1. Task I

   (a) Data generation: The function *generatearima011(mu0, theta, drift, N, sigma)* simulates ARIMA(0,1,1) process.

   (b) The function ARIMA011.train(data) supposed to fit the maximum likelihood parameters but I didn't manage to make it work. I understood the task to mean that I should do the optimization myself and not use an existing implementation of ARIMA. I think that my derivation is correct but the optimization algorithm fails. The optimization algorithms makes σ negative pretty quickly, which makes the maximum likelihood function undefined.
   Derivation: the model is

   $$Y_t - Y_{t-1} = \text{drift} + e_t - \theta e_{t-1}$$
   $$e_0 = 0, Y_0 = \mu_0, e_t \sim N(0, \sigma).$$

   Therefore we get a recursive formula for the errors. Define $Y^t_{stationary} = Y_t - Y_{t-1}$. We get $e_t = \sum_{i=0}^{t-1} (Y^{t-i}_{stationary} - d)\theta^i$.

   (c) Therefore the log likelihood function is:

   $$L = \sum_{i=1}^{N_0} \ln f_{e_t|e_{t-1}}(Y^t_t = \text{drift} + e_t - \theta e_{t-1} \mid e_{t-1}) =$$
   $$= \sum_{i=1}^{N_0} (-\ln \frac{1}{\sqrt{2\pi}\sigma} exp(-e_t^2/(2\sigma^2)))$$

   but as I said, the optimization algorithm doesn't maximize L and instead makes it NaN for some reason. I didn't manage to figure out why.

   (d) Probability to observe the last 6 data points: I'm not sure what you meant by probability to observe the testing data. I've implemented a function *probabilityfuture(observations)* which computes the prediction errors and the value of PDF of the observed error.

2. I would predict the 6 data points in the middle twice: once using the first 7 data points in a regular fashion to get a prediction $(p_0, ..., p_5)$, and once using the last 7 data points backward to get a prediction $(p'_0, ..., p'_5)$. The prediction would be $\tilde{p}_i = ((5-i)p_i + ip'_i)/5$, an average of the two predictions, since we are more confident in the prediction the closer we are to the last data point.