

MARMARA UNIVERSITY
DEPARTMENT OF
COMPUTER ENGINEERING
CSE 2025
PROJECT REPORT

-Dilara Nur Menteş 150119911

-Dilan Dilen 150119912

CSE 2025 PROJECT 1

This project simulates a loan withdrawal system, which allows customers to choose different types of loans and create installments in the payment plan. The linked lists are created using structs and pointers, and the information read from input files is kept as a linked list. Every customer has a loanptr and every loan has an installment ptr. In the main function for this program to run according to the input we receive from the user, the relevant functions are called thanks to the switch case. We continued to receive input from the user and perform operations until zero was entered.

OPTION 1 readCustomers():

The function uses the fopen function to open the file "customer.txt" in read mode. It declares variables to hold the customer's name, surname, and type, and a pointer current_customer of type customer to keep track of the last customer added to the linked list. It reads data from the file using a while loop and fscanf, creating a new customer node, assigning customer data to the corresponding fields of the new customer structure, adding the customer to the linked list, and closing the file. If the customer_list is empty, the customer_list pointer is set to point to the new customer, and current_customer is updated to point to the new customer..

OUTPUT:

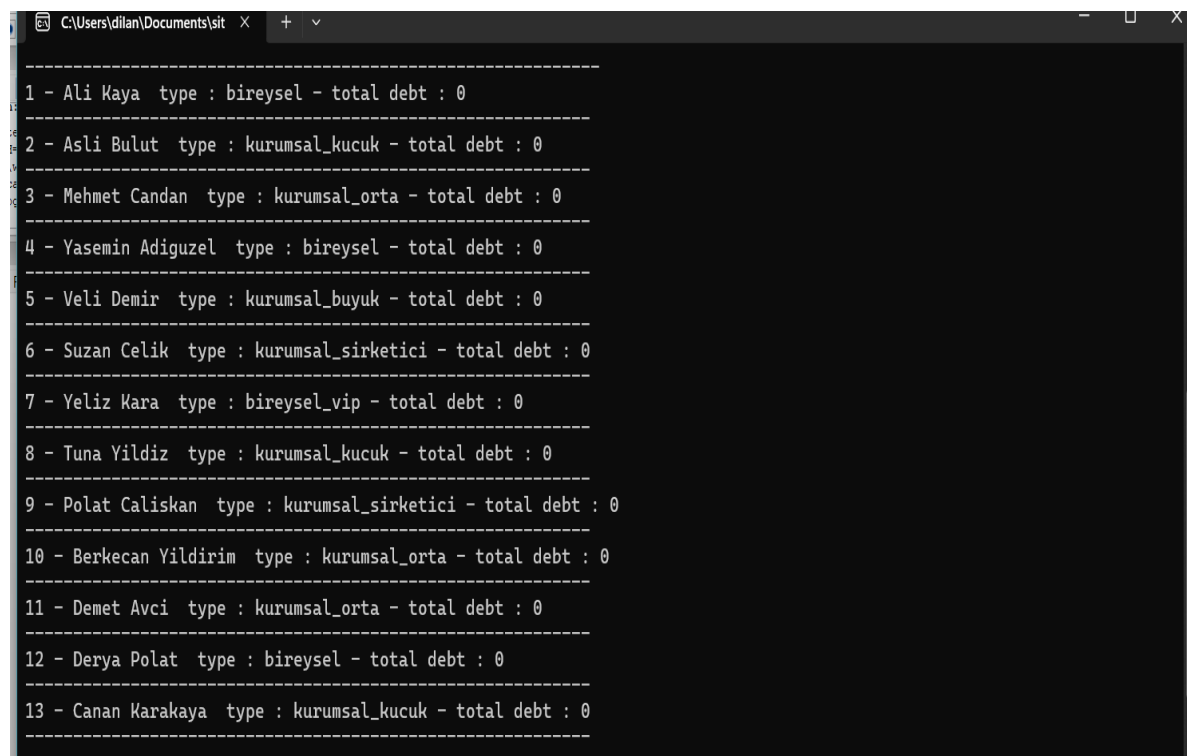
```
#####
1. read customers.
2. print customers.
3. read loans.
4. print loans.
5. create installments.
6. print installments.
7. read payments.
8. find unpaid installments.
9. delete completely paid installments.
please select your option :
1
#####

#####
1. read customers.
2. print customers.
3. read loans.
4. print loans.
5. create installments.
6. print installments.
7. read payments.
8. find unpaid installments.
9. delete completely paid installments.
please select your option :
```

OPTION 2 printCustomer():

This function prints the customer records stored in the linked list to both the console and a file using printf and fprintf functions. It traverses the linked list, prints each customer's information, and separates the output using separator lines. The printed output provides a formatted view of the customer data, including their ID, name, surname, customer type, and total debt.

