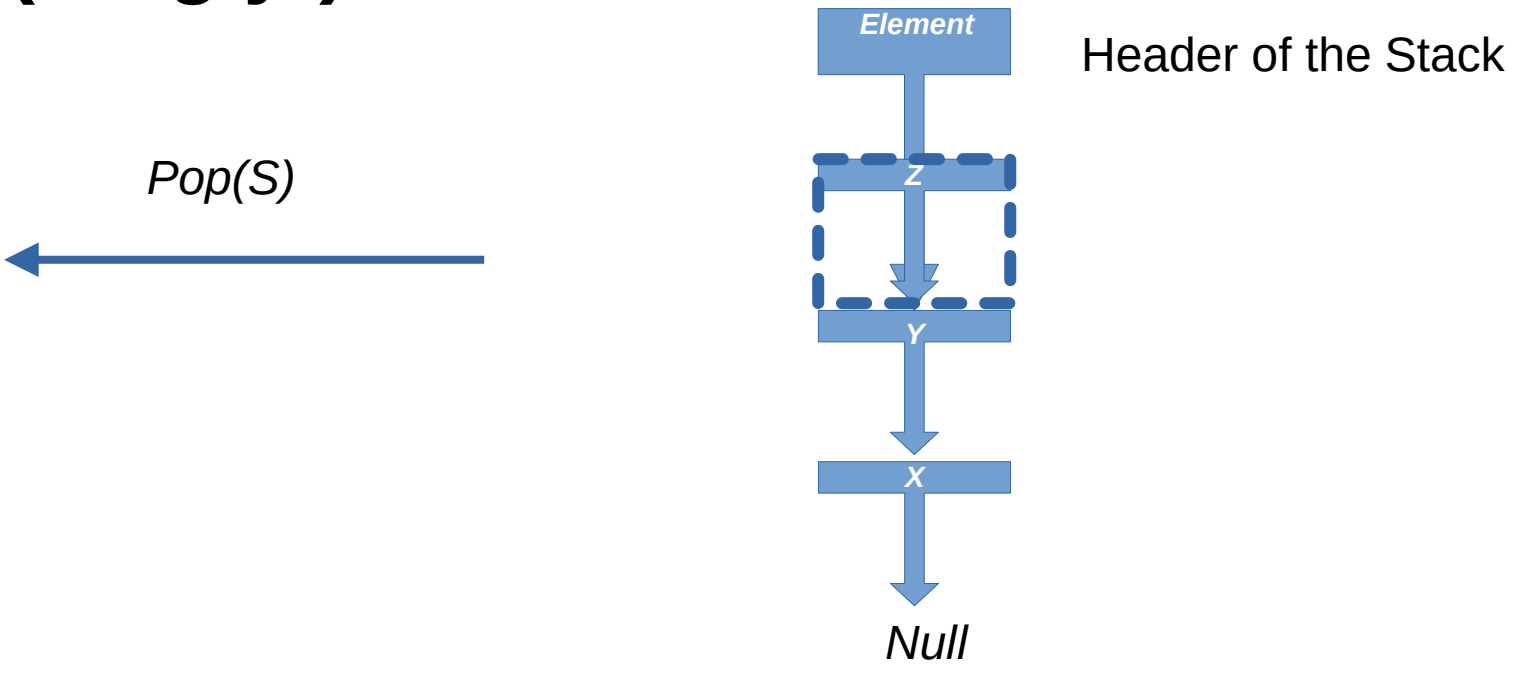
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



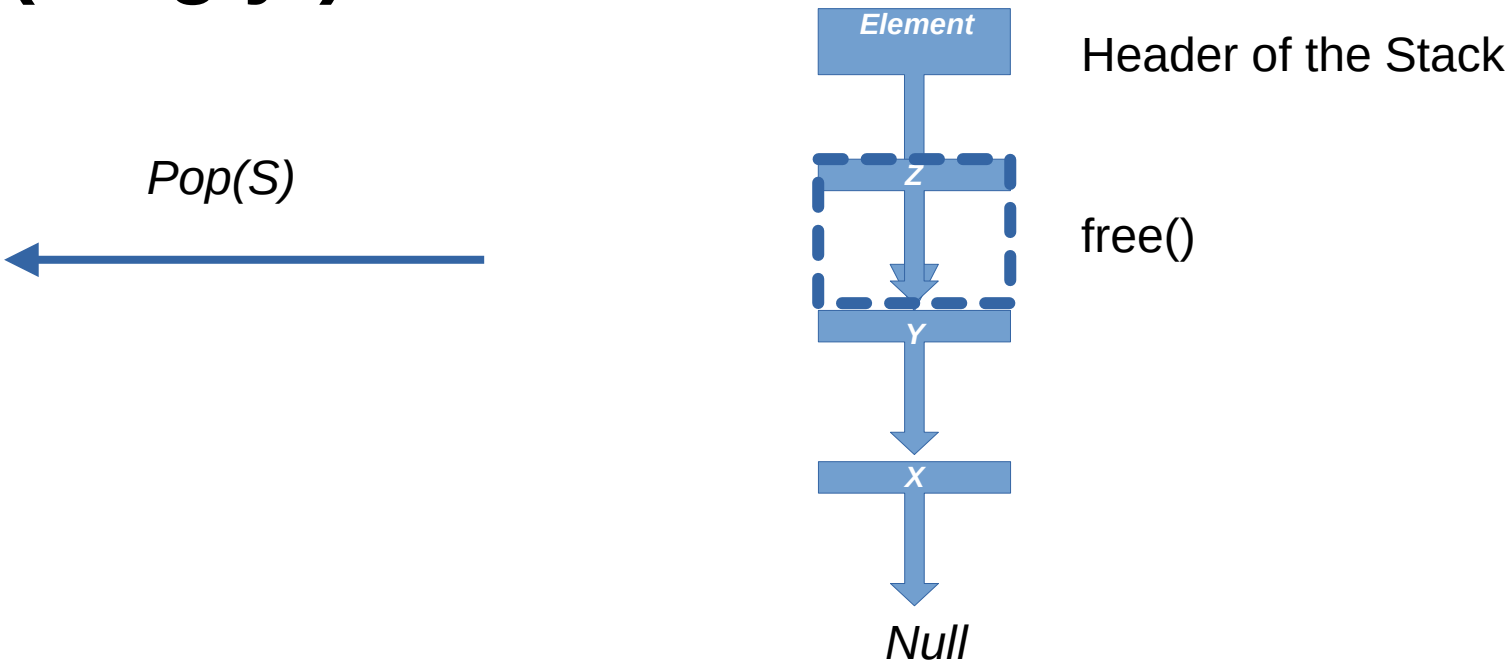
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



