Implementation of Stack and Queue

Agenda

- ADT
- Queue
- Stack

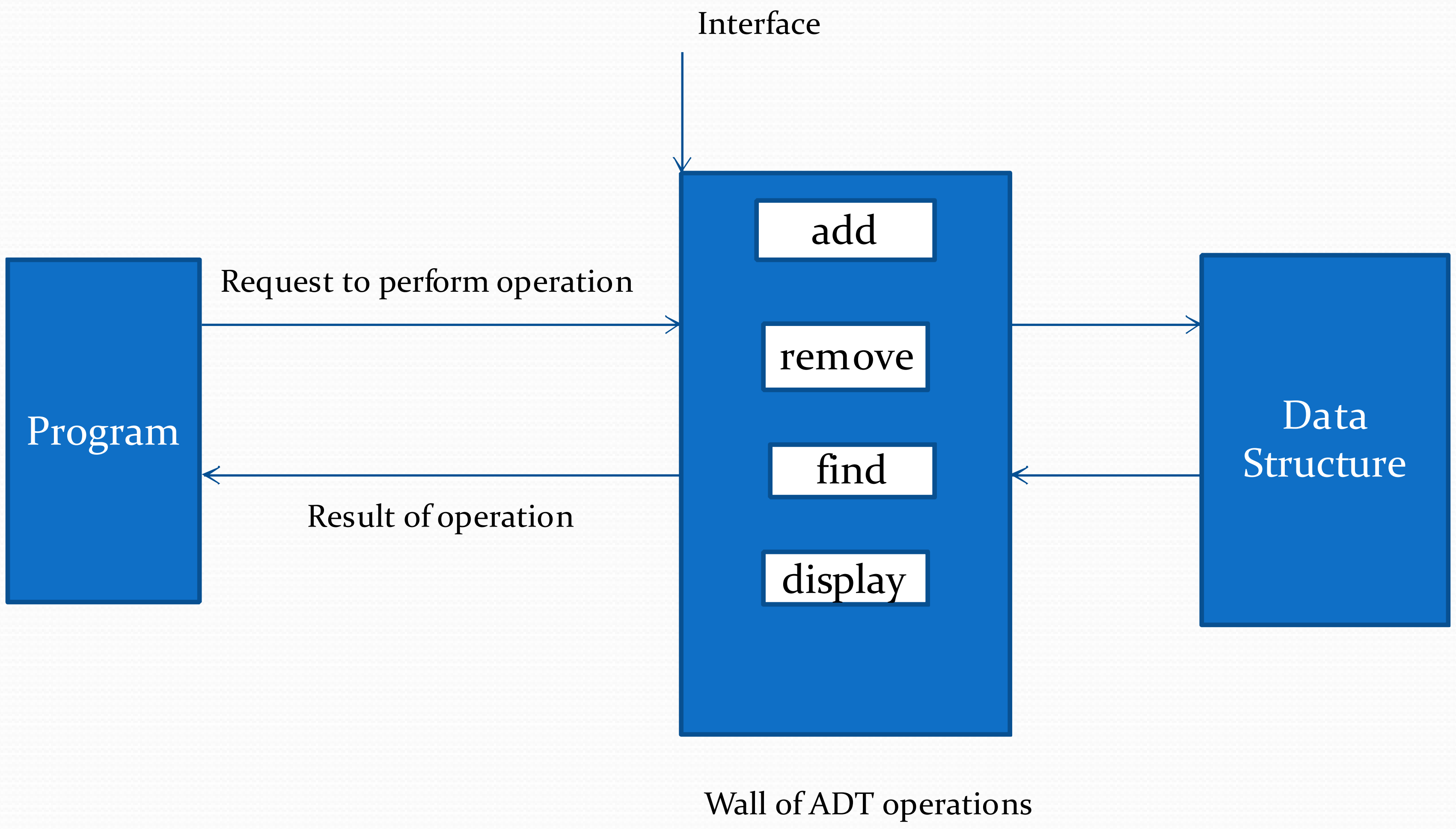
Abstract Data Types

- Data abstraction: *what* you can do to a collection of data independently of *how* you do it.

- Data abstraction is a technique that allows you to develop each data structure in relative isolation from the rest of the solution.

- Abstract data type (ADT) is a collection of data together with a set of operations on that data.

Abstract Data Types



Abstract Data Types

- ADTs versus Data Structures
  - An abstract data type is a collection of data and a set of operation on that data.
  - A data structure is a construct within a programming language that stores a collection of data.

