

Lab 1

Exercise 0:

publisher_node.cpp:

```
#include "ros/ros.h"
#include "std_msgs/Int32.h"
#include <sstream>

class MyPublisher {
public:
    MyPublisher() {
        pub = n.advertise<std_msgs::Int32>("/Kantoreyeva_topic", 9);
        rate_1hz = new ros::Rate(1);
        rate_50hz = new ros::Rate(50);
    }

    void publishNumbers() {
        std::vector<int> idNumbers = {2,0,1,6,2,5,7,2,3};
        while (ros::ok()) {
            for (const auto &digit : idNumbers) {
                std_msgs::Int32 msg;
                msg.data = digit;
                pub.publish(msg);
                rate_1hz->sleep();
            }

            for (int i = 0; i < 50; ++i) {
                for (const auto &digit : idNumbers) {
                    std_msgs::Int32 msg;
                    msg.data = digit;
                    pub.publish(msg);
                    rate_50hz->sleep();
                }
            }
        }
    }

private:
    ros::NodeHandle n;
    ros::Publisher pub;
    ros::Rate *rate_1hz;
    ros::Rate *rate_50hz;
};

int main(int argc, char **argv) {
    ros::init(argc, argv, "my_publisher_node");
    MyPublisher myPublisher;
    myPublisher.publishNumbers();

    return 0;
}
```

subscriber_node.cpp:

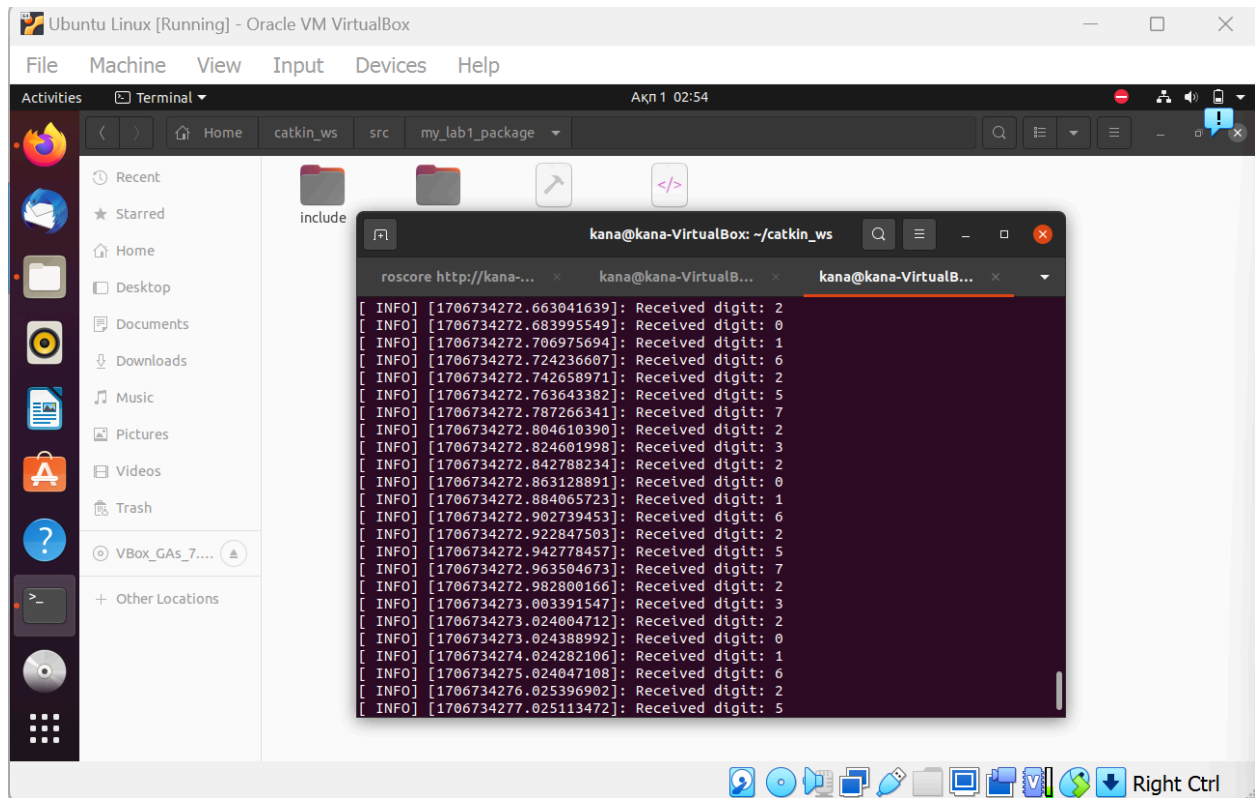
```
#include "ros/ros.h"
#include "std_msgs/Int32.h"

void callback(const std_msgs::Int32::ConstPtr &msg) {
    ROS_INFO("Received digit: %d", msg->data);
}

int main(int argc, char **argv) {
    ros::init(argc, argv, "my_subscriber_node");
    ros::NodeHandle n;
    ros::Subscriber sub = n.subscribe("/Kantoreyeva_topic", 9, callback);
    ros::spin();

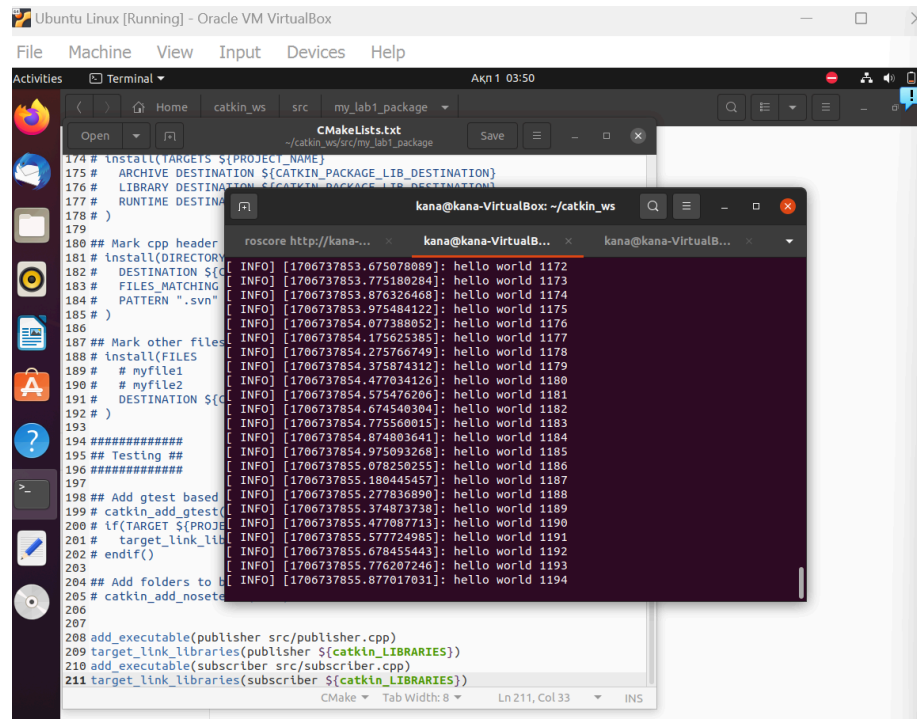
    return 0;
}
```