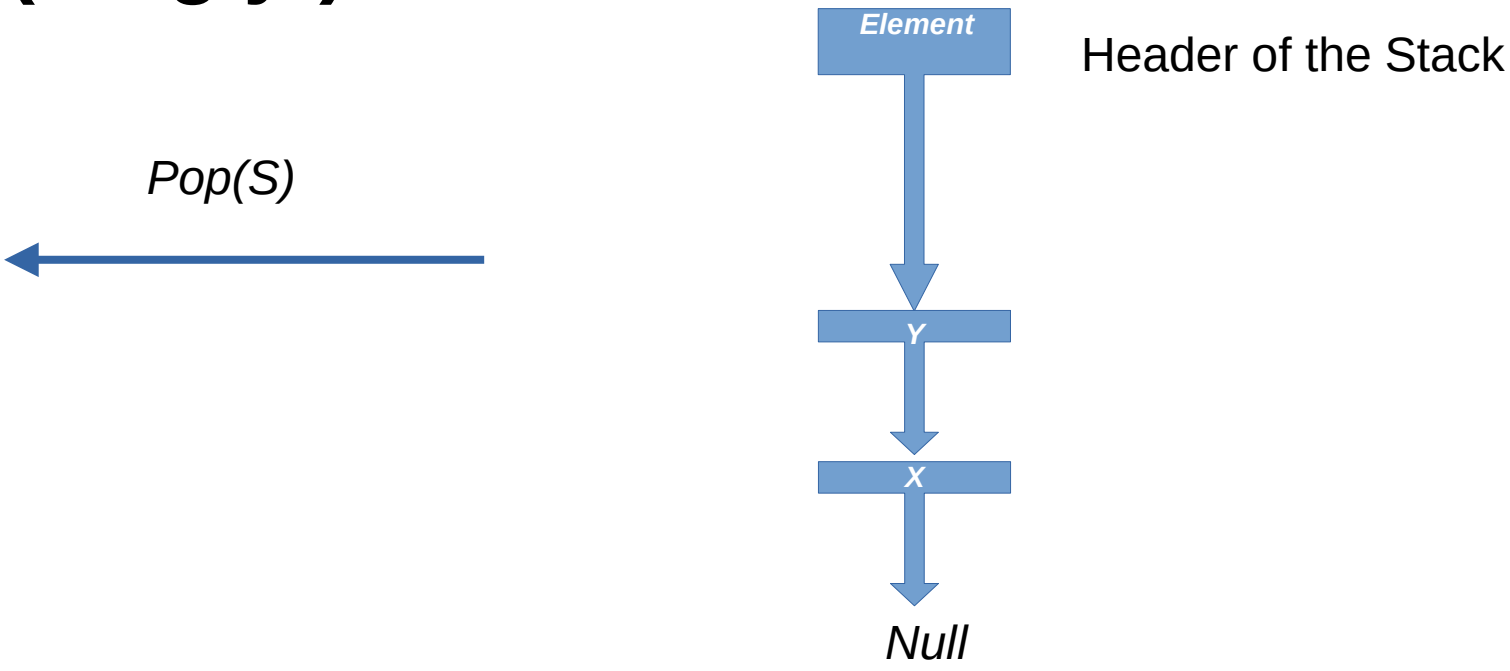
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



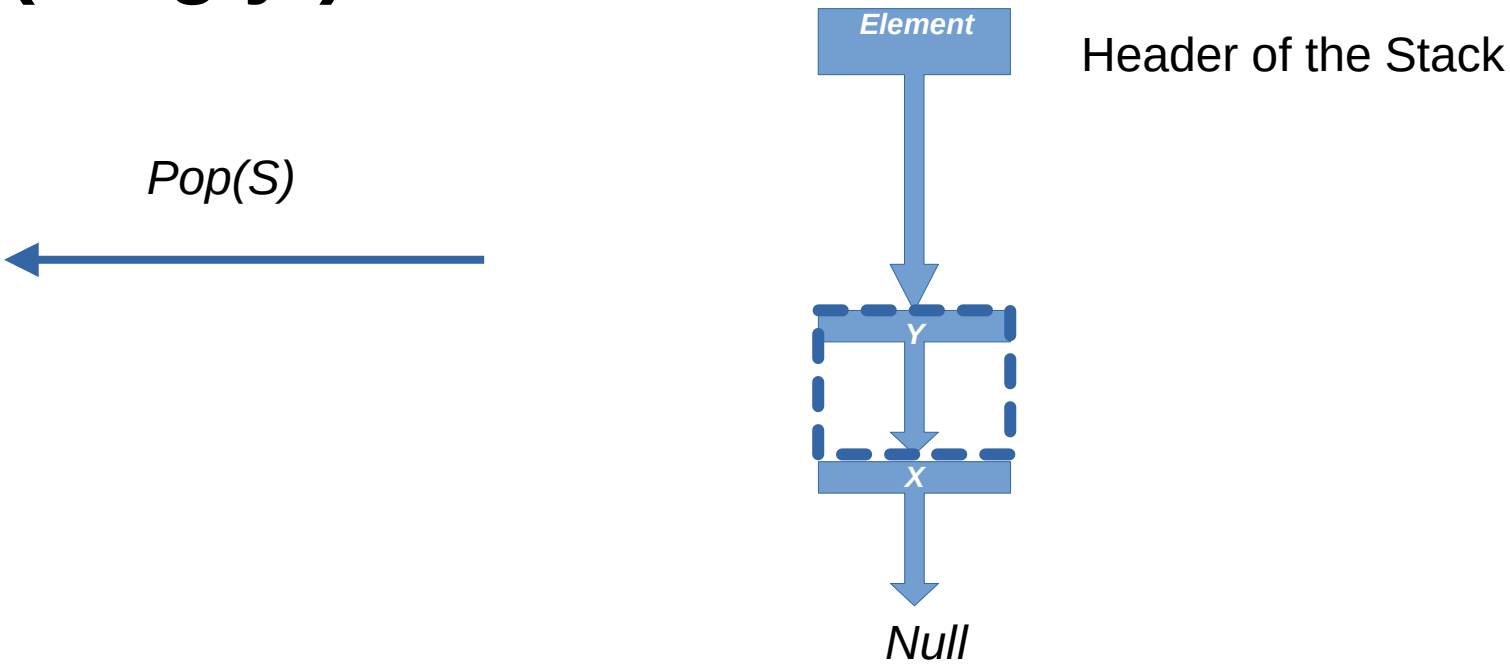
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



