ISTANBUL TECHNICAL UNIVERSITY
2024

Introduction to Scientific and Engineering Computation (C)

Assignment 3

[redacted]

CEN102E: 24999

Objective:

To create a program that manages a fleet of airplanes.

Usage:

- . /fleetManager [fleetFile] [add/display] [args if command == add]

Explanation of the code:

1. **Functions:**

- **Fleet readFleetFromFile(char *filename);**
  - Takes target file pointer as argument and reads the fleet details inside
  - Returns struct type named Fleet
- **void writeFleetToFile(char *filename, Fleet *fleet);**
  - Takes target file pointer and fleet struct pointer as arguments
  - Appends new airplane details into target file
  - Returns nothing
- **void addAirplane(Fleet *fleet, Airplane newAirplane);**
  - Takes fleet struct pointer and airplane struct as inputs
  - Adds airplane into fleet
  - Returns nothing
- **void displayFleet(Fleet fleet);**
  - Takes fleet struct as argument
  - Displays fleet to terminal
  - Returns nothing

2. Structs

a. Airplane

```
typedef struct {
    char tailNumber[50];
    char model[50];
    int year;
    int capacity;
} Airplane;
```

- **char tailNumber[50]**
  - Character array to hold tail number of airplanes
- **char model[50]**
  - Character array to hold model name of airplanes
- **int year**
  - İnteger for manufacturing year of airplane
- **int capacity**
  - İnteger for airplane capacity

b. Fleet

```
typedef struct {
    int count;
    Airplane *airplanes;
} Fleet;
```

- **int count**
  - Integer to hold airplane count in a fleet
- **Airplane* airplanes**
  - Dynamic struct array to hold airplanes

3. How Code Works

Starting from main, code checks if we have at least 3 arguments (executable, filename, command) because in display mode we have 3 arguments.

Next, we read the target file and put all details in it to a local fleet struct to work on with **readFleetFromFile** function.

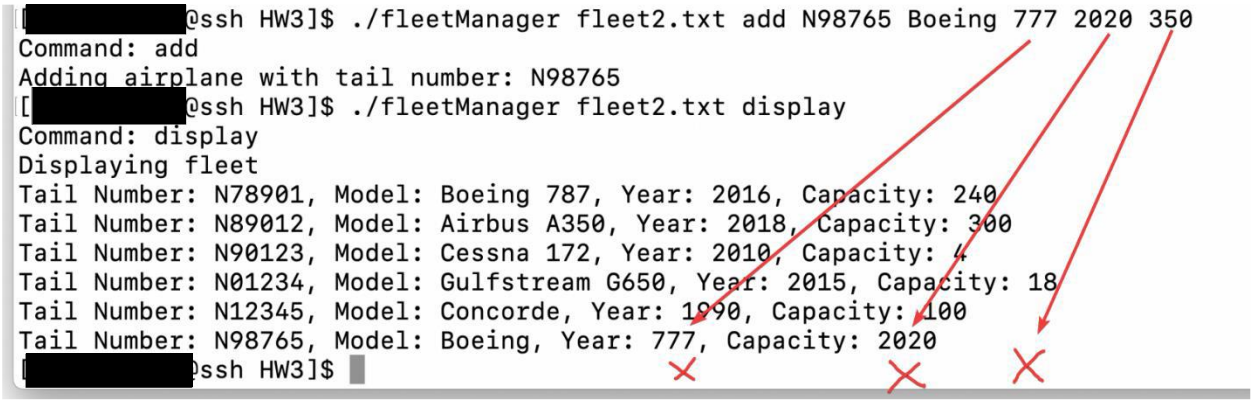
We check whether the command is add or display and move accordingly in an if block.

- **There was an error in pdf file and also a confusion:**

```
./fleet_manager fleet1.txt add N98765,Boeing 777,2018,300
./fleet_manager fleet2.txt display
```

```
Tail Number: N12345, Model: Concorde, Year: 1990, Capacity: 100
[redacted]ssh HW3]$ ./fleetManager fleet2.txt add N98765 Boeing 777 2020 350
Command: add
```