Screenshot:



Exercise 1:

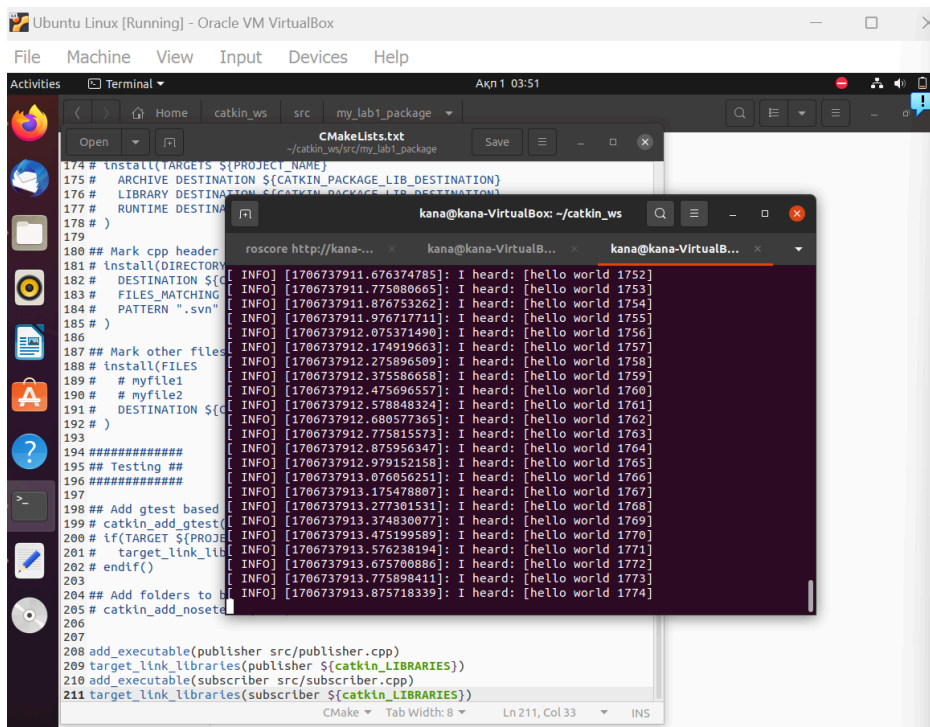
Screenshots:



```
174 # install(TARGETS ${PROJECT_NAME}
175 # ARCHIVE DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
176 # LIBRARY DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
177 # RUNTIME DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
178 # )
179
180 ## Mark cpp header
181 # install(DIRECTORY
182 # DESTINATION ${CATKIN_PACKAGE_INCLUDE_DESTINATION}
183 # FILES_MATCHING
184 # PATTERN "*.h"
185 # )
186
187 ## Mark other files
188 # install(FILES
189 # myfile1
190 # myfile2
191 # DESTINATION ${CATKIN_PACKAGE_INCLUDE_DESTINATION}
192 # )
193
194 #####
195 ## Testing ##
196 #####
197
198 ## Add gtest based
199 # catkin_add_gtest(
200 # if(TARGET ${PROJECT_NAME})
201 # target_link_libraries(
202 # endif()
203
204 ## Add folders to the
205 # catkin_add_nosetup
206
207
208 add_executable(publisher src/publisher.cpp)
209 target_link_libraries(publisher ${catkin_LIBRARIES})
210 add_executable(subscriber src/subscriber.cpp)
211 target_link_libraries(subscriber ${catkin_LIBRARIES})
```