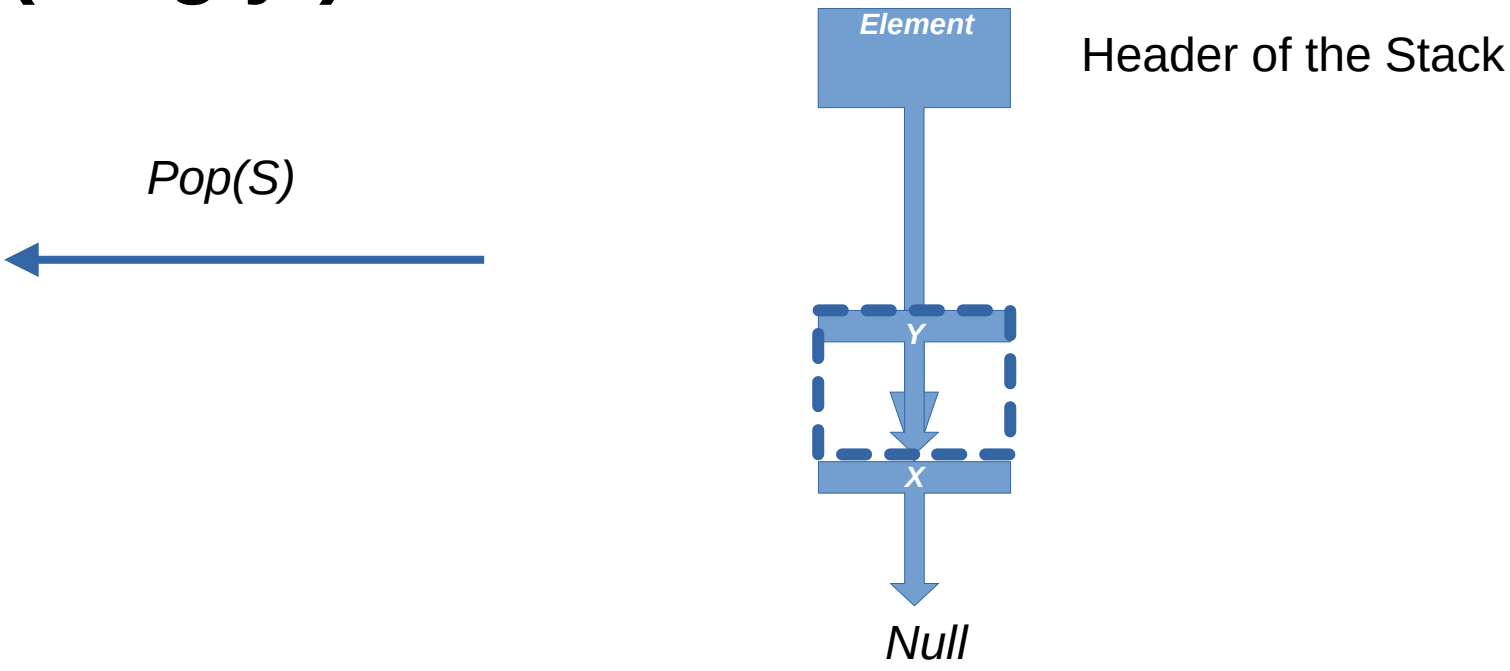
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



