Лабораторная работа № 2

Тема работы:

Разработка механизма планирования деятельности операционной системы.

Теоретическая часть:

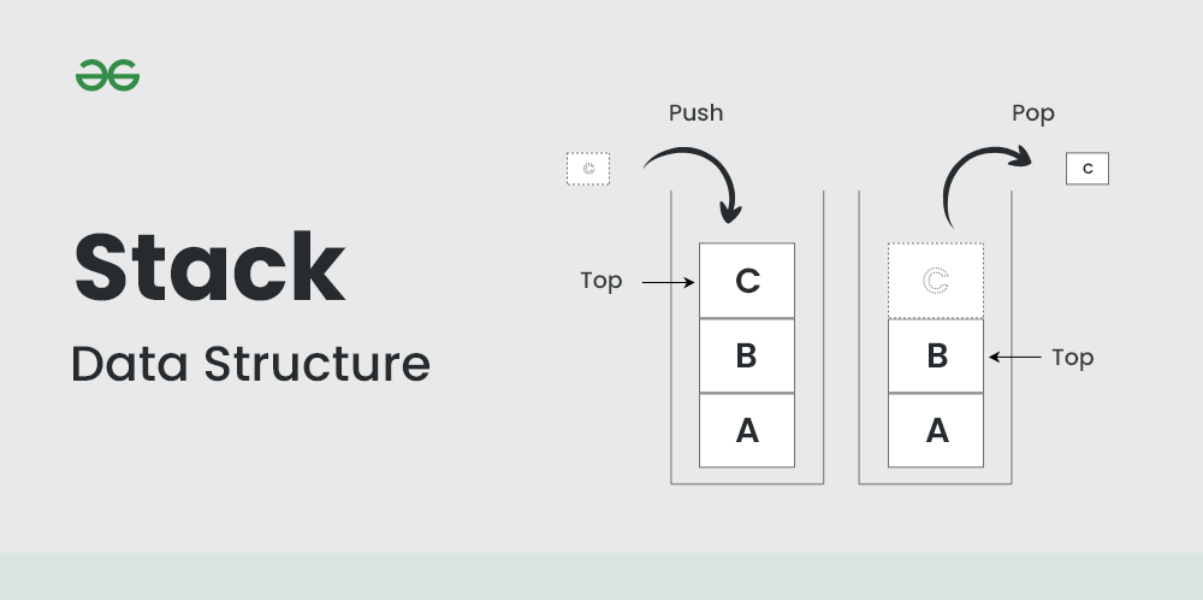
Что такое стек?

Стек – это линейная структура данных, в которой вставка нового элемента и удаление существующего происходят в одном и том же конце, представляемом как вершина стека.

Для реализации стека необходимо поддерживать указатель на вершину стека, которая является последним вставляемым элементом, поскольку мы можем обращаться к элементам только на вершине стека.

LIFO (Last In First Out - Последний внутрь, первый на выход)

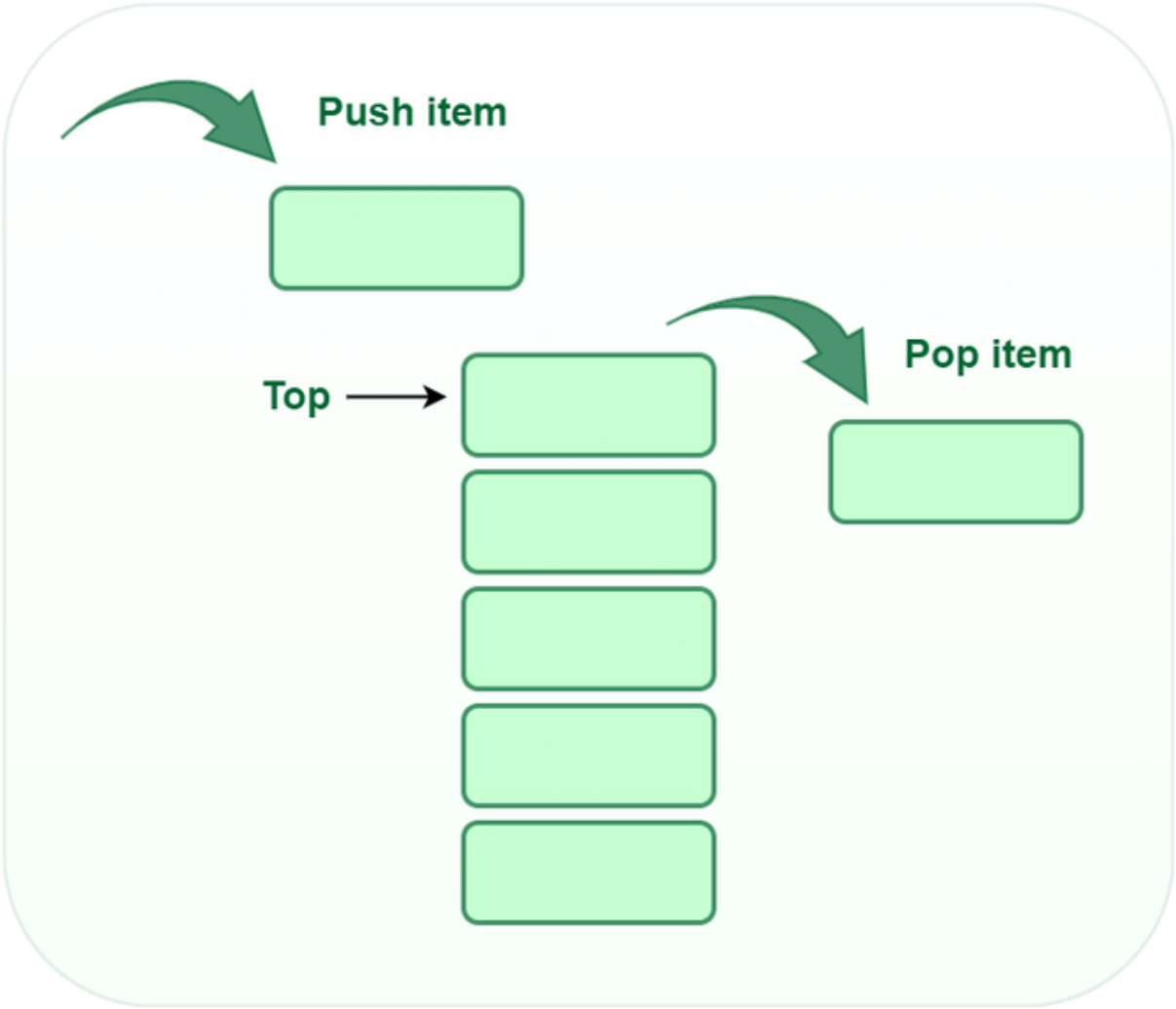
Эта стратегия гласит, что элемент, который был вставлен последним, выйдет первым. В качестве примера можно взять стопку тарелок, поставленных друг на друга. Тарелка, которую мы положили последней, находится наверху, и поскольку мы убираем тарелку, которая находится наверху, можно сказать, что тарелка, которая была положена последней, выходит первой.



