Course Project - Café Exploration

9th March 2022
Algorithms

Abhisar Sen
AU20B1010

Content –

INSTRUCTIONS

One must select any system for which they are expected to design and implement their own algorithms and then analyze the time and space complexity. The system selected for this project must be discussed in detail with the instructor before starting the implementation.

The project is consisted of following stages-

1. Problem Identification: One should identify and finalize the problem. Upon finalization, the problem has to be documented in detail with its algorithmic objective clearly explained. Also, list out the different constraints in the problem which the algorithm needs to satisfy.

2. Design: This stage includes, Algorithm steps and procedure, Pseudo Code, and Flow chart.

3. Implementation: Implement your algorithm in C++.

4. Testing: Test your algorithms for all the constraints mentioned in the problem and put the screenshots of the results.

5. Results: Show the time and space complexity analysis of your algorithm and discuss it in detail. Each and every aspect of the algorithm should be focused on the results.

Problem Identification

The problem is all about the people who always gets confused about how to proceed for selecting any café or is dish in order of having some delicious item.

People are often confused about which Café should they visit so that they get a proper food with max taste and min bill.

There is no such system in which a user can see that a particular Café is Famous for what dish or which place can suits to be the best for fun.

Problem Deconstruction –

What →

- Usually, it is the problem with most of the people, that they are not much sure about which café should they try or if mistakenly they got success in selecting the café then another problem of choosing the best dish to try spoils up the whole mood.

How →

- When people are frustrated of hunger and there is no way to decided whether which new place should they try or if they are successful in picking up the best place to have fun, then which delicious treat can make their fun doubled.

Why →

- As there are many people around us who are very much find of trying out new cafes or dishes so that –
  - They can feel fresh.
  - Get help for betterment of their mood
  - They can have a good meal
  - They can enjoy their happy moments

Constraints And Challenges –

- User should be able to find out the best possible places to hang out.
- Ratings of the dishes should be mentioned clearly with the corresponding dishes.
- The solution should be neat and clean so that user need not to bear any type of frustration in terms of its experience.
- If user is not interested in rating that dish then he should not be forced to rate that dish.
- User should get the facilities of searching the cafes of their choice so that they can choose up the best dish to try.
- Café manager can't input more than 5 dishes.
- User Should get suggestion on the basis of his input budget for the best dishes of that particular café that he selected.

Design

Algorithm –

```
Struct Details {
        CafeName
        DishName
        Rating
        Price
        Quantity
}
```

This Algo is for Searching any dish using its name, it will take the name of that dish as an input and then return its price, quantity and rating in different cafes.

```
SearchByDish (Dish){
        Details arr[n];
        while (endOfFile){
                read {CafeName, DishName, Rating, Price, Quantity}
                if(file.DishName == Dish){
                        write DishDetails
}
}
}
```

This algo is for returning different dish with all of its parameters like quantity, price and rating so that user get to know that which dish is better in that particular café. This algo also asks from the user whether he wants the suggestions of dishes based on his budget so that he need not waste his time in searching of dishes from that café menu instead directly order those special dishes and starts enjoying them instantaneously.