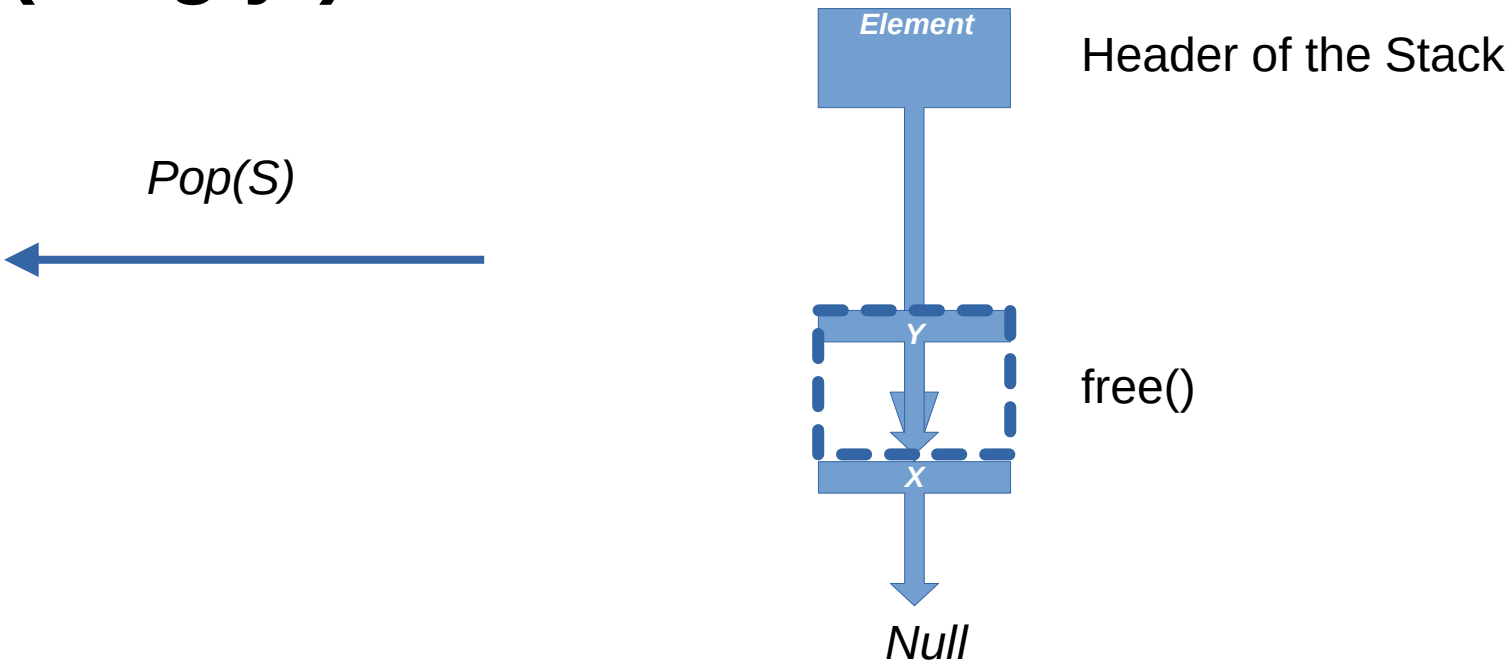
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



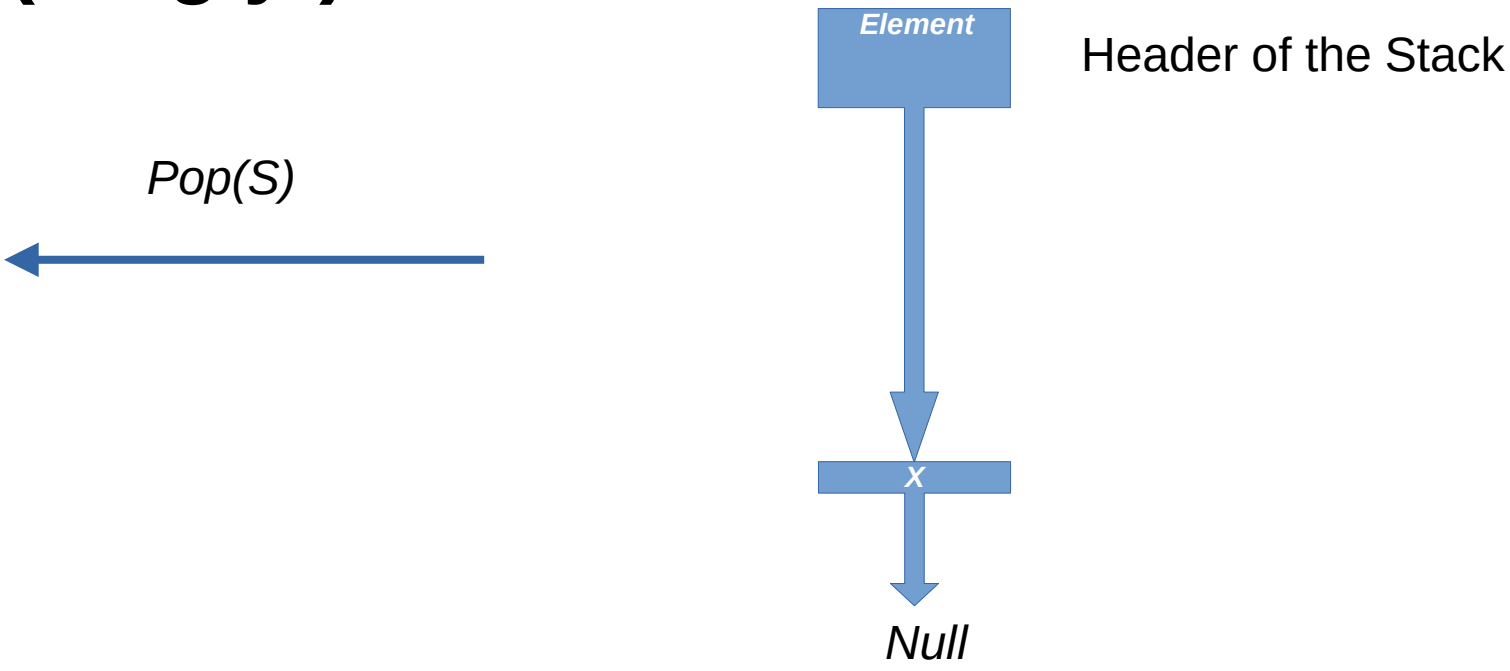
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



