Project Intelligent Robotics
Assignment #4

Task 4.1 Implementing Interfaces:
In addition to ROS *Topic* interfaces, which you have both published to and subscribed on in the previous assignments, you can also implement ROS *Services* and *Actions*.

You already called services, e.g., to reset the turtle's pose in turtlesim, but implementing a service server yourself is quite simple as well. You can find corresponding tutorials here:

C++: `http://wiki.ros.org/ROS/Tutorials/WritingServiceClient(c++)`
Python: `http://wiki.ros.org/ROS/Tutorials/WritingServiceClient(python)`

When you planned and executed robot motions through MoveIt, you already used Actions too, even if you only accessed MoveIt's helper classes that abstract away the full action interface.

C++: `http://wiki.ros.org/actionlib_tutorials/Tutorials/SimpleActionServer(ExecuteCallbackMethod)`
Py: `https://www.theconstructsim.com/ros-5-mins-033-create-ros-action-server/`

Look at the respective tutorials and understand how to implement both communication interfaces.

Task 4.2 Collaborative Project: The aim of this task is to foster successful group interaction and discussion. The supervisors will not get involved unless asked for assistance.

Implement a software system to count from 0 to 20 with the following restrictions:

- You have to use a ROS interface/interfaces to achieve this behavior.
- The running system must execute original code written by each team member. Push your individual components to the tutorial repository.
- Nobody is allowed to edit code of other team members. Communicate instead.
- At the end of this session, everyone must be able to run the full system.

4.2.1: Gather as a group and work on a concept to reach this goal. Plan your communication.

4.2.2: Define common interfaces yourself for everyone to use, e.g., a new *Message type*, *Service* or *Action definition*.

4.2.3: To simplify startup, create a roslaunch file that starts all components.
You can find the relevant documentation here: `http://wiki.ros.org/roslaunch`