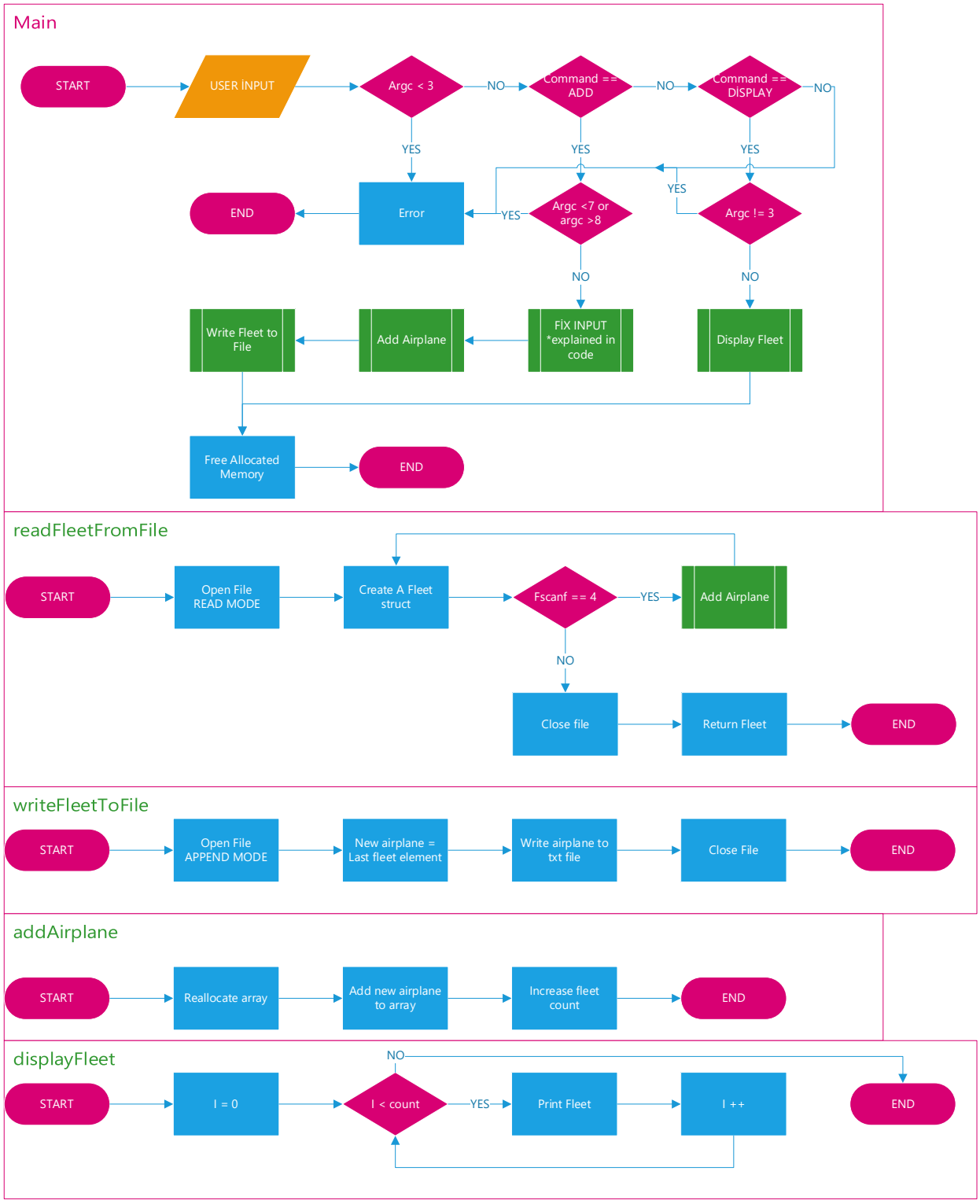
- Confusion: As you can see there is two different uses on pdf. So, I decided to go on with the second one as usually comma is not used on terminal inputs.



- Error: I think the input was meant to be
  - ./fleetManager fleet1.txt add N9876 **"Boeing 777"** 2020 350

I coded my code to work with 8 arguments because of this input issue, and I explained in code's comments. My code takes 7 or 8 arguments, if 8 arguments given 2 of them are taken as model of the airplane.

Inside the if block, we either display the fleet using **displayFleet** function or add a new airplane with **addAirplane** function. In 'add' mode, we take arguments from user and put them into local variables. Then we create an **airplane struct** and pass it into **addAirplane** function. Function reallocates airplanes array in fleet then places the new airplane. Next, we open the file and append the new airplane to **.txt** file with **writeFleetToFile** function.


Main
START
USER İNPUT
Argc < 3
NO
Command == ADD
NO
Command == DİSPLAY
NO
YES
YES
YES
YES
END
Error
YES
Argc <7 or argc >8
Argc != 3
NO
NO
Write Fleet to File
Add Airplane
FİX INPUT *explained in code
Display Fleet
Free Allocated Memory
END
readFleetFromFile
START
Open File READ MODE
Create A Fleet struct
Fscanf == 4
YES
Add Airplane
NO
Close file
Return Fleet
END
writeFleetToFile
START
Open File APPEND MODE
New airplane = Last fleet element
Write airplane to txt file
Close File
END
addAirplane
START
Reallocate array
Add new airplane to array
Increase fleet count
END
displayFleet
NO
START
I = 0
I < count
YES
Print Fleet
I ++
END

4. References

Geeks for Geeks – for fscanf usage

Tutorials Point – for argc argv main arguments

Stack Overflow - for argc argv main arguments