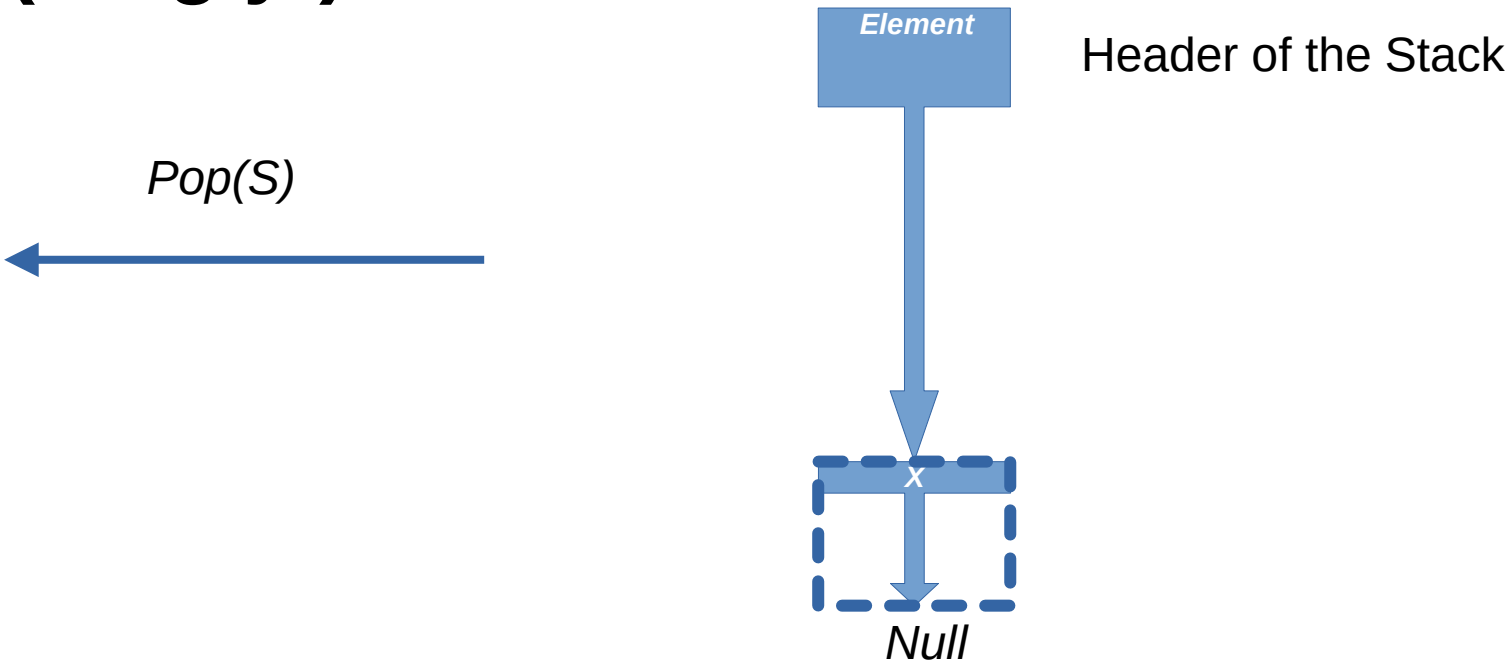
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



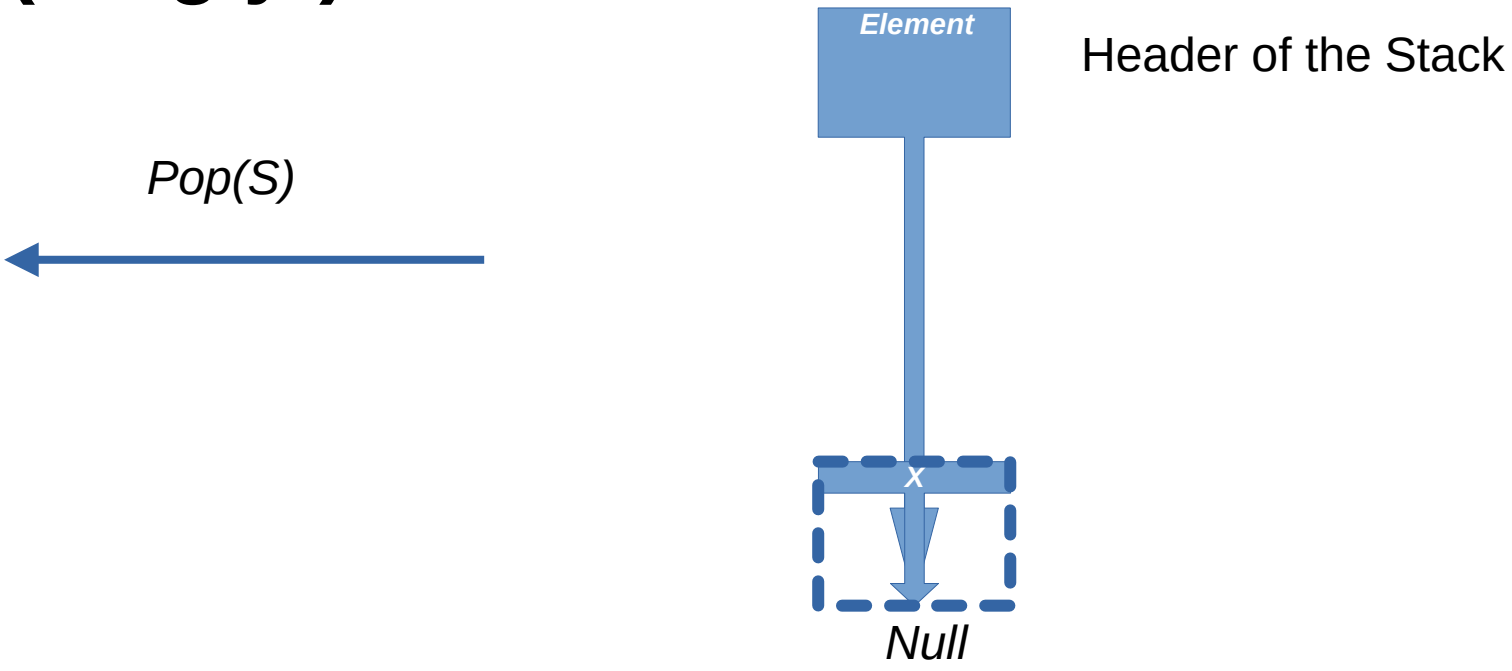
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