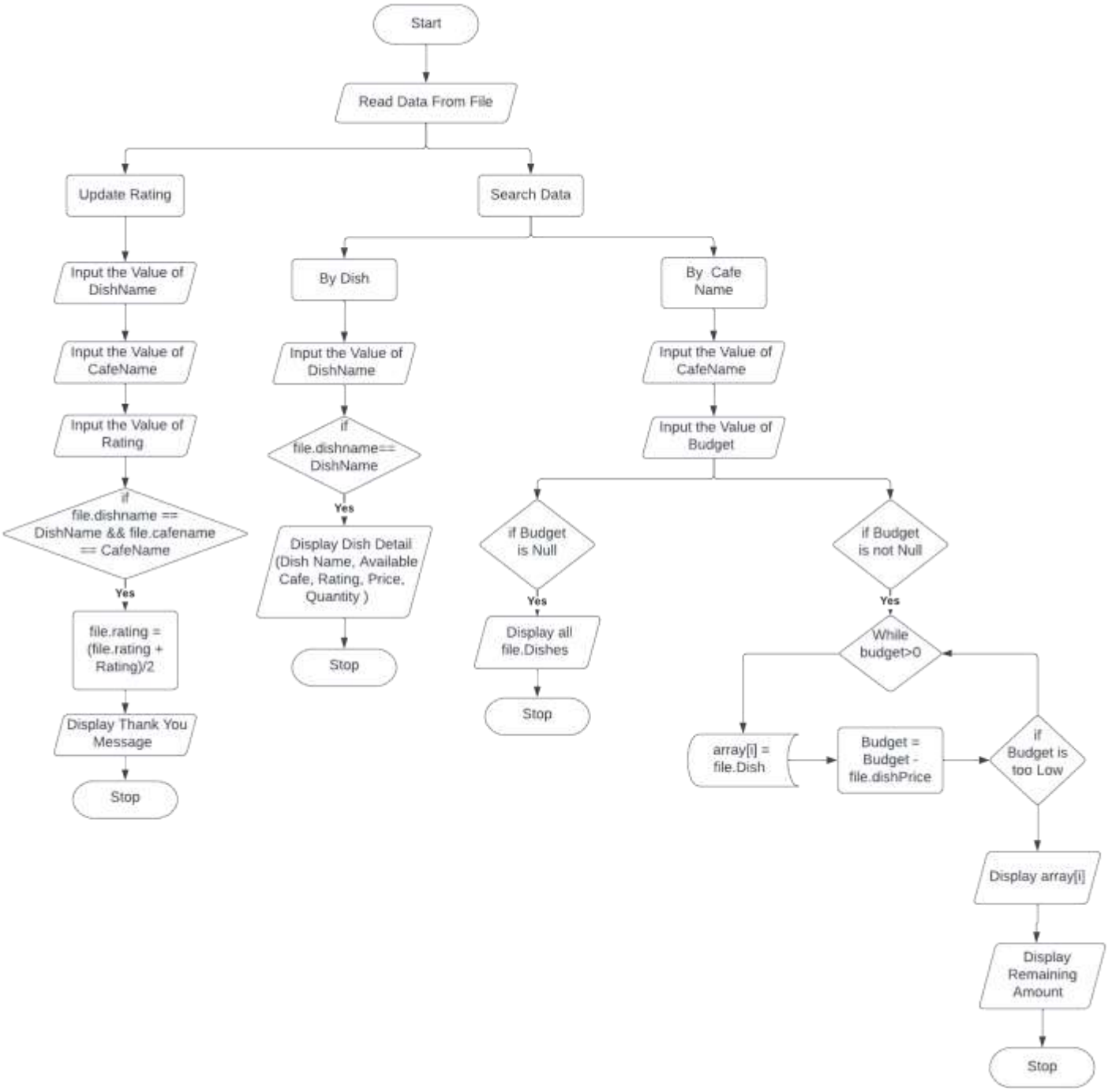
```
SearchByCafe(Budget, CafeName, Suggestion[]){
  for(i=0 to endOfFile){
    read from file ->
    file.CafeName
    file.Price
    file.Quantity
    if(file.CafeName == CafeName){
      arr[] = file.Price(i)/file.Quantity(i);
    }
  }

  Sort(arr);

  if(Budget == 0){
    for (i=0 to endOfFile){
      read Cafe Details ->
      file.CafeName

      if(file.CafeName == CafeName){
        write file.Details;
```

```
        }
      }
    }
    else{
      for(i=0 to endOfFile){
        if(Budget>0 && file.Price<=Budget){
          Suggestion[k] = file.dish[i]
          k++
          Budget = Budget - file.price;
          bill = bill+file.price[i];
        }
        else if(budget > 0){
          temp = (file.price[i]/file.Quantity[i])*budget;
          Suggestion[k] = dish;
          k++
          bill = bill+temp*file.price;
          break;
        }
      }
    }
    for(i = 0 to n){
      write Suggestion[i]
    }
}
```

This algo asks the user about the deliciousness of a particular dish that he recently tried and then update the average of previous database rating and the new rating.

```
UpdateRating (CafeName, dish, Rate){
    while(endOfFile){
      read data from file
      if(file.dish && file.CafeName){
        file.Rate = (file.rate + Rate)/2
        overwrite in file -> file.rate;
      }
    }
}
```

Flowchart –


Start
Read Data From File
Update Rating
Search Data
Input the Value of DishName
Input the Value of CafeName
Input the Value of Rating
if file.dishname == DishName && file.cafename == CafeName
Yes
file.rating = (file.rating + Rating)/2
Display Thank You Message
Stop
By Dish
Input the Value of DishName
if file.dishname== DishName
Yes
Display Dish Detail (Dish Name, Available Cafe, Rating, Price, Quantity)
Stop
By Cafe Name
Input the Value of CafeName
Input the Value of Budget
if Budget is Null
Yes
Display all file.Dishes
Stop
if Budget is not Null
Yes
While budget>0
array[i] = file.Dish
Budget = Budget - file.dishPrice
if Budget is too Low
Display array[i]
Display Remaining Amount
Stop

