HOME ASSIGNMENT – LANGUAGE: JAVA

Write a converter that can take a String representing a game match time in one format, and convert it to a String representing the match time in another format.

The input match time is in the format

```
[period] minutes:seconds.milliseconds
```

The output should be in the format

```
normalTimeMinutes:normalTimeSeconds – period
```

For the output format, minutes should be padded to two digits and milliseconds should be rounded up or down to the nearest whole second. Periods are represented in short-form on the input format and long-form on output format i.e.

Short form	*Long Form*
PM	PRE_MATCH
H1	FIRST_HALF
HT	HALF_TIME
H2	SECOND_HALF
FT	FULL_TIME

When a given period goes into additional time (i.e. > 45:00.000 for first half, > 90.00.000 for the second half), the added minutes and seconds are represented separately in the format

```
normalTimeMinutes:normalTimeSeconds +additionalMinutes:additionalSeconds – period
```

Any input which does not meet the required input format should result in an output of INVALID

Examples

Input	*Expected Output*
“[PM] 0:00.000”	"00:00 – PRE_MATCH"
“[H1] 0:15.025"	"00:15 – FIRST_HALF"
“[H1] 3:07.513"	"03:08 – FIRST_HALF"
“[H1] 45:00.001"	"45:00 +00:00 – FIRST_HALF"
“[H1] 46:15.752"	"45:00 +01:16 – FIRST_HALF"
“[HT] 45:00.000"	"45:00 – HALF_TIME"
“[H2] 45:00.500"	"45:01 – SECOND_HALF"
“[H2] 90:00.908"	"90:00 +00:01 – SECOND_HALF"
“[FT] 90:00.000"	"90:00 +00:00 – FULL_TIME"
“90:00”	“INVALID”
“[H3] 90:00.000"	"INVALID"
“[PM] -10:00.000"	"INVALID"
“FOO”	"INVALID"

Additional instructions:
-Main package should be com.example
-Use Maven
-The code should be written to production standard
-The project should be runnable from both test units and a main stub
-Publish your solution to a github repository
-Provide instructions (README) on how your solution should be run.