Executing the script:
Run "./run.sh [location path] [recording path] [output path] "

Detailed Description of Project:

1. **Reading The Datasets:**
   There are two types of datasets to consider, First subproblem of the project is to load the dataset into spark data frame.
   - Read the locations dataset and filter it by reading only the CTRY value to US and where STATE value is not null
   - A view called "onlyUS_view" is created so we can use this as a table
   - **Read the Recordings dataset:** These text files are little bit tricky because there are variable number of separators. It may have one whitespace ,two or three.
     - First while reading, filter the text file where value does not start with "STN—-"
     - Replace multiple whitespace with a single whitespace using Sparks function "withColumn"
     - Split the columns into USAF, WBAN, MONTH, PRCP and remove the rest of the columns because these are the only ones we need.
     - Combine all the text file data frames into one using "union" of spark
2. **Join both data frames:**
   One dataset has the "STATE" column we need and another data frame contains the "PRCP" value we need. Both data frames have one column which is common between them which is "USAF". So we join these two data frames by condition USAF has to be equal.
3. **Updating the PRCP values:**
   The PRCP values have a float digit and a letter ('A', 'B',…) associated with it. According to rules, if any PRCP value contains 'A' we multiply the digit by 6, if 'B' multiply by 2, if 'C' multiply b y 1.33 etc.
   - To multiply and update the PRCP column, we use Sparks "withColumn" and "when" functions where we can update a columns value according to a given condition.
   - A new view is created called "joined_view".
4. **Calculating average PRCP group by month:**
   We use a sparkSQL query to calculate average prcp value group by month and state. So now we have a view called "groupMonth_view" which have the average PRCP **for each month for each state**.
5. **Finding month with Minimum and Maximum average:**
   To find the highest avg value and its month for each state, we need to first find the MAX(avg(prcp)) value from "groupMonth_view" and then JOIN with itself to find which month has the highest value and retrieve its state.
   - To find the lowest avg same query is executed again with MIN.
   - Two views are created "maxMonth_view" and "minMonth_view"
   - These two views are then joined to have one single table with both highest month and lowest month
6. **Finding the difference:**
   A sql query is used to calculate the difference between highest avg precipitation value and lowest avg prcp value and is ordered by ascending difference and group by STATE.
7. **Getting month name:**
   A *substr* is used to retrieve the Month value in digits from YEARMODA. Then *withColumn* function is used to get the month in alphabets from the digits.
8. **Outputting result:**
   coalesce is used to merge all the partitions into a single csv file and the csv file is written into OUTPUT path.

Runtime of Project: The total runtime of the project is **4 mins 51 secs**

BONUS: To do the bonus part, I added the states with "NULL" value into my calculation. I am approximating the states from their spatial distance. Details is described in the following:

- I am using a dataset which have the columns "STATE" "LATITUDE" "LONGITUDE" from kaggle. In this datset, we can get the center latitude and longitude for every state in USA. The dataset can be found here: (the datasets csv file is hardcoded in the code and the csv file is included in the zip file)
  https://www.kaggle.com/washimahmed/usa-latlong-for-state-abbreviations/data#
- From the LOCATIONS dataset, I am filtering out the states which have NULL value and LAT and LON value is NOT NULL
- I CROSS JOIN the locations dataset with the kaggle dataset using spark dataframe. In the kaggle dataset I have 51 rows. From the cross joined result the columns are the following:
  STATE(locations) | USAF(locations) | LAT(locations) | LON(locations) | STATE(kaggle) | LATITUDE(kaggle) | LONGITUDE(kaggle)
- From research I find out that the average spatial distance in latitudes from the center of a state is between 0 - 1.5 and average spatial distance in longitudes from the center of a state is between 0 - 4.
- So from the cross joined result, if the absolute difference in latitudes(locations - kaggle) is between 0-1.5 AND absolute difference in longitudes(locations - kaggle) is between 0-4 I UPDATE the STATE(locations) value with STATE(kaggle) value. In summary, if lat and lon is within a spatial distance of a state, then i am including that row in that state.
- From 24k states with null value, I was able to approximate 22k states.
- These 22k new rows are UNIONed with the original locations table which have STATE=NOT NULL