Implementation

Code -

This is the implemented code for Searching any dish using its name, it will take the name of that dish as an input and then return its price, quantity and rating from different cafes.

```
void Search_By_Dish(string DishName)
{
   Details CafeDetails[20];
   int i = 0;
   string ar[5];
   string b;
   fstream in("cafe.txt", ios::out | ios::in);
   while (!in.eof())
   {

      in >> CafeDetails[i].CafeName;
      in >> CafeDetails[i].dish;
      in >> CafeDetails[i].rating;
      in >> CafeDetails[i].price;
      in >> CafeDetails[i].quantity;

      if (CafeDetails[i].dish == DishName)
      {
         Dash(105);
         cout << "|\tCafe Name\t|\tDish\t|\tRating\t|\tPrice\t|\tQuantity (In Plates)\t|" << endl;

         cout << "|\t" << CafeDetails[i].CafeName << "\t\t";
         cout << "|\t" << CafeDetails[i].dish << "\t";
         cout << "|\t" << CafeDetails[i].rating << "\t";
         cout << "|\t" << CafeDetails[i].price << "\t";
         cout << "|\t\t" << CafeDetails[i].quantity << "\t\t|";
         cout << endl;
         Dash(105);
      }
      i++;
   }
}
```

This is an implemented code in c++ for returning different dish with all of its parameters like quantity, price and rating so that user get to know that which dish is better in that particular café. This code also asks from the user whether he wants the suggestions of dishes based on his budget so that he need not waste his time in searching of dishes from that café menu instead directly order those special dishes and starts enjoying them instantaneously.

```
void SearchByCafe(string CafName, int budget = 0)
{
   string Suggested[5];
   float pw[5];
   int Bill = 0;
   int k = 0;
   int Profit = 0;
   Details CafeDetails[20];
   ifstream in("cafe.txt");
   for (int i = 0; i = !in.eof(); i++)
   {

      in >> CafeDetails[i].CafeName;
      if (CafeDetails[i].CafeName == CafName && budget == 0)
      {
         Dash(105);
         cout << "|\tCafe Name\t|\tDish\t|\tRating\t|\tPrice\t|\tQuantity (In Plates)\t|" << endl;

         in >> CafeDetails[i].dish;
         cout << CafeDetails[i].dish << " ";
         in >> CafeDetails[i].rating;
         cout << CafeDetails[i].rating << " ";
         in >> CafeDetails[i].price;
         cout << CafeDetails[i].price << " ";
         in >> CafeDetails[i].quantity;
         cout << CafeDetails[i].quantity << " ";
         pw[p] = CafeDetails[i].price / CafeDetails[i].quantity;
         p++;
      }
      cout << endl;
      cout << endl;
   }
   in.close();
   merge_sort(pw, CafeDetails, 0, 5);

   if (budget != 0)
   {
      Details sug[50];
      ifstream inf("cafe.txt");
      for (int u = 0; u < 200; u++)
      {
```

```
            inf >> sug[u].CafeName;
            inf >> sug[u].dish;
            inf >> sug[u].rating;
            inf >> sug[u].price;
            inf >> sug[u].quantity;

            if (budget > 0 && sug[u].price <= budget)
            {
               Suggested[k] = sug[u].dish;
               k++;
               budget = budget - sug[u].price;
               Bill = Bill + sug[u].price;
            }
            else if (budget > 0)
            {

               int temp = pw[u] * budget;
               Suggested[k] = sug[u].dish;
               k++;
               Bill = Bill + temp * sug[u].price;
               Profit = budget - Bill;
               break;
            }
         }
         inf.close();
         cout << "Your Suggestions are Listed Below - " << endl;
         Dash(41);
         cout << "|\tS.No.\t|\tSuggested Dish\t|" << endl;
         for (int i = 0; i < 5; i++)
         {
            cout << "|\t" << i + 1 << ". \t|\t" << Suggested[i] << "\t\t|" << endl;
         }
         Dash(41);
         cout << endl;

         cout << "And you are remained with the profit of " << Profit << endl
              << endl;
      }
   }
```

This is an implemented code in C++ which asks the user about the deliciousness of a particular dish that he recently tried and then update the average of previous database rating and the new rating.