Output from terminal:

```
roscore http://kana... kana@kana-VirtualB... kana@kana-VirtualB...
[INFO] [1706737853.675078089]: hello world 1172
[INFO] [1706737853.775180284]: hello world 1173
[INFO] [1706737853.876326468]: hello world 1174
[INFO] [1706737853.975484122]: hello world 1175
[INFO] [1706737854.077388052]: hello world 1176
[INFO] [1706737854.175625385]: hello world 1177
[INFO] [1706737854.275768749]: hello world 1178
[INFO] [1706737854.375874312]: hello world 1179
[INFO] [1706737854.477034126]: hello world 1180
[INFO] [1706737854.575476206]: hello world 1181
[INFO] [1706737854.674540304]: hello world 1182
[INFO] [1706737854.775560015]: hello world 1183
[INFO] [1706737854.874803641]: hello world 1184
[INFO] [1706737854.975093268]: hello world 1185
[INFO] [1706737855.075292555]: hello world 1186
[INFO] [1706737855.184454577]: hello world 1187
[INFO] [1706737855.277836890]: hello world 1188
[INFO] [1706737855.374873738]: hello world 1189
[INFO] [1706737855.477087713]: hello world 1190
[INFO] [1706737855.577724985]: hello world 1191
[INFO] [1706737855.678455443]: hello world 1192
[INFO] [1706737855.776207246]: hello world 1193
[INFO] [1706737855.877017031]: hello world 1194
```