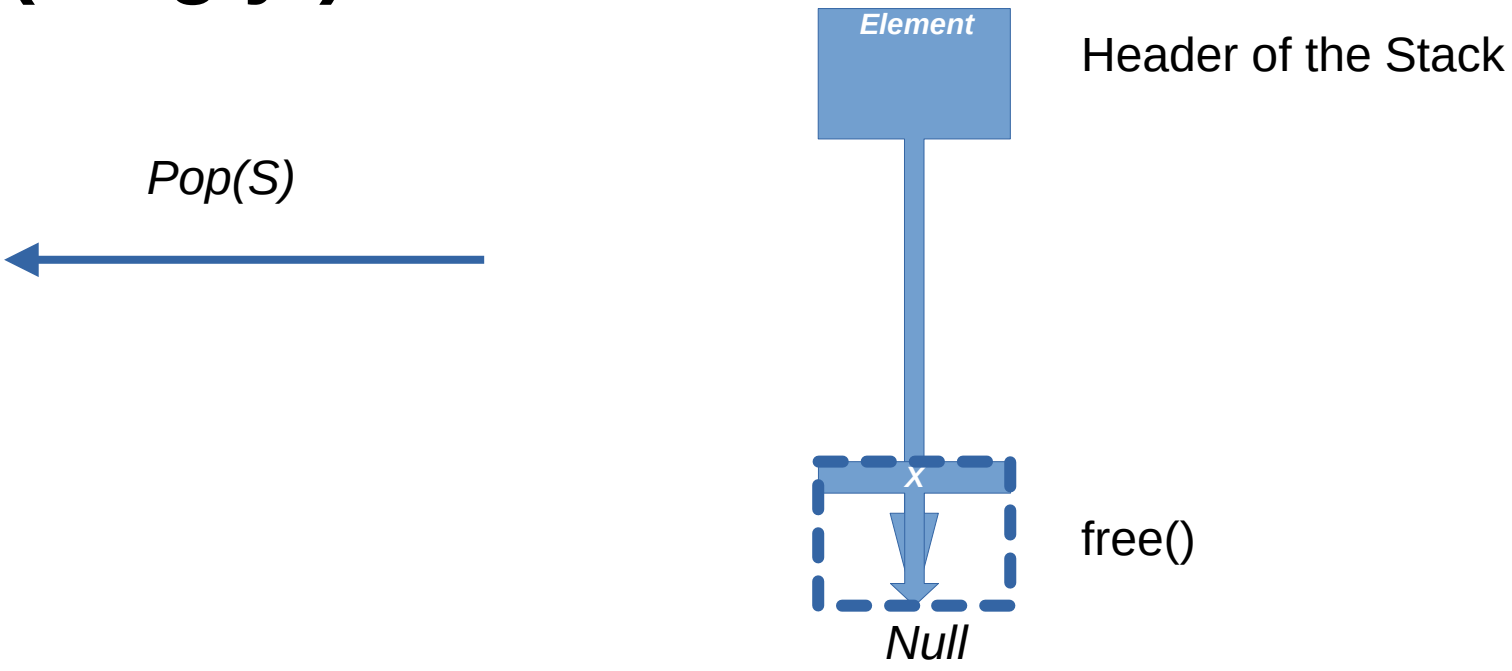
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



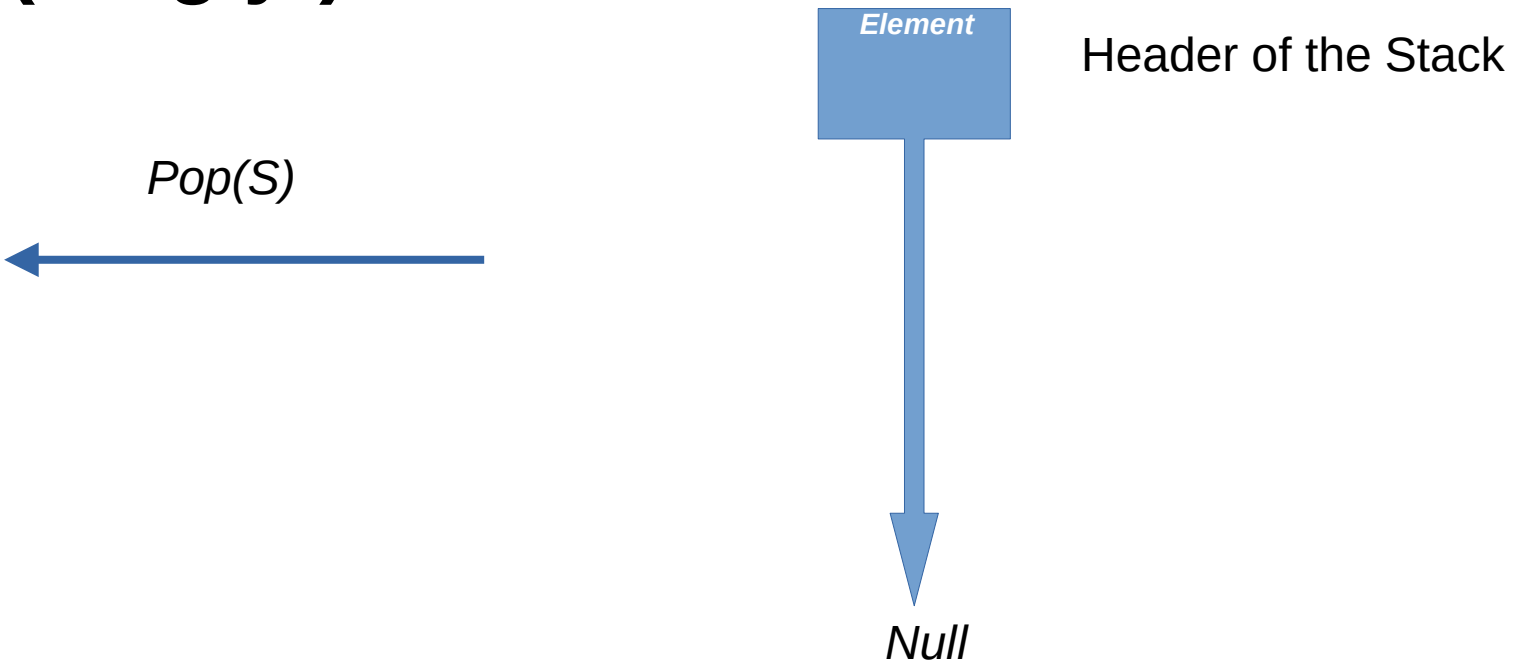
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