Queue

- Queue is like a line of people. The first person to join a line is the first person served and is the first person to leave the line.
- New items enter a queue at its **back**, or **rear**.
- Items leave a queue from its **front**.
- Queue is **first-in, first-out** (FIFO)

Queue

- FIFO: The first item inserted into a queue is a first item out.
- Queue occur in everyday life
- Queue have applications in computer science, e.g. queue of printing.

Queue

ADT Queue Operations

1. Create an empty queue
2. Determine whether a queue is empty
3. Add a new item to the queue
4. Remove from the queue the item that was added earliest.
5. Remove all the items from the queue
6. Retrieve from the queue the item that was added earliest

Queue

- Pseudocode for the ADT Queue Operations

```
ADT Queue Operations
1. Create an empty queue
        +createQueue()
2. Determine whether a queue is empty
        +isEmpty(): boolean()
3. Add a new item to the queue
        +enqueue(in newItem:QueueItemType) throws
QueueException
```

Queue

4. Remove from the queue the item that was added earliest.

 +dequeue(): QueueItemType throws QueueException

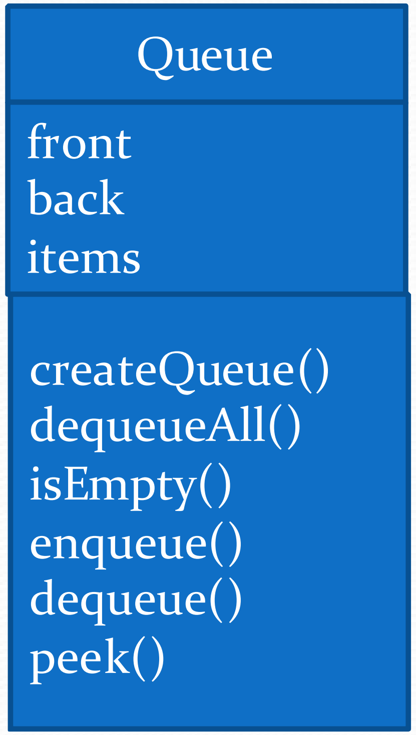
5. Remove all the items from the queue

 +dequeueAll()

6. Retrieve from the queue the item that was added earliest

 +peek():QueueItemType{query} throws QueueException

Queue



UML diagram for the class **Queue**

Queue Implementation

- Class Node and Queue are in Queuedemo.rar

Download from CMU online.

Queue Implementation

- Class Queue

Download from CMU online.

Stack

- A stack has the property that the last item placed on the stack will be the first item removed.
- This property is commonly referred to **last-in, first-out**, or simply **LIFO.**

Stack

ADT Stack Operations

1. Create an empty stack.
2. Determine whether a stack is empty.
3. Add a new item to the stack
4. Remove from the stack the item that was added most recently.
5. Remove all the items from the stack.
6. Retrieve from the stack the item that was added most recently.

Stack

Pseudocode for the ADT Stack Operations

1. Create an empty stack.

```
+createStack()
```

2. Determine whether a stack is empty.

```
+isEmpty(): boolean{query}
```

3. Add a new item to the stack

```
+push(in newItem:StackItemType) throws
StackException
```

Stack

4. Remove from the stack the item that was added most recently.

   +pop():StackItemType throws StackException

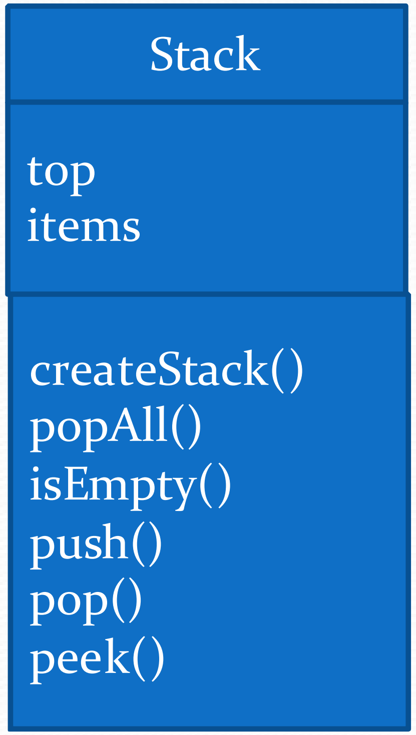
5. Remove all the items from the stack.

   +popAll()

6. Retrieve from the stack the item that was added most recently.

   +peek():StackItemType{query} throws StackException

Stack



UML diagram for the class **Stack**