ITMO UNIVERSITY

How to Win Coding Competitions: Secrets of Champions

Week 2: Computational complexity. Linear data structures
Lecture 5: Stack. Queue. Deque

Pavel Krotkov
Saint Petersburg 2016

Stack, Queue and Deque are just interfaces.

Stack, Queue and Deque are just interfaces.

- sometimes it's reasonable to limit set of possible operations on data structure

Stack, Queue and Deque are just interfaces.

- sometimes it's reasonable to limit set of possible operations on data structure
- implementation of Stack and Queue can be based on Vector or Array

Stack, Queue and Deque are just interfaces.

- sometimes it's reasonable to limit set of possible operations on data structure
- implementation of Stack and Queue can be based on Vector or Array
- different implementations will have different properties

Stack has only two possible operations.

- **push** — inserting element to the end of structure

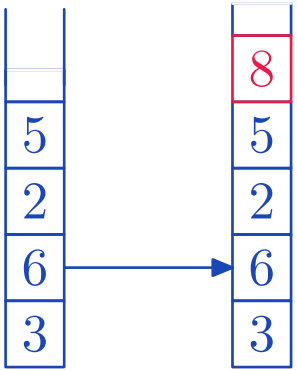
Stack has only two possible operations.

- **push** — inserting element to the end of structure
- **pop** — removing element from the end of structure and returning its value

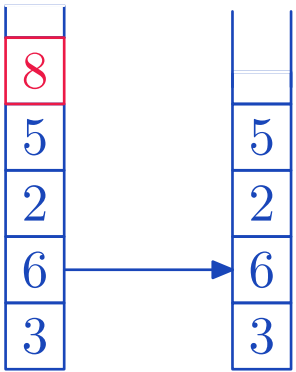
Stack has only two possible operations.

- **push** — inserting element to the end of structure
- **pop** — removing element from the end of structure and returning its value

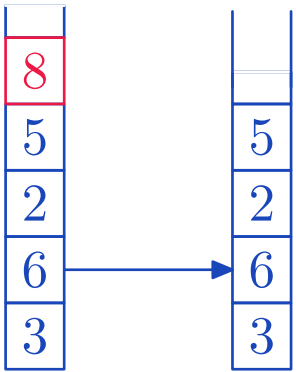
Push operation



Pop operation



Pop operation



The end of the structure is called *top of the stack*.

Stack can be implemented with vector or with list.

- both this data structures support inserting elements to and removing elements from the end of the structure in constant time

Stack can be implemented with vector or with list.

- both this data structures support inserting elements to and removing elements from the end of the structure in constant time

Stack has numerous applications in different areas.

- various graph algorithms (DFS)

Stack can be implemented with vector or with list.

- both this data structures support inserting elements to and removing elements from the end of the structure in constant time

Stack has numerous applications in different areas.

- various graph algorithms (DFS)
- local variables and function calls during execution of yor program are stored in stack

Stack can be implemented with vector or with list.

- both this data structures support inserting elements to and removing elements from the end of the structure in constant time

Stack has numerous applications in different areas.

- various graph algorithms (DFS)
- local variables and function calls during execution of yor program are stored in stack
- during calculation of expressions written in Reverse Polish notation

Stack can be implemented with vector or with list.

- both this data structures support inserting elements to and removing elements from the end of the structure in constant time

Stack has numerous applications in different areas.

- various graph algorithms (DFS)
- local variables and function calls during execution of yor program are stored in stack
- during calculation of expressions written in Reverse Polish notation
- etc.

Queue is a list organized by *FIFO* (first in - first out) rule.

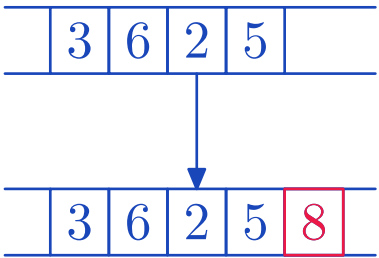
Queue is a list organized by *FIFO* (first in - first out) rule. It supports two operations:

- **enqueue** — inserting element to the end of structure
- **dequeue** — removing element from the beginning of structure

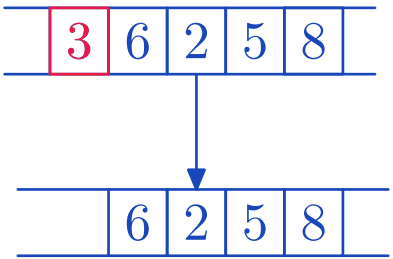
Queue is a list organized by *FIFO* (first in - first out) rule. It supports two operations:

- **enqueue** — inserting element to the end of structure
- **dequeue** — removing element from the beginning of structure

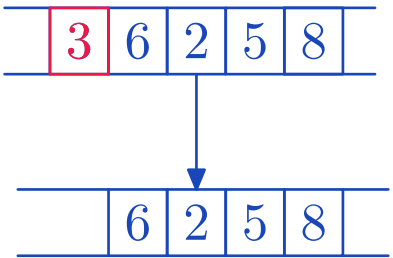
Enqueue operation



Dequeue operation

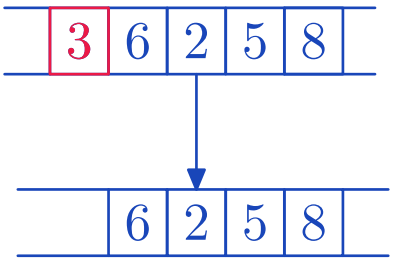


Dequeue operation



- the end of queue is called *tail*

Dequeue operation



- the end of queue is called *tail*
- the beginning of queue is called *head*

Queue implementation

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list
- queue can be implemented with array, if maximum possible size of queue is known beforehand
  - we can reuse free elements at the beginning of array for storing new elements of queue

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list
- queue can be implemented with array, if maximum possible size of queue is known beforehand
  - we can reuse free elements at the beginning of array for storing new elements of queue
- queue can be implemented with vector
  - some additional techniques of removing from the beginning of vector required

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list
- queue can be implemented with array, if maximum possible size of queue is known beforehand
  - we can reuse free elements at the beginning of array for storing new elements of queue
- queue can be implemented with vector
  - some additional techniques of removing from the beginning of vector required
- queue with amortized constant time of operations can be implemented with two stacks
  - details of implementation can be considered as a home work

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list
- queue can be implemented with array, if maximum possible size of queue is known beforehand
  - we can reuse free elements at the beginning of array for storing new elements of queue
- queue can be implemented with vector
  - some additional techniques of removing from the beginning of vector required
- queue with amortized constant time of operations can be implemented with two stacks
  - details of implementation can be considered as a home work

Queue usage.

- graph algorithms (BFS)

Queue implementation

- the easiest way to implement queue is based on list
  - we just need to store links to the first and last elements of list
- queue can be implemented with array, if maximum possible size of queue is known beforehand
  - we can reuse free elements at the beginning of array for storing new elements of queue
- queue can be implemented with vector
  - some additional techniques of removing from the beginning of vector required
- queue with amortized constant time of operations can be implemented with two stacks
  - details of implementation can be considered as a home work

Queue usage.

- graph algorithms (BFS)
- processing some queries in order of their arrival

Deque is a queue with allowed operations of removing element from the end and inserting element to the beginning.

- implementation details are very similar to queue implementation details
  - doubly linked list
  - cycled array (if fixed maximum size)
  - vector

Deque is a queue with allowed operations of removing element from the end and inserting element to the beginning.

- implementation details are very similar to queue implementation details
  - doubly linked list
  - cycled array (if fixed maximum size)
  - vector
- can be used as stack and as queue at the same time

Thank you
for your attention!