Основные операции над стеком

Для того чтобы производить манипуляции в стеке, нам предоставляются определенные операции.

- **push()** - вставка элемента в стек
- **pop()** - удаление элемента из стека
- **top()** - возвращает верхний элемент стека.
- **isEmpty()** - возвращает true, если стек пуст, иначе false.
- **size()** - возвращает размер стека.



- **Push**
  Добавляет элемент в стек. Если стек переполнен, то говорят о состоянии переполнения.
  Алгоритм для push:

```
begin
 if stack is full
    return
 endif
else
 increment top
 stack[top] assign value
end else
```

```
end procedure
```

- **Pop**
  Удаление элемента из стека. Выталкивание элементов происходит в порядке, обратном их вталкиванию. Если стек пуст, то считается, что он переполнен.
  Алгоритм для pop:

```
begin
 if stack is empty
    return
 endif
else
 store value of stack[top]
 decrement top
 return value
end else
end procedure
```

- **Top**
  Возвращает верхний элемент стека.
  Алгоритм для Top:

```
begin
  return stack[top]
end procedure
```

- **isEmpty**
  Возвращает true, если стек пуст, иначе false.
  Алгоритм для isEmpty:

```
begin
 if top < 1
    return true
 else
    return false
end procedure
```

Практическое понимание стека:

Существует множество примеров стопки в реальной жизни. Рассмотрим простой пример со стопкой тарелок, поставленных друг на друга в столовой. Первой убирается тарелка, которая находится наверху, т. е. тарелка, которая была поставлена на самое нижнее место, остается в стопке дольше всего. Таким образом, можно говорить о том, что в стопке соблюдается порядок LIFO/FILO.

Типы стека

- **Стек фиксированного размера** - как следует из названия, стек фиксированного размера имеет фиксированный размер и не может динамически увеличиваться или уменьшаться. Если стек заполнен и в него пытаются добавить элемент, то возникает ошибка переполнения. Если стек пуст и из него пытаются удалить элемент, то возникает ошибка переполнения.
- **Стек динамического размера** - стек динамического размера может динамически увеличиваться или уменьшаться. При заполнении стека его размер автоматически увеличивается, чтобы вместить новый элемент, а при опустошении стека - уменьшается. Этот тип стека реализуется с помощью связного списка, так как он позволяет легко изменять размеры стека.

Реализация стека:

Стек может быть реализован с помощью массива или связанного списка. При реализации на основе массива операция push выполняется путем инкрементирования индекса верхнего элемента и сохранения нового элемента по этому индексу. Операция pop выполняется путем уменьшения индекса верхнего элемента и возврата значения, хранящегося под этим индексом. В реализации на основе связного списка операция push реализуется путем создания нового узла с новым элементом и установки следующего указателя текущего верхнего узла на этот узел. Операция pop выполняется путем установки следующего указателя текущего верхнего узла на следующий узел и возврата значения текущего верхнего узла.

