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

import rospy
from turtlesim.msg import Pose
from geometry_msgs.msg import Twist
from math import atan2, tan

# Variables globales
pose = Pose()
waypoint = (7, 7)

# Callback de la souscription au topic "pose"
def pose_callback(data):
    global pose
    pose = data

# Calcul de l'angle désiré
def calculate_desired_angle():
    global pose, waypoint
    angle_desired = atan2(waypoint[1] - pose.y, waypoint[0] - pose.x)
    return angle_desired

# Fonction principale
def set_way_point():
    # Initialisation du nœud ROS
    rospy.init_node('set_way_point', anonymous=True)

    # Souscription au topic "pose"
    rospy.Subscriber("pose", Pose, pose_callback)

    # Création du publisher pour cmd_vel
```

```
    cmd_vel_pub = rospy.Publisher('cmd_vel', Twist, queue_size=10)

    # Paramètre Kp (constante de régulation en cap)
    kp = rospy.get_param('~Kp', 1.0)

    # Taux de rafraîchissement
    rate = rospy.Rate(10)  # 10Hz

    while not rospy.is_shutdown():
        # Calcul de l'angle désiré
        angle_desired = calculate_desired_angle()

        # Calcul de l'erreur en cap
        error = atan2(tan(angle_desired - pose.theta), 1)

        # Commande en cap
        u = kp * error

        # Création du message Twist pour la commande de vitesse angulaire
        twist_msg = Twist()
        twist_msg.angular.z = u

        # Publication du message Twist
        cmd_vel_pub.publish(twist_msg)

        rate.sleep()

if __name__ == '__main__':
    try:
        set_way_point()
    except rospy.ROSInterruptException:
```

pass