OUTPUT:



```
C:\Users\dilan\Documents\sit x + v
-----
1 - Ali Kaya type : bireysel - total debt : 0
-----
2 - Asli Bulut type : kurumsal_kucuk - total debt : 0
-----
3 - Mehmet Candan type : kurumsal_orta - total debt : 0
-----
4 - Yasemin Adiguzel type : bireysel - total debt : 0
-----
5 - Veli Demir type : kurumsal_buyuk - total debt : 0
-----
6 - Suzan Celik type : kurumsal_sirketici - total debt : 0
-----
7 - Yeliz Kara type : bireysel_vip - total debt : 0
-----
8 - Tuna Yildiz type : kurumsal_kucuk - total debt : 0
-----
9 - Polat Caliskan type : kurumsal_sirketici - total debt : 0
-----
10 - Berkecan Yildirim type : kurumsal_orta - total debt : 0
-----
11 - Demet Avci type : kurumsal_orta - total debt : 0
-----
12 - Derya Polat type : bireysel - total debt : 0
-----
13 - Canan Karakaya type : kurumsal_kucuk - total debt : 0
-----
```

OPTION 3 readLoans():

The readLoans() function is responsible for reading loan data from a file and linking the loan information to the corresponding customer in the customer list. It uses fopen to open the file named "loans.txt" in read mode, and reads the customer's name, surname, loan type, loan amount, installment number, and process date. It then traverses the customer list using a while loop to find the customer that matches the name and surname read from the file. If a matching customer is found, the function dynamically allocates memory for a new loan node using malloc and assigns the loan information to the new node. The process date read from the file is parsed using strtok to extract the day, month, and year values, and sprintf is used to format the date as "year/month/day" and store it in the new loan node. Finally, the function calls the updateLoansID() function to update the IDs of all loans in the system.

OPTION 4 printLoans():

The function uses fopen to open the file named "output.txt" in append mode. It then iterates through each customer in the customer list and prints their ID, name, surname, customertype, and totaldebt. It then iterates through the loans of that customer and prints their ID, type, total amount, process date, and total installment number. Finally, it uses printf to display the information on the console and fprintf to write the same information to the "output.txt" file.

OUTPUT:

```
C:\Users\dilan\Documents\sit X + v
7. read payments.
8. find unpaid installments.
9. delete completely paid installments.
please select your option :
4
#####
1 - Ali Kaya - type : bireysel - total debt: 0
- 1L1: ihtiyac - 10256.75 - 29/11/2022 - 15
- 1L2: saglik_sigortasi - 15361.00 - 25/02/2023 - 7
-----
2 - Asli Bulut - type : kurumsal_kucuk - total debt: 0
- 2L1: ticari_arac - 845701.25 - 08/08/2020 - 12
- 2L2: kobi_destek - 90587.93 - 28/12/2022 - 17
-----
3 - Mehmet Candan - type : kurumsal_orta - total debt: 0
- 3L1: ticari_arac - 1678932.00 - 10/09/2022 - 16
- 3L2: ticari_kasko - 6589.00 - 16/04/2023 - 3
-----
4 - Yasemin Adiguzel - type : bireysel - total debt: 0
- 4L1: konut - 5998764.50 - 01/02/2021 - 27
- 4L2: bireysel_emeklilik - 20896.13 - 21/07/2021 - 9
-----
5 - Veli Demir - type : kurumsal_buyuk - total debt: 0
- 5L1: yatinim - 152745.98 - 10/12/2022 - 5
- 5L2: ticari_kasko - 17921.00 - 12/12/2022 - 5
- 5L3: calisan_saglik_sigortasi - 15410.00 - 15/12/2022 - 5
- 5L4: ticari_trafik_sigortasi - 4158.03 - 02/04/2023 - 4
- 5L5: isyeri_alimi - 6874581.00 - 03/04/2023 - 12
-----
```

OPTION 5 createInstallments():

The createInstallments() function is responsible for creating installment records for each loan in the system. It uses nested while loops to iterate through each customer and their associated loans, calculating the installment amount by dividing the total loan amount (totalamount) by the total number of installments (totalinstallmentnum). It then initializes variables for tracking the current installment, previous installment, and the first installment, and extracts the year, month, and day components from the loan's process date using sscanf. It then loops through the installments, inserting the installment into the sorted list, updating variables and advancing to the next month, and moving to the next loan and customer. When all loans are processed for a customer, it moves to the next customer in the customer list.

OPTION 6 printInstallments():

This code generates a report that contains information about customers' loans and loan installments in a financial institution. The printInstallments function iterates through the customer list and prints the details of each customer's loans and corresponding installment information to both the console and an "output.txt" file. The function includes data such as customer ID, name, loan type, total loan amount, loan process date, total number of installments, installment ID, installment date, installment amount, and payment status (paid, to be paid, or delayed payment). The function writes this information in a structured format to provide a comprehensive overview of the customers' loan statuses and installment payments.