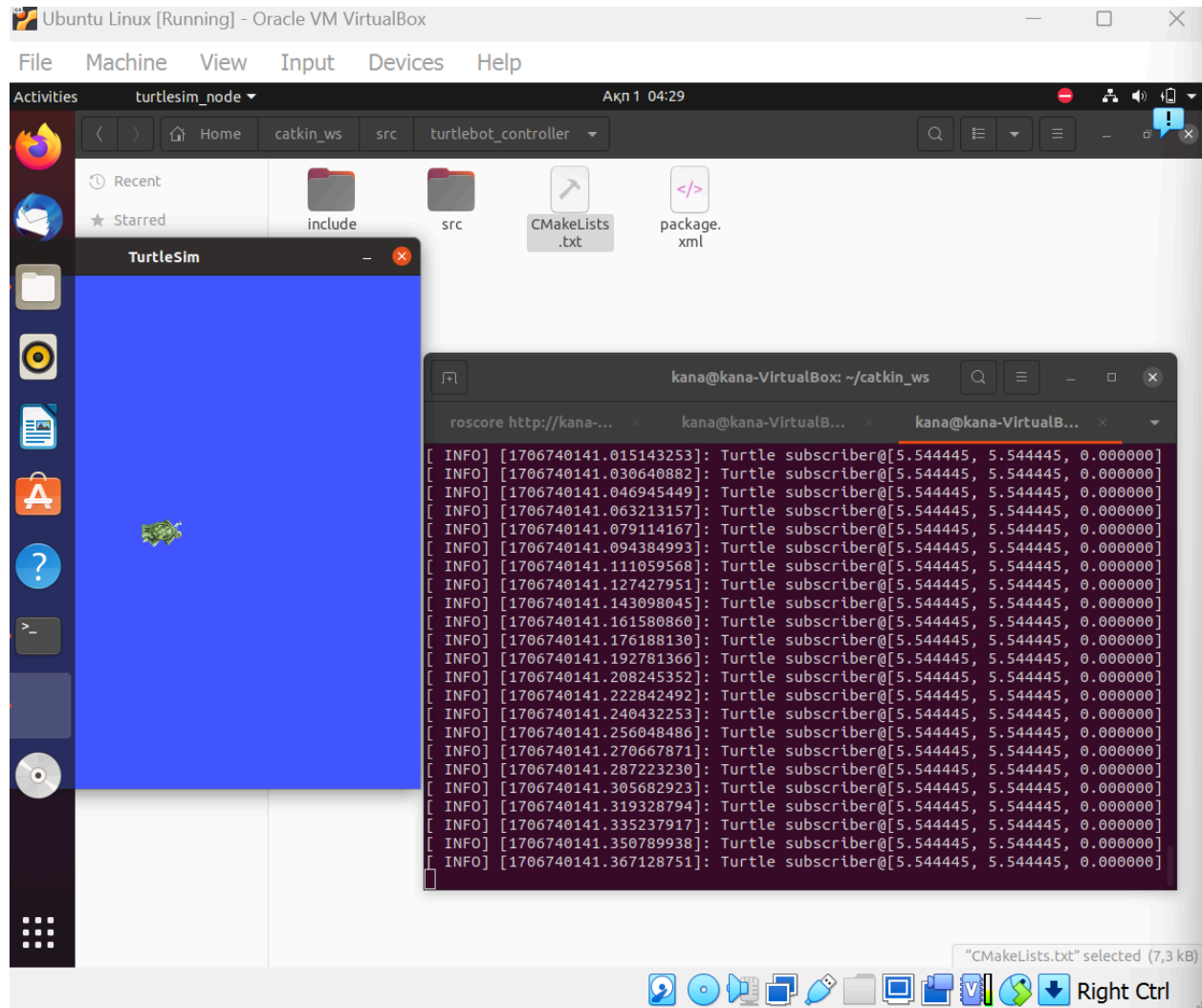
```
174 # install(TARGETS ${PROJECT_NAME}
175 # ARCHIVE DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
176 # LIBRARY DESTINATION ${CATKIN_PACKAGE_LIB_DESTINATION}
177 # RUNTIME DESTINATION ${CATKIN_PACKAGE_BIN_DESTINATION}
178 # )
179
180 ## Mark cpp header
181 # install(DIRECTORY
182 # DESTINATION ${CATKIN_PACKAGE_INCLUDE_DESTINATION}
183 # FILES_MATCHING
184 # PATTERN "*.h"
185 # )
186
187 ## Mark other files
188 # install(FILES
189 # myfile1
190 # myfile2
191 # DESTINATION ${CATKIN_PACKAGE_INCLUDE_DESTINATION}
192 # )
193
194 #####
195 ## Testing ##
196 #####
197
198 ## Add gtest based
199 # catkin_add_gtest(
200 # if(TARGET ${PROJECT_NAME})
201 # target_link_libraries(
202 # endif()
203
204 ## Add folders to the
205 # catkin_add_nosetup
206
207
208 add_executable(publisher src/publisher.cpp)
209 target_link_libraries(publisher ${catkin_LIBRARIES})
210 add_executable(subscriber src/subscriber.cpp)
211 target_link_libraries(subscriber ${catkin_LIBRARIES})
```

