```
#The Processing part of the Console application is done using R programming language. All the algorithms used
in the script are written in R Version 3.4.0 and R studio Version 1.0.143.
#The libraries/packages that need to be preinstalled for the smooth running of the script are "TTR" and "forecast".
#The required text files needs to be preset before triggering the R script, from the C# code.

Following is the complete R Script code that is used to handle the mathematical and statistical processing of
the model. The section wise description and logic of the code is mentioned along with it.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Script Initialization*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
args <- commandArgs()
headFilePath <- args[2]
tailFilePath <- args[3]
outputFilePath <- args[4]
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* The above part of the script is declared for the initialization of the R Script inputs. These parameters of
the script are supplied by the C# code, which handles the front end of the console application. The parameters
are:

headFilePath : It contains the path for the text file which contains historical data for the L2 head cut set
points.
tailFilePath : It contains the path for the text file which contains historical data for the L2 tail cut set
points.
outputFilePath : It contains the path for the text file which stores the pre-final values, i.e. the forecasted
value for the head and tail cut set point.

All these are passed using the command line to the R Engine. The detailed function for doing this is in the C#
code file.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Importing Libraries*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
library("TTR")
library("forecast")
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* The above are inbuilt R libraries that are used for standard operations in R Script. They mainly handle the
```

```
plotting part and the functions related to the Time Series manipulation and Forecasting.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Scanning and Forming the Time Series*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
# SCANNING THE PREVIOUS VALUES OF HEAD AND TAIL SETPOINTS
gd_h <- scan(headFilePath, skip=1);
gd_t <- scan(tailFilePath, skip=1);

# FORMULATION OF THE TIME SERIES
gd_hts <- ts(gd_h);
gd_tts <- ts(gd_t);
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* This section of code scans the data of the historical head and tail data of the set points  for the particular
grade into the R environment. It then forms their corresponding Discrete Time Series for further analysis.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Plotting of Time Series*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
#PLOTTING THE TIME SERIES CHARTS AND SAVING IT
png(file = "C:/Users/Dewal Agarwal/Desktop/Data/DA/R_TS/Console_AF/gd_hts_plot.jpg");
plot(gd_hts, type="o", col= "red", main = "Time Series Head");
dev.off();
png(file = "C:/Users/Dewal Agarwal/Desktop/Data/DA/R_TS/Console_AF/gd_tts_plot.jpg");
plot(gd_tts, type="o", col= "red", main = "Time Series Tail");
dev.off();
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* This section helps in plotting the time series of the input historical data and exporting it directly to a
JPEG image.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
#[OP] CALCULATION OF MOVING AVERAGES
# gd_htsSMA5 <- SMA(gd_hts, n=5);
# gd_ttsSMA5 <- SMA(gd_tts, n=5);

# HEAD & TAIL SES smoothing
# gd_hSES <- HoltWinters(gd_hts, beta =  FALSE, gamma = FALSE)
# gd_tSES <- HoltWinters(gd_tts, beta = FALSE, gamma = FALSE)
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Double Exponential Smoothing*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
# HEAD & TAIL DES smoothing
gd_hDES <- HoltWinters(gd_hts, gamma = FALSE, l.start = gd_h[1], b.start = gd_h[2]-gd_h[1]);
gd_tDES <- HoltWinters(gd_tts, gamma = FALSE, l.start = gd_t[1], b.start = gd_t[2]-gd_t[1]);

# [op] Printing the value of the smooth parameters.
# print(gd_hDES);
# print(gd_tDES);
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* This section is the main part of the mathematical model, it does the following things:
    i) It uses Holt-Winters Algorithm to calculate the fitted value alongside the actual data points.
    ii) It calculates the parameters using the " -------"  Algorithm, which minimization of the SSE as the basic
principle.
    iii) The further details related to the main documentation are the part of the main documentation.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Plotting of Fitted Values*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
# PLOTTING INPUT V/S THE FITTED VALUES
png(file = "C:/Users/Dewal Agarwal/Desktop/Data/DA/R_TS/Console_AF/gd_hDES_plot.jpg");
plot(gd_hDES, type="o", main = "Observed & Fitted Head");
dev.off();
png(file = "C:/Users/Dewal Agarwal/Desktop/Data/DA/R_TS/Console_AF/gd_tDES_plot.jpg");
plot(gd_tDES, type="o", main = "Observed & Fitted Tail");
dev.off();
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* This section of code is meant for plotting the head and tail time series along with their corresponding fitted
values.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
Forecasting of Set points*
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
```

```
# FORECASTING THE NEXT 'h' VALUES FOR THE HEAD AND THE TAIL
gd_hDESfc = forecast.HoltWinters(gd_hDES, h=1);
gd_tDESfc = forecast.HoltWinters(gd_tDES, h=1);

# [op] Print the forecast for the head and tail
# print("The Head setpoint is");
# print(gd_hDESfc)
# print("The Tail setpoint is");
# print(gd_tDESfc)

forecastH <- as.numeric(gd_tDESfc$mean)
forecastT <- as.numeric(gd_hDESfc$mean)

# OUTPUTTING THE FORECASTED VALUE IN THE TEXT FILE
sink(outputFilePath)
print(forecastH)
print(forecastT)
sink()
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
* This is the final section of the R Script that deals with the forecasting of the next set point based upon
the fitted values and the different smoothing parameters. Here it is calculating the next value for the head
and tail cut set points. It stores them in a text file for further manipulation by the C# code. The mathematical
expressions involved are the part of the main documentation.
-----------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------
## Functions for adding the bias.
#Final_SPH <- function(L2Hinput)
#{ sp = (forecastH + L2Hinput)/2
# There should be a provision for changing the entry from the exixting data base.
#}
#print("Displaying the final setpoint with offset");
#print(Final_SPH(500))

##Final_SPT <- function(L2Tinput)
#{ sp = (forecastT + L2Tinput)/2
# There should be a provision for changing the entry from the exixting data base.
#}
#print("Displaying the final setpoint with offset");
#print(Final_SPT(500))
-----------------------------------------------------------------------------------------------------------
```