Assignment 3

Time and space complexity
Stack and Queue Manipulation
Static fields and methods

Data Structures, Fall 2018
TA: Marija Stanojevic

Time complexity

$T(3n+5)=O(n)$ $T(4n^2+n*\log(n))=O(n^2)$

$T(2n*\log(n)+3n+4)=O(n*\log(n))$

Big-O complexity: remove all numbers and take just the biggest degree of n.

```
for (i = 0; i < N; i++) {
          for (i = 0; i < N; i++) {
    for (j = 0; j < M; j++) {
          sequence of statements
        sequence of statements
    }

} // O(N * M)
```

```
for (i = 0; i < N; i++) {

for (j = i+1; j < N; j++) {

sequence of statements                    }
                                       }
for (j = 0; j < M; j++) {
          } // T(N * N) => O(N^2)
                    sequence of statements
                                          } // O (N + M)
```

```
for (i = 0; i < 10; i++) {
          x = n; // O(1)
```

```
for (i = 0; i < N; i++) {
```

Space complexity

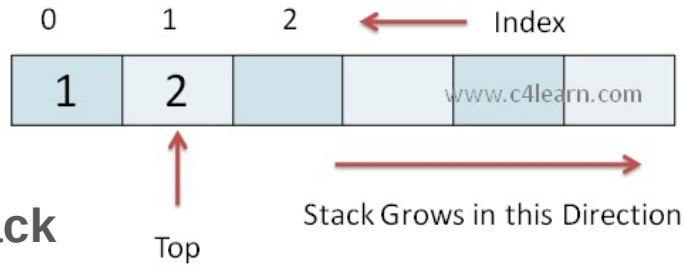
```
int sum(int x, int y) {
   int r = x + y - 20;
   return r;
} // O(1)
```

```
int sum(int a[], int n) {
   int r = 0;
   for (int i = 0; i < n; i++) {
      r += a[i];
   }
   return r;
} // O(n)
```

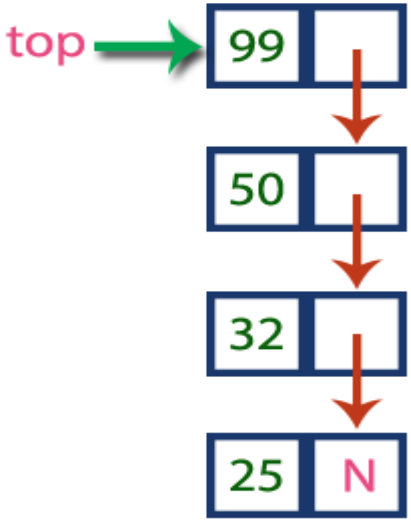
```
int multiply(int a[], int b[], int n) {
   int [][] c = new int[n][n];
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++){
         c[i][j] = a[i] * b[j];
      }
   }
   return c;
} // O(n^2)
```

- Variables have O(1) space complexity.
- Arrays/linked lists have O(n) space complexity.

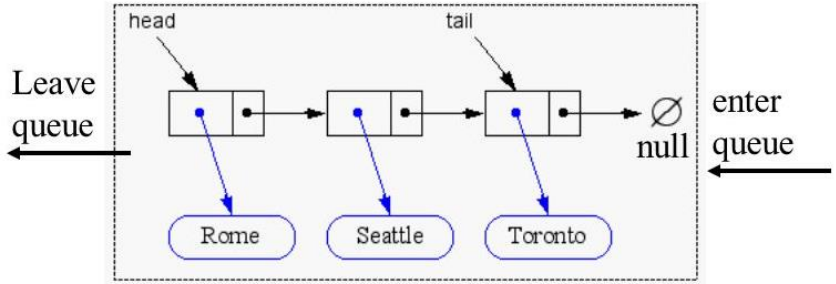
Stack



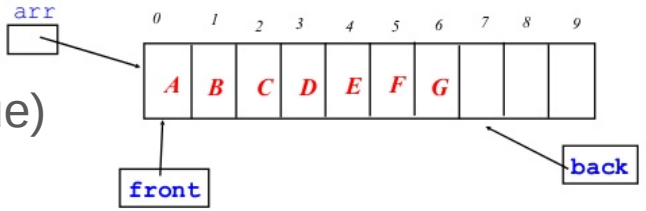
- Last In First Out Structure (LIFO)
- Maintains only one pointer to the **top of the stack**
- Usual operations:
  - push (adds new element at the top of the stack)
  - pop (removes an element from the top of the stack)
  - peek (returns value at the top of the stack)
  - empty (is top of the stack null)
  - size (number of elements in the stack)
- Can be implemented as array (java.util.Stack, java.util.ArrayDeque) or linked list (java.util.LinkedList)



Queue



- First In First Out (FIFO)
- Maintains two pointers **front/head** and **back/rear/tail**
- Usual operations:
  - offer (adds element to the end of the list), if full returns false
  - add (adds element to the end of the list), if full throws an error
  - poll (removes element from the end of the list), if empty returns false
  - remove (removes element from the beginning of the list), if empty throws an error
  - peek (shows value of the element at the beginning of the list)
  - empty (size is 0)
  - size (number of elements in the queue)
- Can be implemented as array (java.util.PriorityDeque) or linked list (java.util.LinkedList)



Static fields and functions

```
public class ExampleClass {
    private static LinkedList<String> exampleList = new LinkedList<String>();
    public static void main(String [] args) {
        int input = args[0];
        switch (input) {
                case 1:
                        addNewElement();
                        break;
                case 2:
                        ExampleClass.removeElement();
                        break;
        }
    }
        public static String addNewElement() { return “Example added”; }
        public static String removeElement() { return “Example removed”;}
```