Output from terminal:

```
roscore http://kana... kana@kana-VirtualB... kana@kana-VirtualB...
[INFO] [1706737911.676374785]: I heard: [hello world 1752]
[INFO] [1706737911.775080665]: I heard: [hello world 1753]
[INFO] [1706737911.876753262]: I heard: [hello world 1754]
[INFO] [1706737911.976717711]: I heard: [hello world 1755]
[INFO] [1706737912.075371490]: I heard: [hello world 1756]
[INFO] [1706737912.174919663]: I heard: [hello world 1757]
[INFO] [1706737912.275896509]: I heard: [hello world 1758]
[INFO] [1706737912.375586658]: I heard: [hello world 1759]
[INFO] [1706737912.475696557]: I heard: [hello world 1760]
[INFO] [1706737912.578848324]: I heard: [hello world 1761]
[INFO] [1706737912.680577365]: I heard: [hello world 1762]
[INFO] [1706737912.775815573]: I heard: [hello world 1763]
[INFO] [1706737912.875956347]: I heard: [hello world 1764]
[INFO] [1706737912.979152158]: I heard: [hello world 1765]
[INFO] [1706737913.076056251]: I heard: [hello world 1766]
[INFO] [1706737913.175478807]: I heard: [hello world 1767]
[INFO] [1706737913.277301531]: I heard: [hello world 1768]
[INFO] [1706737913.374830077]: I heard: [hello world 1769]
[INFO] [1706737913.475199589]: I heard: [hello world 1770]
[INFO] [1706737913.576238194]: I heard: [hello world 1771]
[INFO] [1706737913.675700886]: I heard: [hello world 1772]
[INFO] [1706737913.775898411]: I heard: [hello world 1773]
[INFO] [1706737913.875718339]: I heard: [hello world 1774]
```

codes were provided

Exercise 2: Screenshots:



codes were provided

Exercise 3:

turtle_listener.cpp:

```
#include "ros/ros.h"
```

```
#include "turtlesim/Pose.h"
```

```
#include "geometry_msgs/Twist.h"
```

```
ros::Publisher pub;
```

```
void turtleCallback(const turtlesim::Pose::ConstPtr& msg)
```

```
{
```

```
    ROS_INFO("Turtle subscriber @ [%f, %f, %f]", msg->x, msg->y, msg->theta);
```

```

    geometry_msgs::Twist my_vel;
    my_vel.linear.x = 1.0;
    my_vel.angular.z = 1.0;
    pub.publish(my_vel);
}

int main(int argc, char** argv)
{
    // Initialize the node, setup the NodeHandle for handling the communication with the ROS system
    ros::init(argc, argv, "turtlebot_subscriber");
    ros::NodeHandle nh;

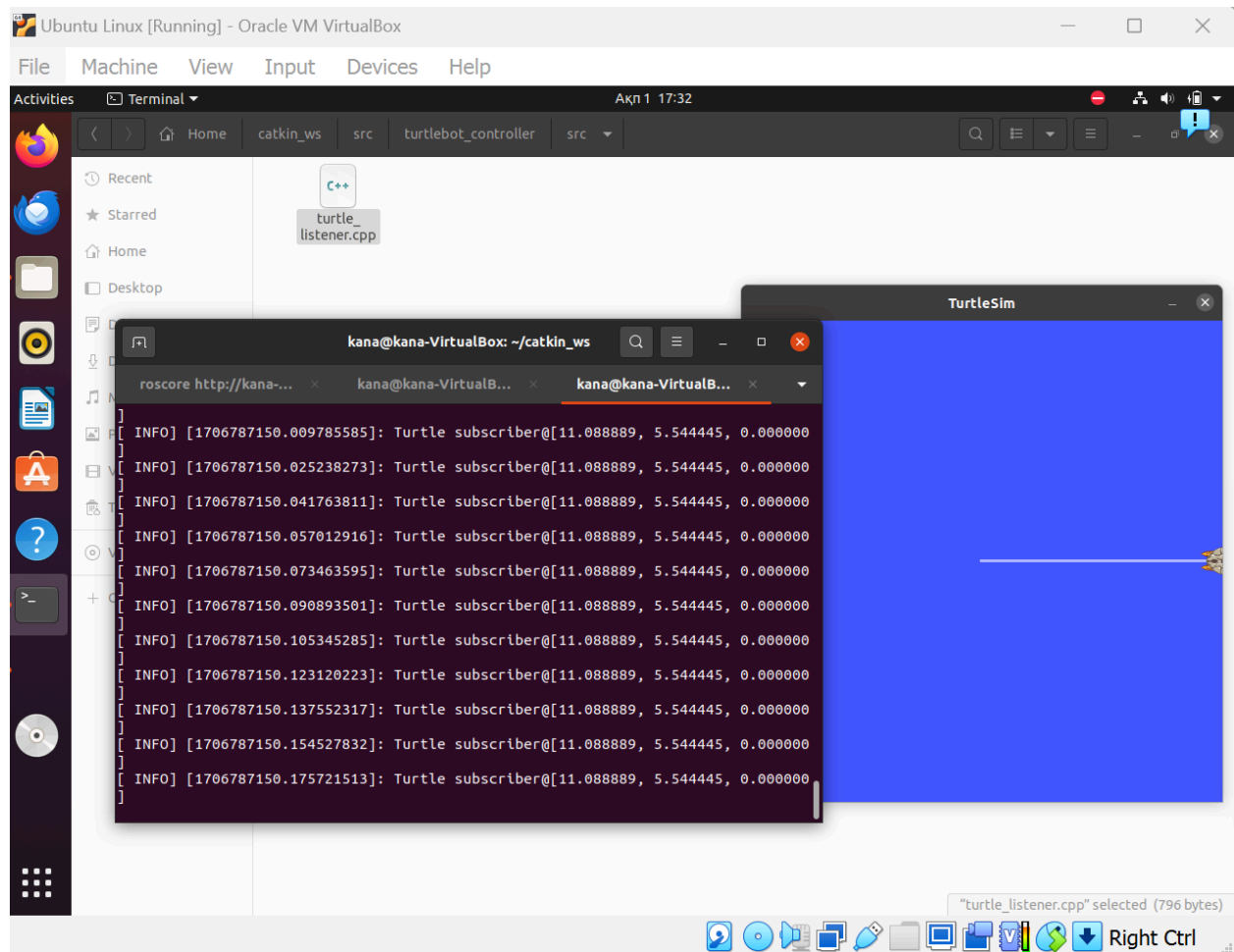
    pub = nh.advertise<geometry_msgs::Twist>("turtle1/cmd_vel", 1);

    // Define the subscriber to turtle's position
    ros::Subscriber sub = nh.subscribe("turtle1/pose", 1, turtleCallback);

    ros::spin();
    return 0;
}

