1. A MapReduce program was executed on the provided Saavn data, to produce the score of each <songId,day> pair.

2. Following command was used to execute map-reduce program:

```
hadoop jar saavntrend.jar com.upgrad.trendproject.SaavnTrend s3a://mapreduce-project-bde/part-00000 s3a://saavn-output-sagar/output18
```

3. In Mapper, Exponential decay algorithm is used to calculate score of each <songId,day> pair.
   1. Formula: $(1-c)^r$, where c=0.05 and r=23 - hour_of_song_playing
   2. Window Size: 1 day
   3. Result: <<songId, day>, score>

4. Map output is passed to combiner and then combiner adds the score of key <songId, day> and passed out put to partitioner.

5. Partitioner sends the data to respective reducer based on date.

6. Reducer calculates the total score by adding score of each <song_id, date> pair and writes to output file.

7. Output of MapReduce program is <<songId, day>, score> in which data is unsorted.

8. Following Unix Commands were used to sort the output:
   sort -rg -k3,3 <mr_output_filename> | head -100 | cut -d' ' -f1 > <output_file_path>

   For Example,

```
sort -rg -k3,3 part-r-00000 | head -100 | cut -d' ' -f1 > ~/Desktop/25.txt
```

9. Following is description of above commands:
   1. 'sort' command -> Sort the file part-r-00000 data in reverse order based on column 3, which is decay_score.
      1. 'r' option -> reverse order
      2. 'g'option -> general numeric to convert parse column 3 as float
      3. 'k' option -> specifies the column based on which data is to be sorted.
   2. '|' operator -> passes output of one command to the next command.
   3. 'head' command -> Take top 100 rows
   4. 'cut' command -> picks only columns of songId and removes other columns
      1. 'd' -> specifies delimiter which is our case is ' '(space_bar).
      2. 'f' -> Field number i.e. column 1 (which is songId)
   5. '>' -> redirection operator, which writes the output of command to specified file