OUTPUT:

```
1 - Ali Kaya - type : bireysel - total debt: 0
  1L1:  ihtiyac - 10256.75 - 29/11/2022 - 15
    1L1I1 -> 29/11/2022- 683.78: To be Paid
    1L1I2 -> 29/12/2022- 683.78: To be Paid
    1L1I3 -> 29/01/2023- 683.78: To be Paid
    1L1I4 -> 29/02/2023- 683.78: To be Paid
    1L1I5 -> 29/03/2023- 683.78: To be Paid
    1L1I6 -> 29/04/2023- 683.78: To be Paid
    1L1I7 -> 29/05/2023- 683.78: To be Paid
    1L1I8 -> 29/06/2023- 683.78: To be Paid
    1L1I9 -> 29/07/2023- 683.78: To be Paid
    1L1I10 -> 29/08/2023- 683.78: To be Paid
    1L1I11 -> 29/09/2023- 683.78: To be Paid
    1L1I12 -> 29/10/2023- 683.78: To be Paid
    1L1I13 -> 29/11/2023- 683.78: To be Paid
    1L1I14 -> 29/12/2023- 683.78: To be Paid
    1L1I15 -> 29/01/2024- 683.78: To be Paid
  1L2:  saglik_sigortasi - 15361.00 - 25/02/2023 - 7
    1L2I1 -> 25/02/2023- 2194.43: To be Paid
    1L2I2 -> 25/03/2023- 2194.43: To be Paid
    1L2I3 -> 25/04/2023- 2194.43: To be Paid
    1L2I4 -> 25/05/2023- 2194.43: To be Paid
    1L2I5 -> 25/06/2023- 2194.43: To be Paid
    1L2I6 -> 25/07/2023- 2194.43: To be Paid
    1L2I7 -> 25/08/2023- 2194.43: To be Paid
2 - Asli Bulut - type : kurumsal_kucuk - total debt: 0
  2L1:  ticari_arac - 845701.25 - 08/08/2020 - 12
    2L1I1 -> 08/08/2020- 70475.10: To be Paid
    2L1I2 -> 08/09/2020- 70475.10: To be Paid
    2L1I3 -> 08/10/2020- 70475.10: To be Paid
```

OPTION 7 readPayments():

First, the code opens the "payments.txt" file. If the file opening operation fails, it displays an error message and returns. It reads the loan ID and payment information from each line of the file. Then, it iterates through each customer in the customer list. For each customer, it checks their loans. If the loan ID matches the one read from the file, it enters an inner loop to process the corresponding installments. If the payment is specified as "ALL", it marks all installments as paid. Otherwise, it converts the payment information to an integer and marks the corresponding installment as paid. Once the processing is complete, it closes the file.

OUTPUT:

```
6
#####
-----
1 - Ali Kaya - type : bireysel - total debt: 0
  1L1:  ihtiyac - 10256.75 - 29/11/2022 - 15
    1L1I1 -> 29/11/2022- 683.78: Paid
    1L1I2 -> 29/12/2022- 683.78: To be Paid
    1L1I3 -> 29/01/2023- 683.78: Paid
    1L1I4 -> 29/02/2023- 683.78: Paid
    1L1I5 -> 29/03/2023- 683.78: To be Paid
    1L1I6 -> 29/04/2023- 683.78: To be Paid
    1L1I7 -> 29/05/2023- 683.78: To be Paid
    1L1I8 -> 29/06/2023- 683.78: To be Paid
    1L1I9 -> 29/07/2023- 683.78: To be Paid
    1L1I10 -> 29/08/2023- 683.78: To be Paid
    1L1I11 -> 29/09/2023- 683.78: To be Paid
    1L1I12 -> 29/10/2023- 683.78: To be Paid
    1L1I13 -> 29/11/2023- 683.78: To be Paid
    1L1I14 -> 29/12/2023- 683.78: To be Paid
    1L1I15 -> 29/01/2024- 683.78: To be Paid
  1L2:  saglik_sigortasi - 15361.00 - 25/02/2023 - 7
    1L2I1 -> 25/02/2023- 2194.43: Paid
    1L2I2 -> 25/03/2023- 2194.43: Paid
    1L2I3 -> 25/04/2023- 2194.43: To be Paid
    1L2I4 -> 25/05/2023- 2194.43: To be Paid
    1L2I5 -> 25/06/2023- 2194.43: To be Paid
    1L2I6 -> 25/07/2023- 2194.43: To be Paid
    1L2I7 -> 25/08/2023- 2194.43: To be Paid
2 - Asli Bulut - type : kurumsal_kucuk - total debt: 0
  2L1:  ticari_arac - 845701.25 - 08/08/2020 - 12
```


Options 1,2,3,4,5,6,7 in our project work smoothly and correctly. The 8th and 9th options are not written and do not give any output.