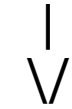
```
void UpdateRating(string cafe_name, string Dish, float rate)
{
   Details CafeDetails[5];
   int i = 0;
```

```
    string ar[5];
    string b;
    fstream in("cafe.txt", ios::out | ios::in);
    while (!in.eof())
    {

        in >> CafeDetails[i].CafeName;
        in >> CafeDetails[i].dish;
        in >> CafeDetails[i].rating;
        in >> CafeDetails[i].price;
        in >> CafeDetails[i].quantity;

        // ifstream  inf("data3.txt");

        if (CafeDetails[i].dish == Dish && CafeDetails[i].CafeName == cafe_name)
        {

            float r = (CafeDetails[i].rating + rate) / 2;
            CafeDetails[i].rating = r;
            cout << "\n\n\t!!! Thank You For Your Valuable Feedback !!!\n\t\tIt Will Help Us To
Improve";
        }
        else
        {
            cout << "!!!Error!!!\nThis Cafe Doesn't Contain This Dish";
        }
        ofstream out("data.txt");
        for (int j = 0; j < 5; j++)
        {

            out << CafeDetails[j].CafeName << " ";
            out << CafeDetails[j].dish << " ";
            out << CafeDetails[j].rating << " ";
            out << CafeDetails[j].price << " ";
            out << CafeDetails[j].quantity << " " << endl;
        }
        out.close();
        i++;
    }
    in.close();
    remove("cafe.txt");
    rename("data.txt", "cafe.txt");
}
```

Result

Output –

Program Flow



```
                        Welcome To The World of Cafe

        Please Select The Helping Option
        So That We Can Get A Chance To Help YOu-

        1. Search for the dish
        2. Search for the cafes
        3. Rate the dishes of any Cafe
        4. Exit

  Enter your Choice -
```

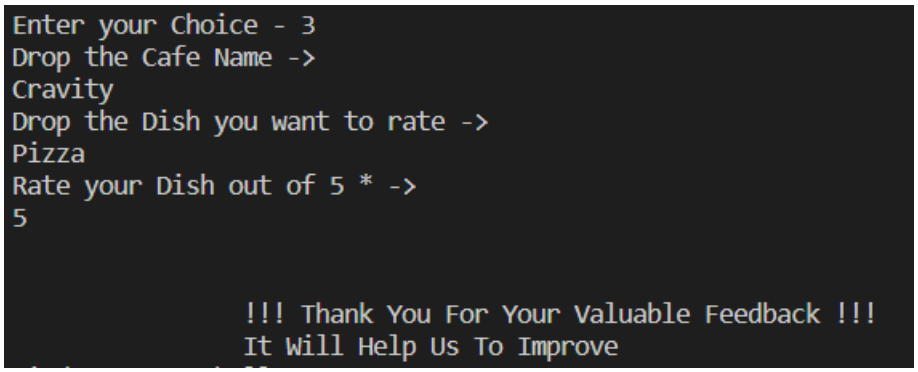
If (Option == 1)

```
Enter your Choice - 1
Drop Your Fvt Dish ->
Burger
-------------------------------------------------------------------------------------------------
|       Cafe Name       |       Dish    |       Rating  |       Price   |       Quantity (In Plates)    |
|       Cravity         |       Burger  |       3.5     |       100     |               1               |
-------------------------------------------------------------------------------------------------
```

If (Option == 2)

```
Enter your Choice - 2
Enter the Name of Your Fvt Cafe ->
Cravity
Press 1 if You want to get suggestion of dish from this cafe -
1
Enter Your Budget -> 500
Your Suggestions are Listed Below -
---------------------------------------
|       S.No.   |       Suggested Dish  |
|       1.      |       Paneer          |
|       2.      |       Pizza           |
|       3.      |       Burger          |
|       4.      |       Samosa          |
|       5.      |                       |
---------------------------------------

And you are remained with the profit of -240
```

If Option == 3


Enter your Choice - 3
Drop the Cafe Name ->
Cravity
Drop the Dish you want to rate ->
Pizza
Rate your Dish out of 5 * ->
5
!!! Thank You For Your Valuable Feedback !!!
It Will Help Us To Improve

Constraints Analysis –

Constraint	Status
1. User should be able to find out the best possible places to hang out.	Passed User is able to see all the different cafes cafés.
2. Ratings of the dishes should be mentioned clearly with the corresponding dishes.	Passed All the ratings are mentioned with each dish.
3. The solution should be neat and clean so that user need not to bear any type of frustration in terms of its experience.	Passed The complete is output is in the tabular form thus everything is neat and with proper space.
4. If user is not interested in rating that dish then he should not be forced to rate that dish.	Passed There is a different option for rating in the menu so that if user is interested in rating the dish then he can rate that one.
5. User should get the facilities of searching the cafes of their choice so that they can choose up the best dish to try.	Passed There is option “Search by café name” so that the user can search his favorite café.
6. Café manager can’t input more than 5 dishes.	Failed Currently there is no option for registering any café nor dish registration by any cafe.
7. User Should get suggestion on the basis of his input budget for the best dishes of that particular café that he selected.	Passed The Program asks the user his budget and then returns the array of suggested dishes to the user

Complexity Analysis –

Function Name	Complexity Order [T(n)+S(n)]
SearchByDish	O(n)
SearchByCafe	O(nLog(n))
UpdateRating	O(n)

Since, we have the data about which function is taking higher time or space in order of execution and also that “SearchByCafe” function is of the highest complexity order i.e., O(nLog(n)) thus the complexity of the code is the same as that of “SearchByCafe” function.

