练习题一：

修改 roslab0 中的 talker.py，使其按照命令行参数输入的频率发布话题，而非按照程序中固定的频率。

【提示】：
可参照下面的片段修改程序：

```
if rospy.has_param('~rate'):
    ra = rospy.get_param('~rate')
else:
    rospy.logwarn('no parameter given; using the default value %d' %ra)

rate = rospy.Rate(ra)
```

运行如下命令执行节点：

% rosrun roslab0 talker.py _rate:=50

练习题二：

在你的 roslab0 功能包的 scripts 子目录中，输入一个新的简单 Python 程序如下，命名为 param.py。

```
#!/usr/bin/env python

"""Just testing parameter mapping."""

import rospy

rospy.init_node("param")

print(rospy.get_param("foo"))

print(rospy.get_param("bar"))

print(rospy.get_param("~baz"))
```

注意：此处，foo, bar 为全局参数，而~baz 则为节点 param 的私有参数。
输入后，不要忘记将此程序属性改为可执行。(chmod +x param.py, 然后用 ls 查看，文件名由白色变为绿色，即为可执行 python 文件)

进入 launch 子目录，输入如下的新 launch 文件，命名为 param.launch。

```
<launch>

   <node name="param" pkg="roslab0" type="param.py" output="screen">

      <remap from="bar" to="foo" />

      <remap from="~baz" to="foo" />

   </node>

</launch>
```

注意：全局和私有参数。remap 表示将名称改变。

在命令行运行：

% rosparam set foo 42

可查看 foo 参数被赋予了一个值：

% rosparam get foo

运行节点：

% roslaunch roslab0 param.launch

你会看到，三个参数均被赋值为 foo 的值。

思考题：

【需要已经运行 ROS Master】

% rosrun turtlesim turtlesim_node

打开一个乌龟窗口。

% rostopic list

你会看到一系列话题，注意有一个乌龟的运动相关话题 /turtle1/cmd_vel，并没有名为 /cmd_vel 的话题。

发出一个命令行指令如下，驱动乌龟运动：

% rostopic pub -r 10 turtle1/cmd_vel geometry_msgs/Twist "{linear: {x: 0.1, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"

按 Ctrl +c 退出。

现在，请输入下面命令行：

% rostopic pub -r 10 /cmd_vel geometry_msgs/Twist "{linear: {x: 0.1, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"

乌龟不会运动，分析原因。请找到办法，使得乌龟在这条命令执行后也能运动。