```

Screenshot:



Exercise 4:

turtle_listener.cpp:

```
#include "ros/ros.h"
#include "turtlesim/Pose.h"
#include "geometry_msgs/Twist.h"
#include "turtlesim/Spawn.h"

ros::Publisher pub;
ros::ServiceClient client1;

void turtleCallback(const turtlesim::Pose::ConstPtr& msg)
{
    ROS_INFO("Turtle subscriber @ [%f, %f, %f]", msg->x, msg->y, msg->theta);

    geometry_msgs::Twist my_vel;
    my_vel.linear.x = 1.0;
    my_vel.angular.z = 1.0;
    pub.publish(my_vel);
}

int main(int argc, char** argv)
{
    // Initialize the node, setup the NodeHandle for handling the communication with the ROS system
    ros::init(argc, argv, "turtlebot_subscriber");
    ros::NodeHandle nh;

    pub = nh.advertise<geometry_msgs::Twist>("turtle1/cmd_vel", 1);

    ros::Subscriber sub = nh.subscribe("turtle1/pose", 1, turtleCallback);

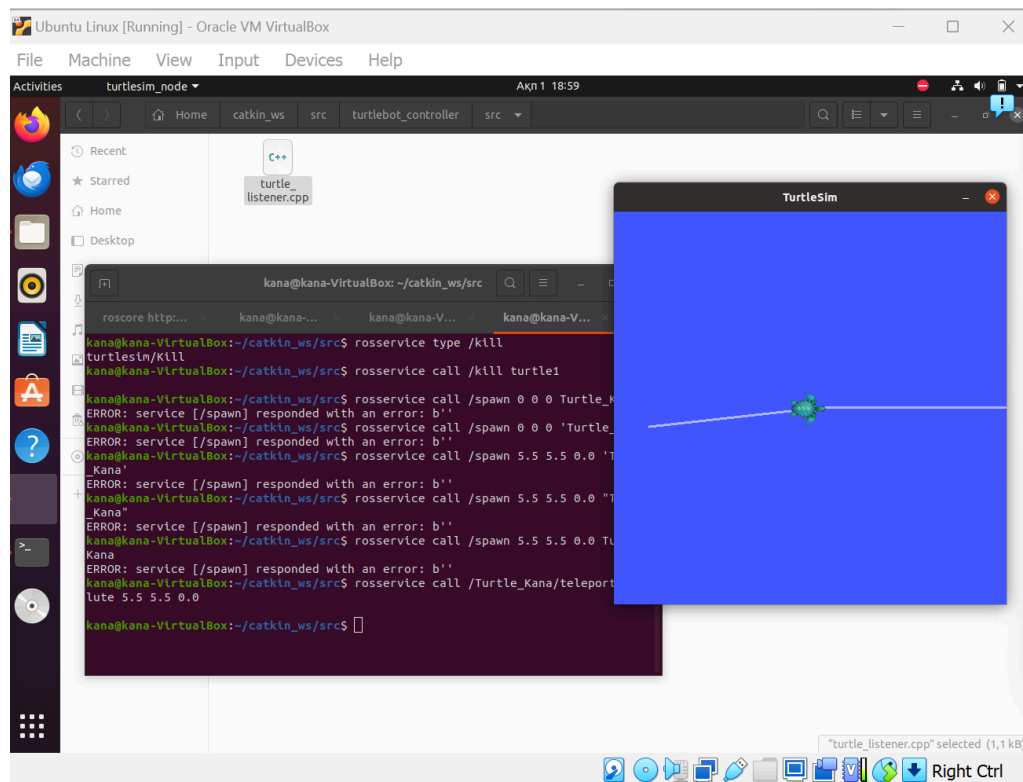
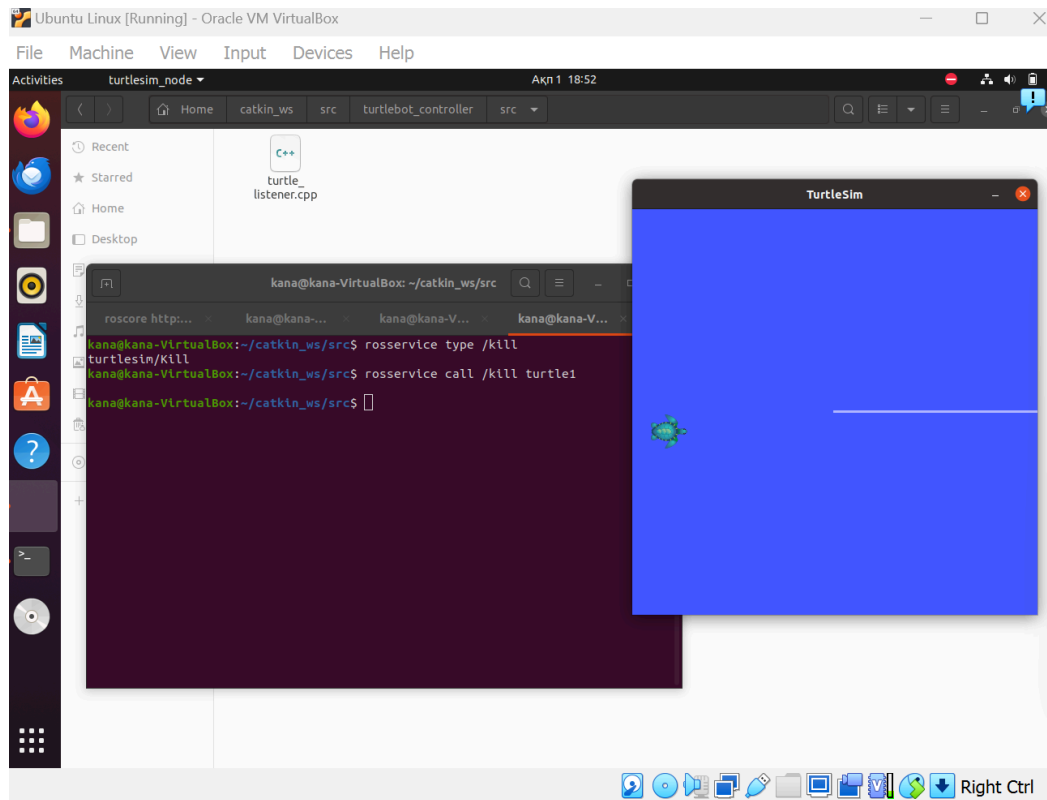