Now, even in the worst case, when **the ratio of dish’s price and quantity** is in the reversed order then also the complexity will be same only

OR

Even if the ratio is in already sorted manner then also the complexity would be the same.

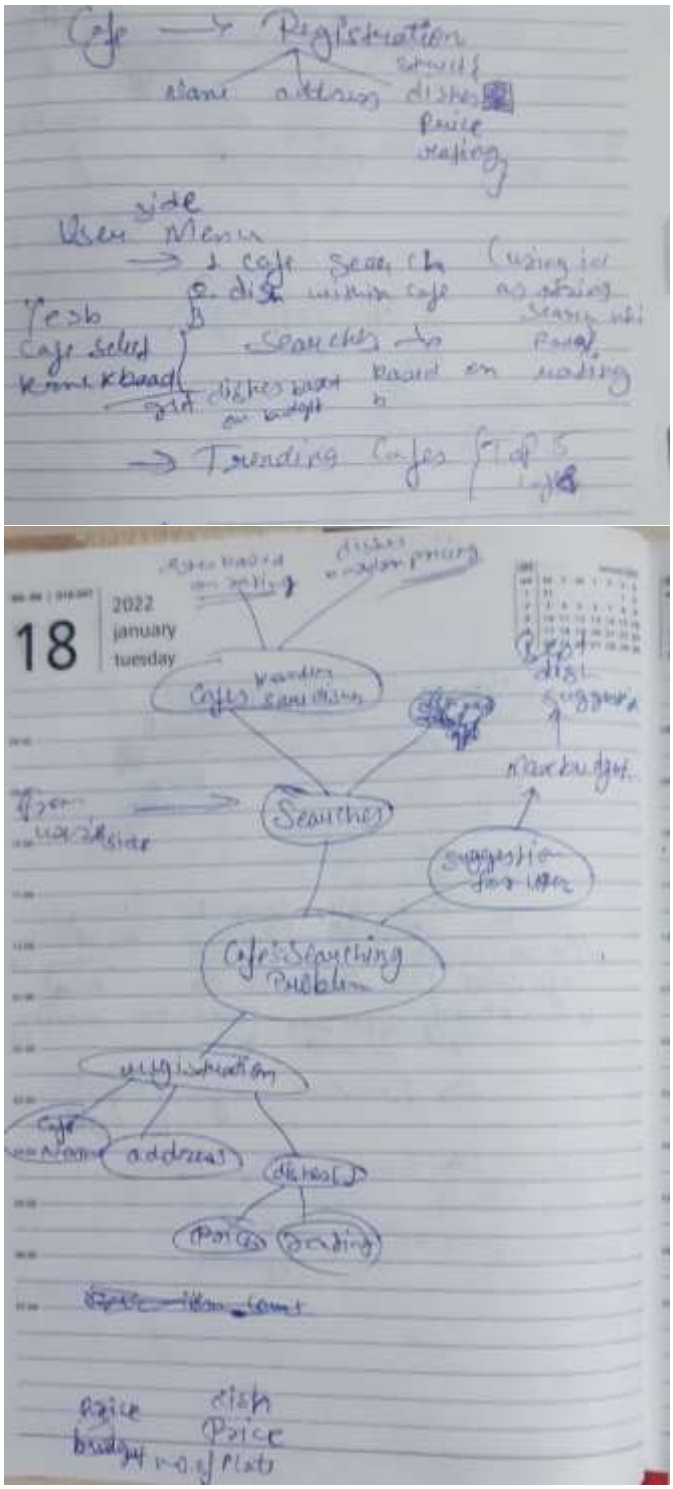
This similarity comes from the function “Sorting()” within the function “SearchByCafe” which has the highest complexity of order O(nLog(n)).

