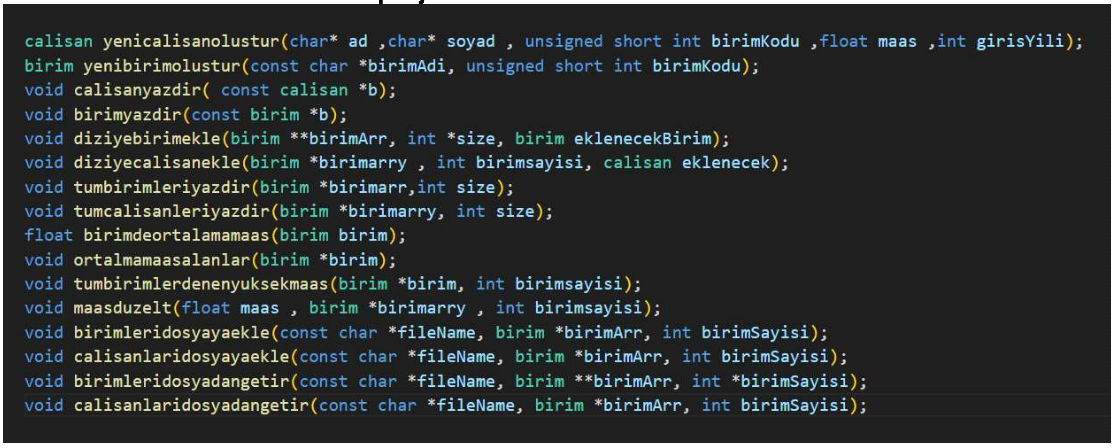
Report: Employee and Unit Management System
Introduction

-This project implements an Employee and Unit Management System using the C programming language. The system is designed to manage Birim (unit) and Calisan (employee) records, providing functionalities to add, modify, and retrieve this data efficiently. The system also supports persistence by storing and loading data from files, which makes it suitable for real-world applications where data storage is crucial. The design focuses on modularity and data integrity by ensuring operations such as preventing duplicate entries and adhering to memory management principles.

-All the Functions used in the project:



```c
calisan yenicalisanolustur(char* ad ,char* soyad , unsigned short int birimKodu ,float maas ,int girisYili);
birim yenibirimolustur(const char *birimAdi, unsigned short int birimKodu);
void calisanyazdir( const calisan *b);
void birimyazdir(const birim *b);
void diziyebirimekle(birim **birimArr, int *size, birim eklenecekBirim);
void diziyecalisanekle(birim *birimarry , int birimsayisi, calisan eklenecek);
void tumbirimleriyazdir(birim *birimarr,int size);
void tumcalisanleriyazdir(birim *birimarry, int size);
float birimdeortalamamaas(birim birim);
void ortalmamaasalanlar(birim *birim);
void tumbirimlerdenenyuksekmaas(birim *birim, int birimsayisi);
void maasduzelt(float maas , birim *birimarry , int birimsayisi);
void birimleridosyayaekle(const char *fileName, birim *birimArr, int birimSayisi);
void calisanlaridosyayaekle(const char *fileName, birim *birimArr, int birimSayisi);
void birimleridosyadangetir(const char *fileName, birim **birimArr, int *birimSayisi);
void calisanlaridosyadangetir(const char *fileName, birim *birimArr, int birimSayisi);
```

Project Features:

Dynamic Data Management:

Struct Definitions: The Birim and Calisan structures are used to represent units and employees respectively. Each Birim can hold an array of up to 20 Calisan records.
Dynamic Arrays: Using dynamic memory allocation, the system allows flexible resizing of arrays to accommodate new units and employees as needed.
Data Input/Output:

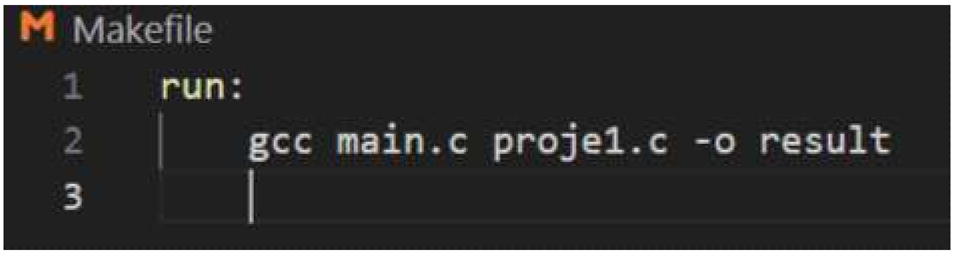
File Operations: The system supports reading data from files (birimleridosyadangetir and calisanlaridosyadangetir) and writing data to files (birimleridosyayaekle and calisanlaridosyayaekle). This ensures data persistence even after the program exits.
Command Line Arguments: Input files for units (birimler.txt) and employees (calisanlar.txt) are passed as command-line arguments, improving usability.
Core Functionalities:

Adding Units and Employees: Functions like yenibirimolustur and yenicalisanolustur

create new units and employees dynamically. These entries can then be added to arrays using diziyebirimekle and diziyecalisanekle.
Preventing Duplicates: Checks are in place to prevent duplicate units and employees from being added to the system.
Data Retrieval: The system includes functions (birimyazdir and calisanyazdir) to retrieve and print specific details of units and employees.
Salary Adjustments: The maasduzelt function adjusts salaries for employees who have worked for more than 10 years but earn less than a specified amount.
Analysis and Reporting:

Salary Analysis: Functions such as birimdeortalamamaas calculate the average salary in a unit. The ortalmamaasalanlar function identifies employees earning above the average.
Top Earners: The tumbirimlerdenenyuksekmaas function identifies and displays the highest-paid employees in each unit.

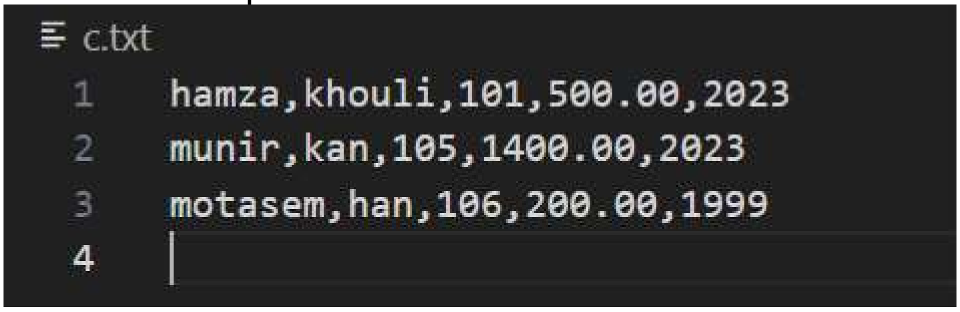
-Makefile command



Makefile

```
run:
	gcc main.c proje1.c -o result
	
```

-Calisan file added pre calisans

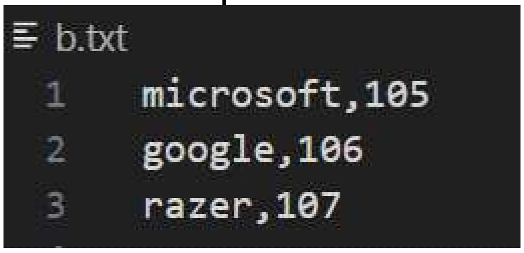


c.txt

```
hamza,khouli,101,500.00,2023
munir,kan,105,1400.00,2023
motasem,han,106,200.00,1999

```

-Birim file added pre birims



b.txt

```
microsoft,105
google,106
razer,107
```

Handling Duplicate Issues in the File System:

One of the significant challenges encountered during the implementation of the Employee and Unit Management System was managing duplicate entries when reading data from files or adding new records. This issue arises when:

Duplicate Units (Birim): Multiple units with the same birimKodu are either found in the birimler.txt file or accidentally added through the program.
Duplicate Employees (Calisan): Employees with the same name, birimKodu, salary, and join year are mistakenly added to the same unit multiple times.

Solutions to the Duplication Issues:

1. Preventing Duplicate Units
The diziyebirimekle function was updated to check for duplicates before adding a new unit:

Each new unit's birimKodu is compared against existing entries in the birimArr.
If a match is found, the unit is not added to the array.

Hamza khouli
2321021368