Стеки широко используются в информатике для различных задач, включая оценку выражений, вызов функций и управление памятью. При оценке выражений стек может использоваться для хранения операндов и операторов по мере их обработки. При вызове функций стек может использоваться для отслеживания порядка вызова функций и возврата управления нужной функции при ее возврате. При управлении памятью стек может использоваться для хранения значений программного счетчика и регистров в компьютерной программе, что позволяет программе вернуться в предыдущее состояние при возврате функции.

Переведено с помощью www.DeepL.com/Translator (бесплатная версия)

Источник: https://www.geeksforgeeks.org/introduction-to-stack-data-structure-and-algorithm-tutorials/

Основное задание:

Реализовать алгоритм, работающий со стеком, который будет заносить информацию, обрабатывать информацию либо до внесения в стек, либо извлекая из стека и вывод той или иной информации из стека.

Варианты для реализации задания:

- **Легкий вариант:**
  Префикс - выражение называется префиксным, если оператор появляется в выражении перед операндами.
  Пример: *+AB-CD (Инфикс: (A+B) * (C-D))

  Постфикс - выражение называется постфиксным, если оператор стоит в выражении после операндов.
  Пример: AB+CD-* (Инфикс: (A+B * (C-D))

  Задав префиксное выражение, преобразуйте его в постфиксное.

  Средний вариант:
  Дана последовательность, состоящая из первых n натуральных чисел в случайном порядке. С помощью стека расположить элементы данной последовательности в возрастающем порядке.

  Сложный вариант:
  Задан массив из **n** элементов и **q** запросов, для каждого запроса, имеющего индекс элемента, найти следующий больший элемент и вывести его значение. Если справа от него нет такого большего элемента, то вывести -1.

- **Легкий вариант:**
  Постфикс - выражение называется постфиксным, если оператор стоит в выражении после операндов.
  Пример: AB+CD-* (Инфикс: (A+B * (C-D))

  Префикс - выражение называется префиксным, если оператор появляется в выражении перед операндами.
  Пример: *+AB-CD (Инфикс: (A+B) * (C-D))

  Задав постфиксное выражение, преобразуйте его в префиксное.

Средний вариант:
Ханойская башня – это математическая головоломка. Она состоит из трех столбов и нескольких дисков разного размера, которые можно нанизывать на любой столб. Головоломка начинается с того, что диски аккуратной стопкой в порядке возрастания размера располагаются на одном столбе, самый маленький - на самом верху, образуя таким образом конусообразную форму. Задача головоломки - переместить все диски с одного полюса (скажем, "исходного") на другой (скажем, "конечный") с помощью третьего полюса (скажем, вспомогательного).

В головоломке действуют следующие два правила:

- Нельзя помещать больший диск на меньший.
- Одновременно можно перемещать только один диск

Сложный вариант:
На вечеринке, состоящей из N человек, только один человек известен всем. Такой человек может присутствовать на вечеринке, если да, то он никого на ней не знает. Мы можем только задавать вопросы типа "знает ли A знакомого B? ". Найдите незнакомца (знаменитость) за минимальное количество вопросов.

- **Легкий вариант:**
Инфикс: Выражение называется Инфикс-выражением, если оператор стоит между операндами в выражении.
Пример: (A+B) * (C-D)

Префиксное: Выражение называется префиксным, если оператор стоит в выражении перед операндами.
Пример: +AB-CD (Инфикс: (A+B) (C-D))

Задав префиксное выражение, преобразуйте его в инфиксное.

Средний вариант:
Отсортировать стек при помощи временного стека в возрастающем или убывающем порядке.

Сложный вариант:
Нахождение в стеке дублирующих скобок. Набор скобок считается дублирующимся, если одно и то же подвыражение окружено несколькими

скобками. Пример: ((a+b)+((c+d)))

- **Легкий вариант:**
  Сделать реверсирование стека путем переноса из одного стека в другой

  Средний вариант:
  Сделать реверсирование стека, при этом чтобы элементы в итоге остались в исходном стеке.

  Сложный вариант:
  Сделать реверсирование стека при помощи пузырчатой сортировки.

- **Легкий вариант:**
  Определить все четные или нечетные числа в стеке.

  Средний вариант:
  Определить все числа в стеке, соответствующие конкретной математической формуле.

  Сложный вариант:
  Определить все совершенные числа в стеке. (Выполнять проверку через формулу, без заготовленного массива)

- **Легкий вариант:**
  Определить все совершенные числа в стеке.

  Средний вариант:
  Удалить средний элемент стека.

  Сложный вариант:
  Нахождение следующего меньшего из следующего большего в массиве

По завершению работы, составьте отчет, в котором должно быть – Ваша фамилия, имя, группа, тема работы, Ваш вариант для реализации задания, краткое описание реализации задания, ссылку на исходный код на GitHub. Исходный код push-ите в вашу ветку в соответствующем репозитории - https://github.com/FCIM-SO/Practice-Work-RU.

Сохранить отчет в формате PDF и отправить на ELSE - https://else.fcim.utm.md/mod/assign/view.php?id=43454.