Conclusion –

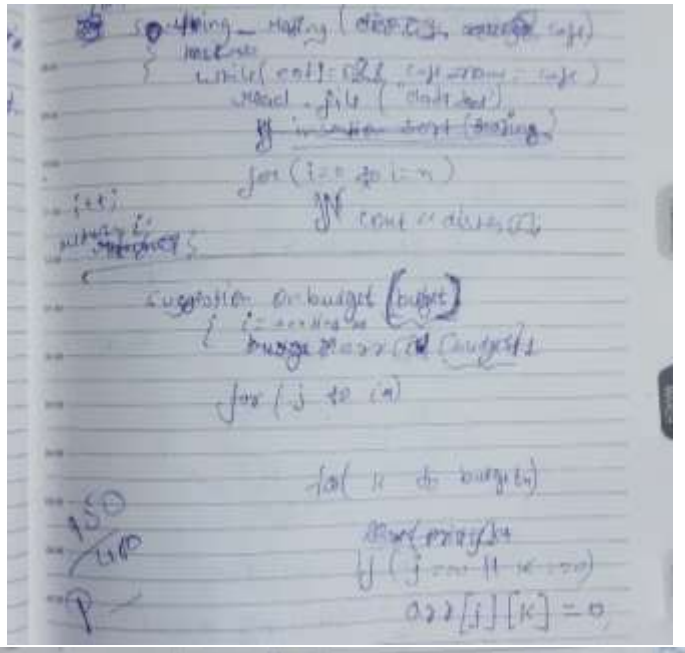
This assignment successfully solving the problem of people who gets frustrated while searching for the café and its dishes. This project first starts from understanding the problem along with the constraints and challenges and then it moved towards the designing of Algorithm and its flowchart. Then I implemented the whole things in the form of proper system which takes certain inputs and return the things as per the user's wish. At last, I analyzed the outputs by some testing parameters and the complexities (time and space) from the implemented code.

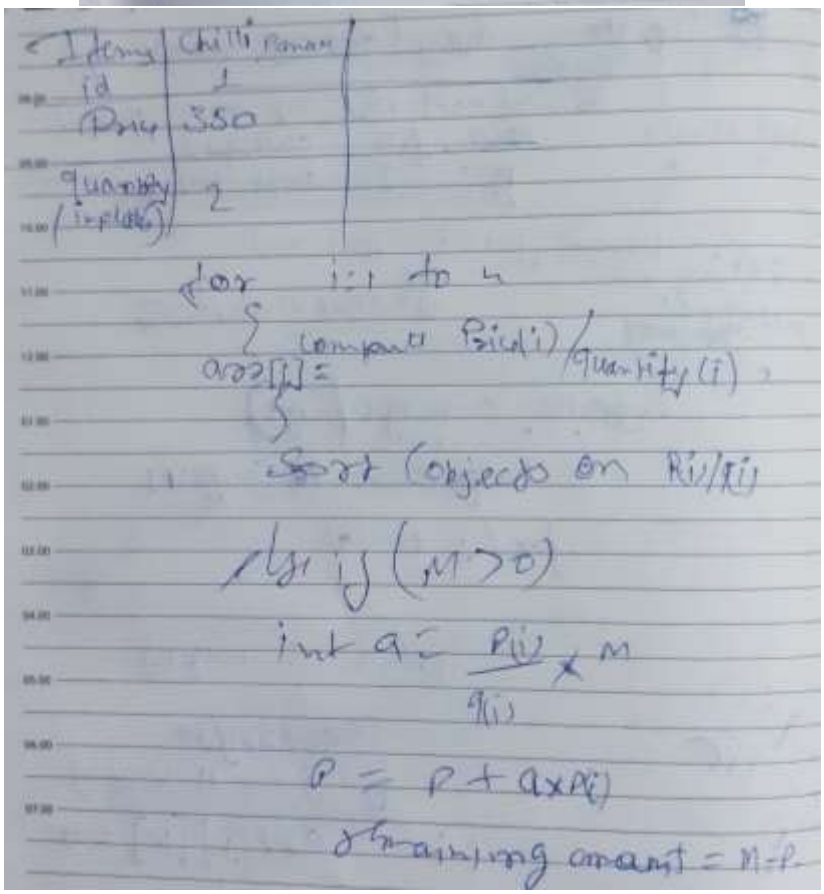
Appendix –

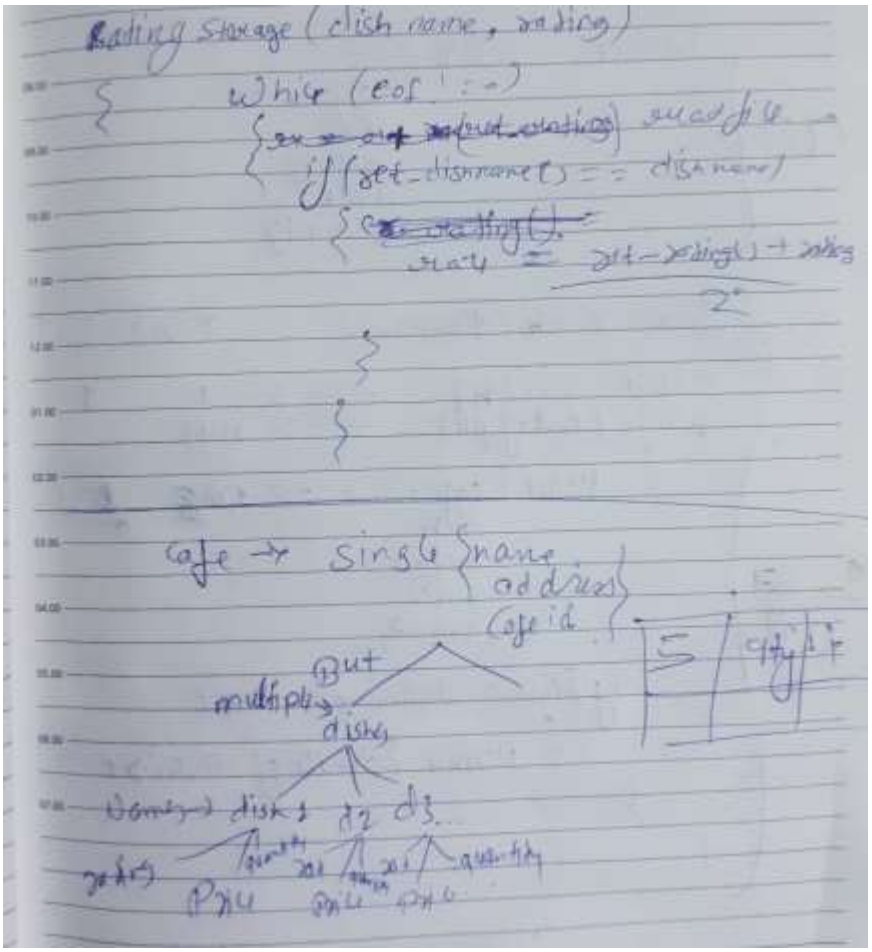
Appendix A –



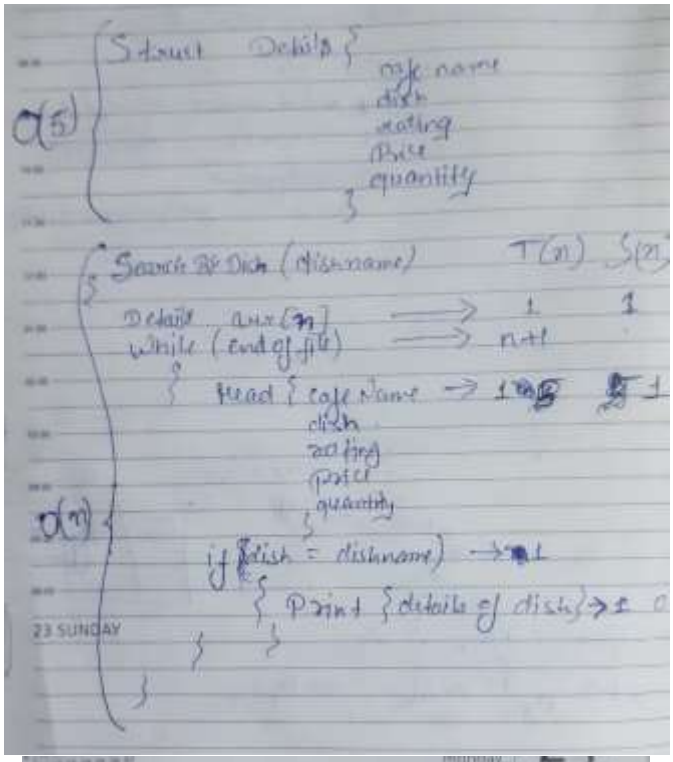
Appendix B –

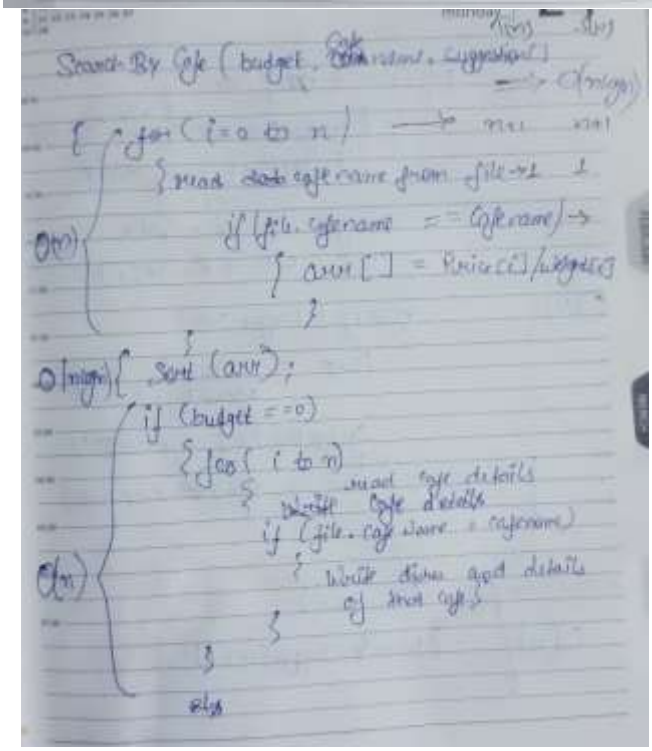


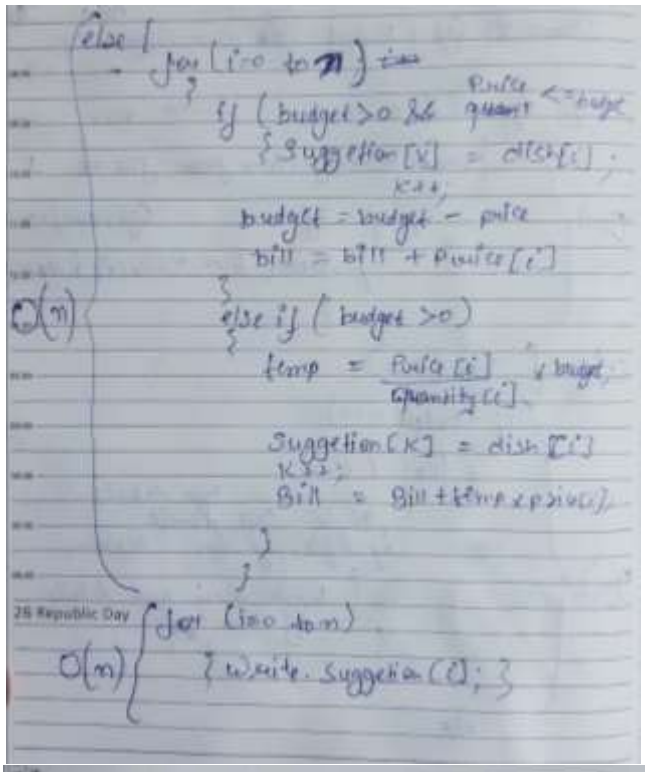

Item
Chilli Paneer
id
Price
350
quantity
(in plate)
2
for i:1 to 4
{
compute P[i]/quantity[i]
arr[i] =
}
Sort (objects on Pi/Wi)
if (M>0)
int q = P[i] / q[i] x M
P = P + q x P[i]
remaining amount = M-P

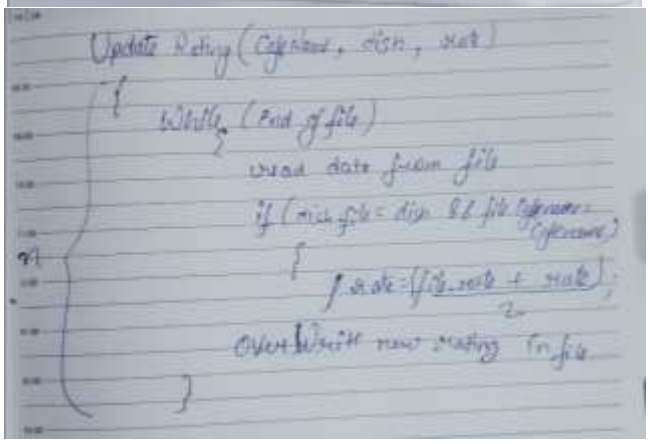

Rating storage (dish name, rating)
while (eof != -)
if (get_dishname() == dishname)
cafe → single {name
address
cafe id
But
multiple
dishes
dish1
d2
d3
price
qty

Appendix C –


Struct Details {
cafe name
dish
rating
Price
quantity
}
Search By Dish (dishname)
Details arr[n]
while (end of file)
{ read { cafe name
dish
rating
price
quantity
}
if (dish = dishname)
{ Print {details of dish}
23 SUNDAY


Search By Cafe (budget,
for (i=0 to n)
{read cafe name from file
if (file.cafename = = cafename)
arr[] = Price[i]/budget[i]
Sort (arr);
if (budget = =0)
{for(i to n)
read cafe details
if (file.cafe name = cafename)
Write down and details
of that cafe}


else {
for (i=0 to n) {
if (budget>0 && price <= budget
{ Suggestion[k] = dish[i];
k++;
budget = budget - price
bill = bill + Price[i]
}
else if (budget >0)
{
temp = Price[i] * budget
Quantity[i]
Suggestion[k] = dish[i]
k++;
Bill = Bill + temp x price[i];
}
}
O(n)
25 Republic Day
O(n) for (i=0 to n)
{ write. Suggestion[i]; }


Update Rating (Cafename, dish, rate)
{
While (End of file)
{
read data from file
if (dish_file = dish && file Cafename = Cafename)
{
new rate = (file.rate + rate)/2;
Overwrite new rating in file
}
}

Appendix D –