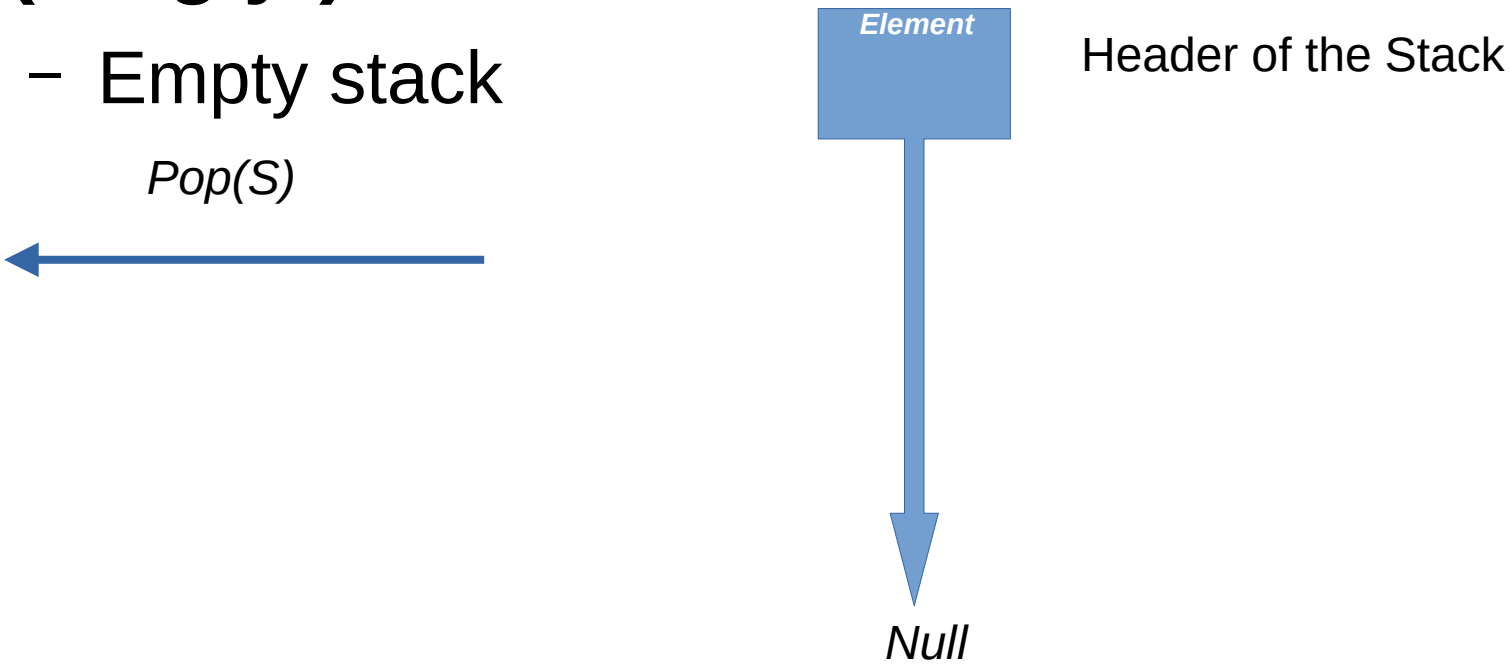
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack

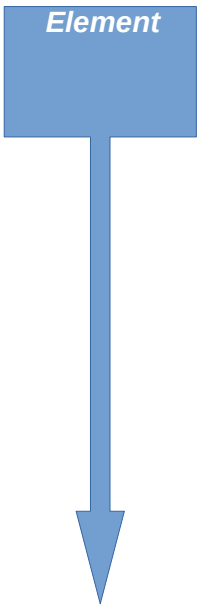


Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack

Pop(S)

Pop(S) also has O(1)



Header of the Stack

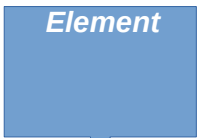
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**
  - Empty stack

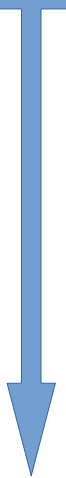
Pop(S)

Pop(S) also has O(1)