    client1 = nh.serviceClient<turtlesim::Spawn>("/spawn");

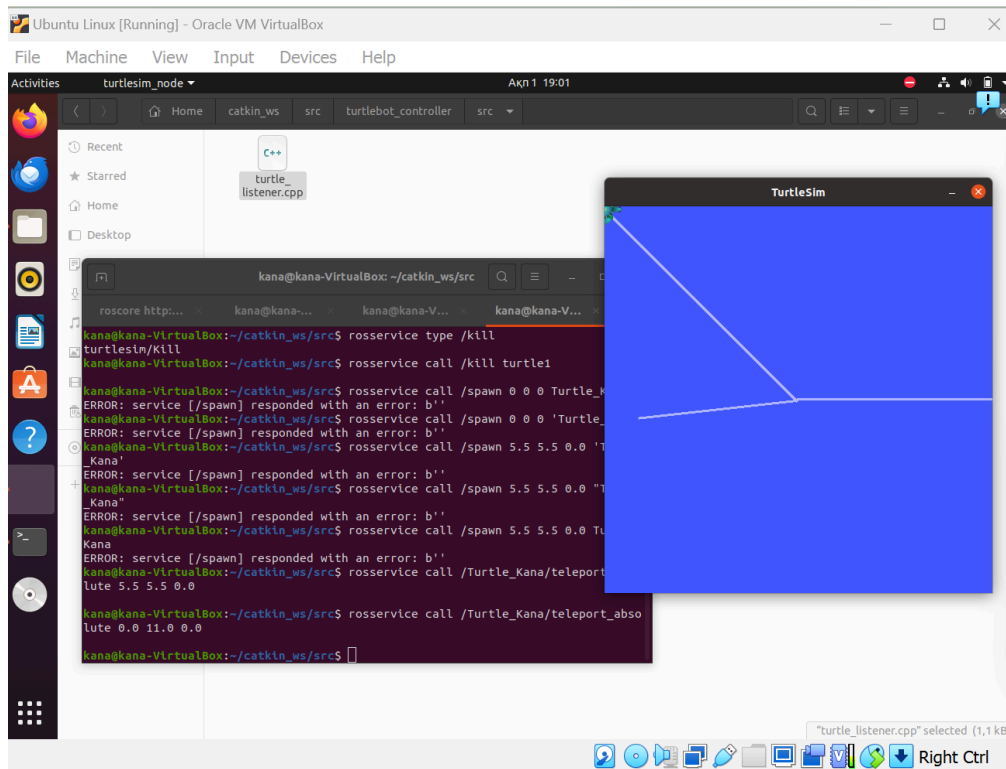
    turtlesim::Spawn srv1;
    srv1.request.x = 1.0;
    srv1.request.y = 5.0;
    srv1.request.theta = 0.0;
    srv1.request.name = "Turtle_Kana";

    client1.call(srv1);

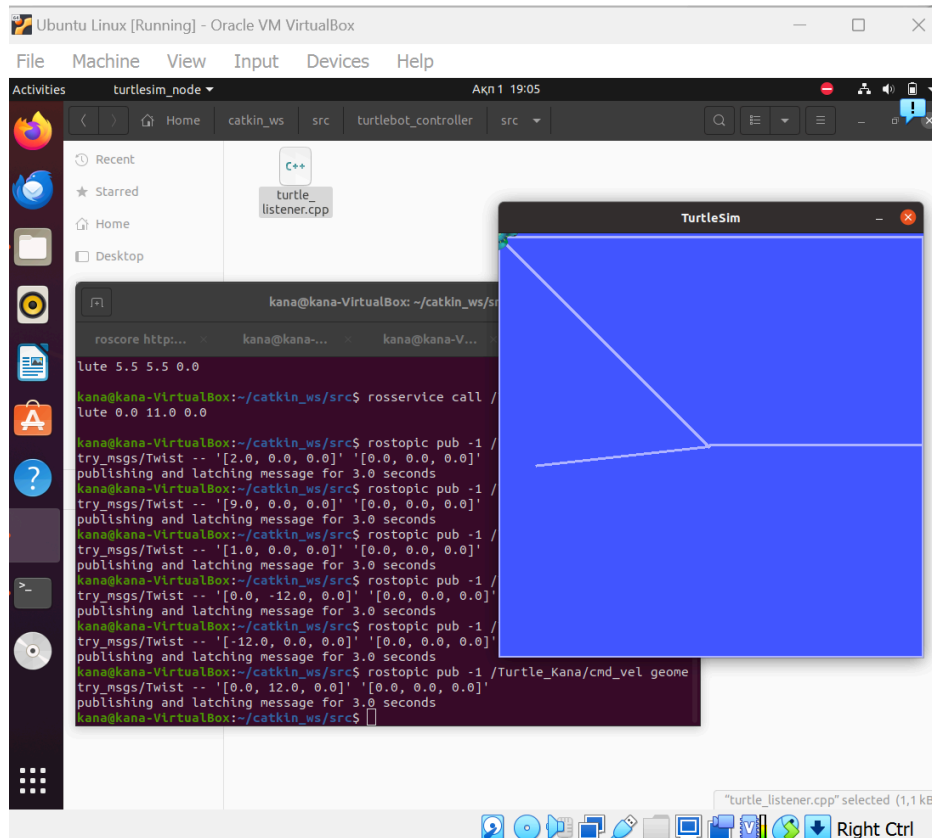
    ros::spin();
    return 0;
}
```

Exercise 5: Screenshots:





Square trajectory:



Triangular trajectory:

