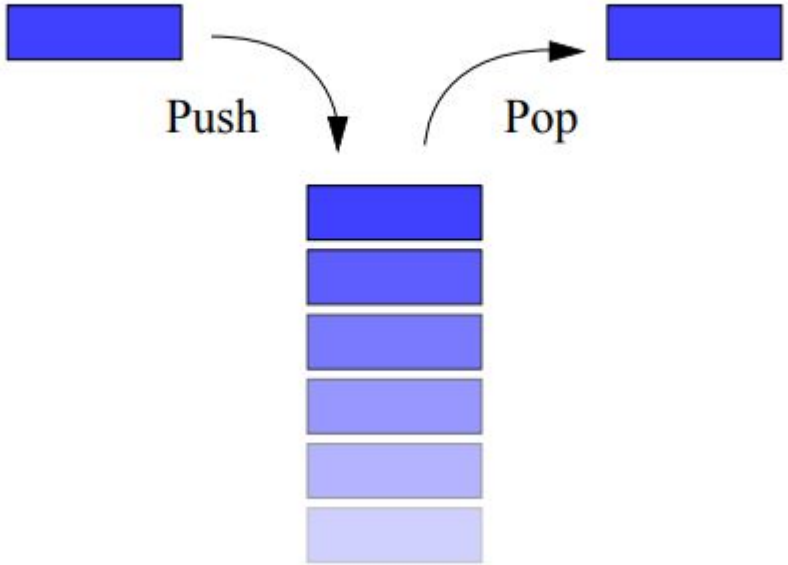
Stos jako liniowa struktura danych

Stos to liniowa struktura danych, w której dane dokładane są na wierzch stosu i z wierzchołka stosu są pobierane. Czasami określa się go jako bufor typu **LIFO** (ang. *Last In, First Out*).

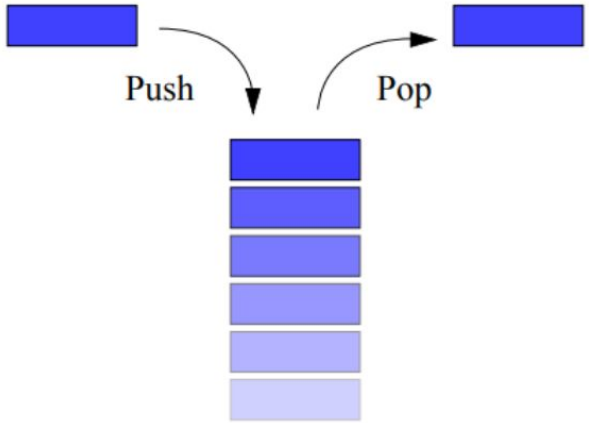


Stosy można odnaleźć w wielu przykładach z życia codziennego.
W kawiarniach i na stołówkach często mamy do czynienia ze stosami talerzy lub tac.
Na wielu biurkach znajdziemy stosy książek położonych jedna na drugiej.

Definicja stosu

Abstrakcyjny typ danych stos zdefiniowany jest jako uporządkowany zbiór elementów, w którym elementy dodawane są na jego koniec (zwany "wierzchołkiem") i z końca są ściągane. Ten typ danych musi wspierać następujące operacje:

- Stack() tworzenie nowego pustego stosu//
- push(item) wstawienie nowego elementu na wierzchołek stosu
- pop() pobranie (usunięcie z jednoczesnym zwróceniem wartości) elementu z wierzchołka stosu
- peek() odczytuje wartość z wierzchołka stosu. Nie usuwa jej (stos pozostaje niezmieniony)
- isEmpty() testuje, czy stos jest pusty
- size() liczba elementów znajdujących się na stosie



Stos jest strukturą liniowo uporządkowanych danych, z których jedynie ostatni element, zwany wierzchołkiem, jest w danym momencie dostępny.

W wierzchołku odbywa się dołączanie nowych elementów, również jedynie wierzchołek można usunąć.

Ze stosu usuwany jest element, który został dodany najpóźniej - architektura LIFO (Last In, First Out)

Stos jest bardzo często wykorzystywaną strukturą danych.

Przykłady zastosowania stosu

Stosowany jest przez procesory do chwilowego zapamiętywania rejestrów procesora, do przechowywania zmiennych lokalnych, a także w programowaniu wysokopoziomowym.

Wstecz/dalej w przeglądarkach internetowych. Przeglądarka przechowuje na stosie adresy odwiedzanych stron. Przejście do nowej strony kładzie na stos adres aktualnie opuszczanej strony. Operacja "wstecz" ściąga ten adres ze stosu i przechodzi do tego adresu. Jednocześnie operacja "wstecz" może odłożyć adres opuszczanej strony na drugi stos, używany w analogicznie sposób przez operację "dalej".

Problem weryfikacja nawiasowania, parsowanie plików HTML, XML i innych.

Parsowanie i obliczanie wyrażeń arytmetycznych.

Wyrażanie liczby w zadanym systemie liczbowym

Kroki **weryfikacji poprawności nawiasowania** (wejście - ciąg nawiasów `(` i `)`):

Kroki **weryfikacji poprawności nawiasowania** (wejście - ciąg nawiasów `(` i `)`):

1. Stwórz pusty stos. Na stos będziemy odkładać nawiasy `(`.
2. Dla każdego nawiasu z wejścia:
   2a. Jeśli nawiasem jest `(`, odłóż go na stos.
   2b. Jeśli nawiasem jest `)`: ściągnij nawias `(` ze stosu (znosząc go z właśnie odczytanym nawiasem). Jeśli stos jest pusty, to przeczytany nawias `)` nie zamyka żadnego nawiasu, więc zwróć `False`.
3. Po wyczerpaniu nawiasów z wejścia, sprawdź czy stos jest pusty i zwróć `True` jeśli tak, `False` w przeciwnym wypadku (jeśli stos nie jest pusty, to nawiasy `(` które w nim są nie są sparowane).