IE 343 PROJECT REPORT

BERİL KARAGÖL S021329 - DİLARA SARITAŞ S021412

We initially had three different algorithm ideas in order to solve the first problem. We decided to compare the results of these three algorithms, and use the one that would give us the highest album value. In this report, all of those algorithms will be explained and the one that gives us the highest value will be used to solve the problem. Our approach for programming the said algorithm will also be explained in detail. Then, our results and the result from the “exact\_method.ipynb” file will be compared. Finally, our algorithm for the second problem along with our programming approach will be explained in detail.

For the first question, our first idea was to sort the popularity values of each track from highest to lowest. We exported the data to Excel to work on it. After sorting the popularity values from highest to lowest, we selected all of the top values until their duration reached 30 minutes. Since the track durations were given in milliseconds, we converted 30 minutes into 1800000 milliseconds. Firstly, we added up the first 8 tracks’ duration values, which was equal to 1717084 ms. But when we added the ninth duration value as well, the total duration became 19996025 ms. Which is why we did not include the ninth track, because if we did, the total duration would be over 30 minutes. Seeing that we were under 30 minutes we looked for tracks to add to the album that would not exceed the time limit. If we subtract 1717084 ms from 1800000 ms, we can see that we have 82916 ms to use. Then, we started looking for track durations under 82916 ms to add to the album. The closest duration value to 82916 was in the 22nd row, which was 72000 ms. Then we added 1717084 ms to 72000 ms. The total duration of the album became 1789084 ms. There were no other songs to add to our total duration value without exceeding our time limit. Once our tracks were selected, the total value of the album became 369. Then, we moved on to calculating the cost of being far from 30 minutes. Firstly, we converted 1789084 milliseconds to 1789.084 seconds. Then, we subtracted 1789.084 seconds from 1800 seconds, and found 10.916 seconds. The cost of each missing second decreases the album value by 0.02. For this case, since our album lacks 10.916 seconds, the album value decreased by 0.21832. When we subtracted this cost from our album value, we reached a total value of 368.7817.

              Our second idea was to duration values of each track from lowest to highest. After sorting, we selected all the values from the top until, once again, their duration reached 30 minutes. Then, we added up the first 19 tracks’s duration values, which was equal to 1773722 ms. If we were to add the twentieth value as well, our total duration would be 1935722 ms. It would exceed our time limit, so we did not include it. If we subtract 1773722 ms from 1800000 ms, we can see that we have 26278 ms left in the album. Then we started looking for track durations under 26278 ms to add to the album. There are no tracks which have a duration time less than 26278 ms. Therefore, we did not add any other tracks to the album. Our final album included tracks 25, 48, 43, 33, 27, 37, 32, 35, 39, 26, 47, 41, 42, 46, 49, 0, 15, 8 and 16. The value of the album, with these tracks included, became 508. Then, we moved on to calculating the cost of being far from 30 minutes. The album lacked 26.278 seconds. The cost of each missing second decreases the album value by 0.02. When we multiplied 26.278 with 0.02, we got 0.52556 as the cost. Finally, we subtracted 0.52556 from our value, which is 508, we found the final total value as 507.4744.

Our third and final idea was to find the value per millisecond for each track. To do that, we calculated the “popularity/duration” of each track. Then, we sorted them from highest to lowest. Once sorted, the duration of the top 17 songs was 1765171 ms. We subtracted 1765171 ms from 1800000 ms, to see how much ms we have left to use on our album. Seeing that we have 34829 ms left, we looked for tracks which have shorter duration times than that number. There are no such tracks, so we did not add any more tracks to the album. We left the album at 17 tracks and 1765171 ms. The value of all those tracks added up to 530. However, we need to calculate the cost of being far from 30 minutes, and subtract that from 530. 34829 ms equals 34.829 minutes. Each minute left unused for the album costs 0.02. We calculated 34.829\*0.02=0.69685 to find the cost. Finally, we subtracted the cost from our value and found our final value, which is 530-0.69685=529.3034. This algorithm gives us the highest value, hence why we will be using it for the first question.

To start with coding, we created an array list called sortedList, which sorted our popularity per duration for each track, as explained above. We added up the popularity and the duration values in a for loop. We kept our totallength value less than 30 minutes, so we knew where to stop adding songs to the album. We created another array list called allSongs, to add all those songs into the album we created. In the end, we printed out the final total value after reducing the calculated cost from our value.

In addition, when we compare our algorithm with the exact solution in the “exact\_method.ipynb” file, which is solved by Gurobi, we see that the final total value of the exact solution is 532. The final total value in our algorithm was 529.3. Although the final total value of the exact solution is more than the value of the 3rd algorithm, we see that the results are quite close. Therefore, our algorithm yields a highly usable result.

For the second part of the project, as stated on the project manual, we set the first track as Track #0, which is called “Back Home''. That is because it is the most popular song in the album we created. The second most popular track on the album was Track #11, which is called “Todo Colores”. We set this song as the last song. To put the tracks in an optimal order, we decided to use Greedy Algorithm. As we set track #0 as the first one, we looked for the track that the listeners enjoyed the most after listening track #0. The highest rate was 7.07, when focus group listened to Track #6 after listening Track #0. Therefore, we decided to set Track #6 as the second track of the album. Again, for Track #6, the highest rate was 7.28, when focus group listened to Track #27 after it. Which is why we decided to set Track #27 as the third song on the album. The process went on like this, until the last song. Unfortunately, we could not achieve what we actually wanted to achieve, as we had some issues while coding. Coding approach will be explained below.

Firstly, we created a 2D array which included the track id and the sequential data called songOrder. We set the first song and the last song. Then, as I explained above, we chose the next song according to Greedy Algorithm with a for loop. If that song had been chosen before, we would not choose that song. Also, we paid attention to choose a song that we had chosen for our album from the first part of the project. We added such songs to songOrder. Finally, we printed out the songs in order.