LIFO in action



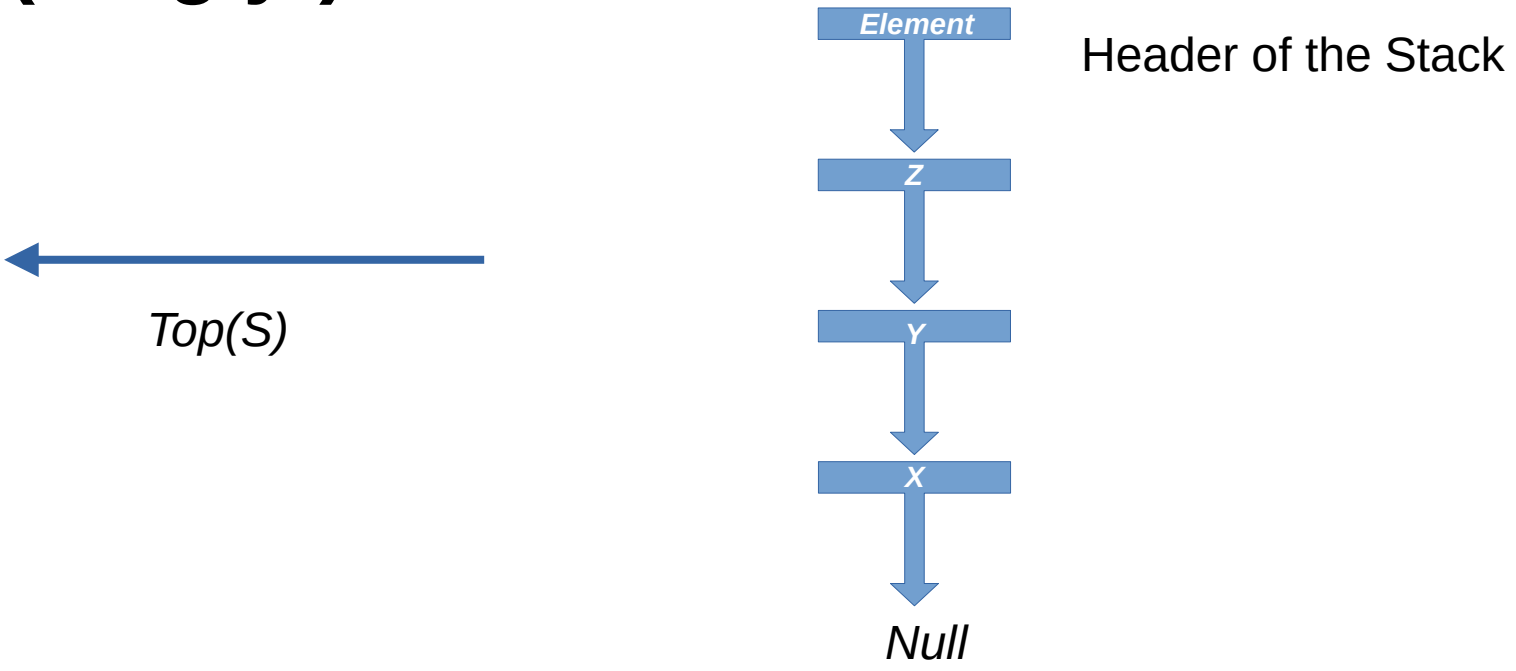
Header of the Stack



Null

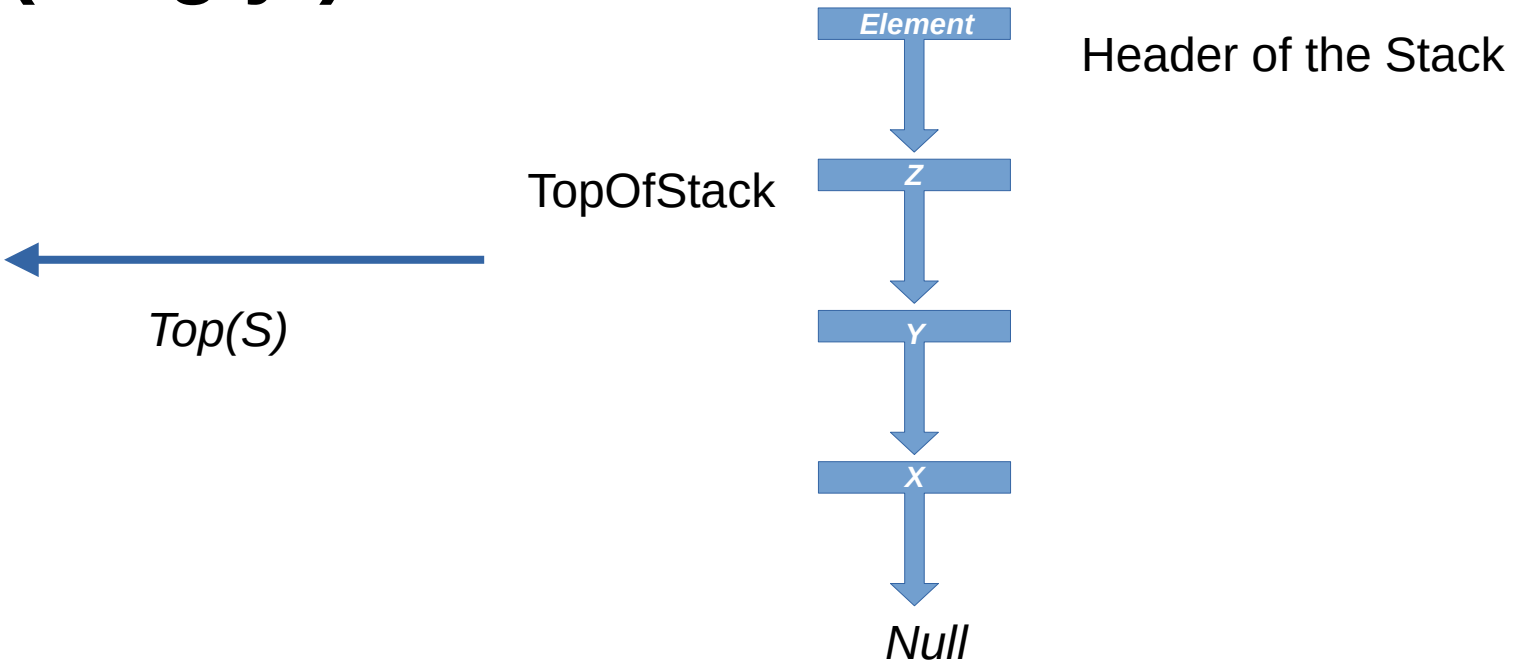
Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



Stack ADT – Implementation approaches

- **(Singly-)Linked-list**



Stack ADT – Fundamental operations

- Summary

Push(X, S), Pop(S), and Top(S) all have O(1)

Stack ADT – Up next

- Array implementation

Stack ADT – Up next

- Array implementation
- Applications of the Stack ADT

Stack ADT – Up next

- Array implementation
- Applications of the Stack ADT
  - Balancing symbols (Postfix, Infix notations)

Stack ADT – Up next

- Array implementation
- Applications of the Stack ADT
  - Balancing symbols (Postfix, Infix notations)
  - Handling function calls

Stack ADT – References

- **Book:**
  - Data structures and algorithm analysis in C by Mark Allen Weiss
- **Companion repository:**
  - **https://github.com/alianwaar73/dataStructuresC.git**
- **Contact:**
  - **Email:** **ali.anwaar73@gmail.com**