ROS2_day1-2

day1-2

day1-1에서 다룬 기능들은 제외하겠습니다.

여기서는 main.cpp파일은 따로 만들지 않아서 while(rclcpp::ok())을 사용해 cli가 끊기지 않게 하였다.

rclcpp::ok()는 ros2 노드가 실행중인지 확인하기 위해 루프내에서 사용된다. 즉 ctrl+c 누르면 false로 설정

모드 도형 그리기 선택시

각 함수 실행

```
if (shape == 1)
        {
            draw_triangle(size);
        }
        else if (shape == 2)
        {
            draw_circle(size);
        }
        else if (shape == 3)
        {
            draw_square(size);
        }
```

```
void TurtlesimDraw::draw_triangle(int size)
{
    auto msg = geometry_msgs::msg::Twist();
    for (int i = 0; i < 3; i++)
    {
        msg.linear.x = size;
        msg.angular.z = 0.0;
        cmd_vel_pub_->publish(msg);
```

```
        rclcpp::sleep_for(std::chrono::seconds(1));
        msg.linear.x = 0.0;
        msg.angular.z = 2.094; // 120도
        cmd_vel_pub_->publish(msg);
        rclcpp::sleep_for(std::chrono::seconds(1));
    }
}
```

120도를 직접 잡아주어 3번 반복하여 삼각형
직진→회전(3번 반복)

rclcpp::sleep_for(std::chrono::seconds(1));
: 스레드를 멈추게 하는 부분인데 딜레이 기능이라고 생각해 직진후 충분한 회전시간을 주기 위해 넣었다.

```
void TurtlesimDraw::draw_circle(int size)
{
    auto msg = geometry_msgs::msg::Twist();
    for (int i = 0; i < 12; i++)
    {
        msg.linear.x = size / 2;
        msg.angular.z = 0.5236; // 30도
        cmd_vel_pub_->publish(msg);
        rclcpp::sleep_for(std::chrono::seconds(1));
    }
}
```

마찬가지로 30도로 12번 돌려주었다.
대신 원으로 그려야 하기에 ex(1.0, 1.0)식으로 자동차처럼 앞으로 가며 회전하는게 특징

```
void TurtlesimDraw::draw_square(int size)
{
    auto msg = geometry_msgs::msg::Twist();
    for (int i = 0; i < 4; i++)
    {
        msg.linear.x = size;
        msg.angular.z = 0.0;
        cmd_vel_pub_->publish(msg);
```

```
            rclcpp::sleep_for(std::chrono::seconds(1));
            msg.linear.x = 0.0;
            msg.angular.z = 1.5708; // 90도
            cmd_vel_pub_->publish(msg);
            rclcpp::sleep_for(std::chrono::seconds(1));
        }
    }
```

90도 4번 반복

로직은 삼각형과 동일하다