

Name	Jhaveri Varun Nimitt
UID no.	2023800042
Experiment No.	10

AIM:	Implement various operations on files to solve a given problem.
-------------	---

Program 1

PROBLEM STATEMENT :	A publishing company holds in a file details of all the books they publish. However, in the future, they wish to maintain two distinct files (i) paperbacks (ii) hardbacks. Write a program which reads a file containing details of both paperback and hardback books and creates two files as specified above. Assume that the first character in each input record indicates if the book is paperback(p) or hardback(h) or both(b).
----------------------------	--

ALGORITHM:	<ol style="list-style-type: none"> 1. Start 2. Declare FILE pointers for the main file, paperback books file, and hardback books file. 3. Declare a character array 'record' to store each line read from the input file. 4. Open the main input file "input.txt" in read mode (r) and check for errors. <ol style="list-style-type: none"> a. If an error occurs, print an error message and return 1. 5. Open the paperback books file "paperbacks.txt" in write mode (w) and check for errors. <ol style="list-style-type: none"> a. If an error occurs, print an error message and return 1. 6. Open the hardback books file "hardbacks.txt" in write mode (w) and check for errors. <ol style="list-style-type: none"> a. If an error occurs, print an error message and return 1. 7. Read each line from the main file using fgets until the end of the file is reached. <ol style="list-style-type: none"> a. Check if the first character of the line (record[0]) is 'p' (indicating paperback). <ol style="list-style-type: none"> i. If true, write the record to the paperback books file using fputs. b. If the first character is 'h' (indicating hardback), <ol style="list-style-type: none"> i. Write the record to the hardback books file using fputs. 8. Close all the opened files: main file, paperback books file, and hardback books file. 9. End
-------------------	--

PROGRAM:

```
#include <stdio.h>

int main() {
    FILE *main_file, *paperback_books_file, *hardback_books_file;
    char record[100];

    main_file = fopen("input.txt", "r");
    if (main_file == NULL)
    {
        printf("Error opening main book file\n");
        return 1;
    }

    paperback_books_file = fopen("paperbacks.txt", "w");
    if (paperback_books_file == NULL)
    {
        printf("Error opening paperback book file\n");
        return 1;
    }

    hardback_books_file = fopen("hardbacks.txt", "w");
    if (hardback_books_file == NULL)
    {
        printf("Error opening hardback book file\n");
        return 1;
    }

    while (fgets(record, sizeof(record), main_file))
    {
        if (record[0] == 'p')
        {
            fputs(record, paperback_books_file);
        }
        else if (record[0] == 'h')
        {
            fputs(record, hardback_books_file);
        }
        /*else if (record[0] == 'b')
        {
            fputs(record, hardback_books_file);
        }
    }
}
```

```

        fputs(record, paperback_books_file);
        fputs(record, hardback_books_file);
    }*/
    //question does not state what to do with
    books that are both hardback and paperback

}

fclose(main_file);
fclose(paperback_books_file);
fclose(hardback_books_file);

return 0;
}

```

RESULT:

INPUT.TXT:

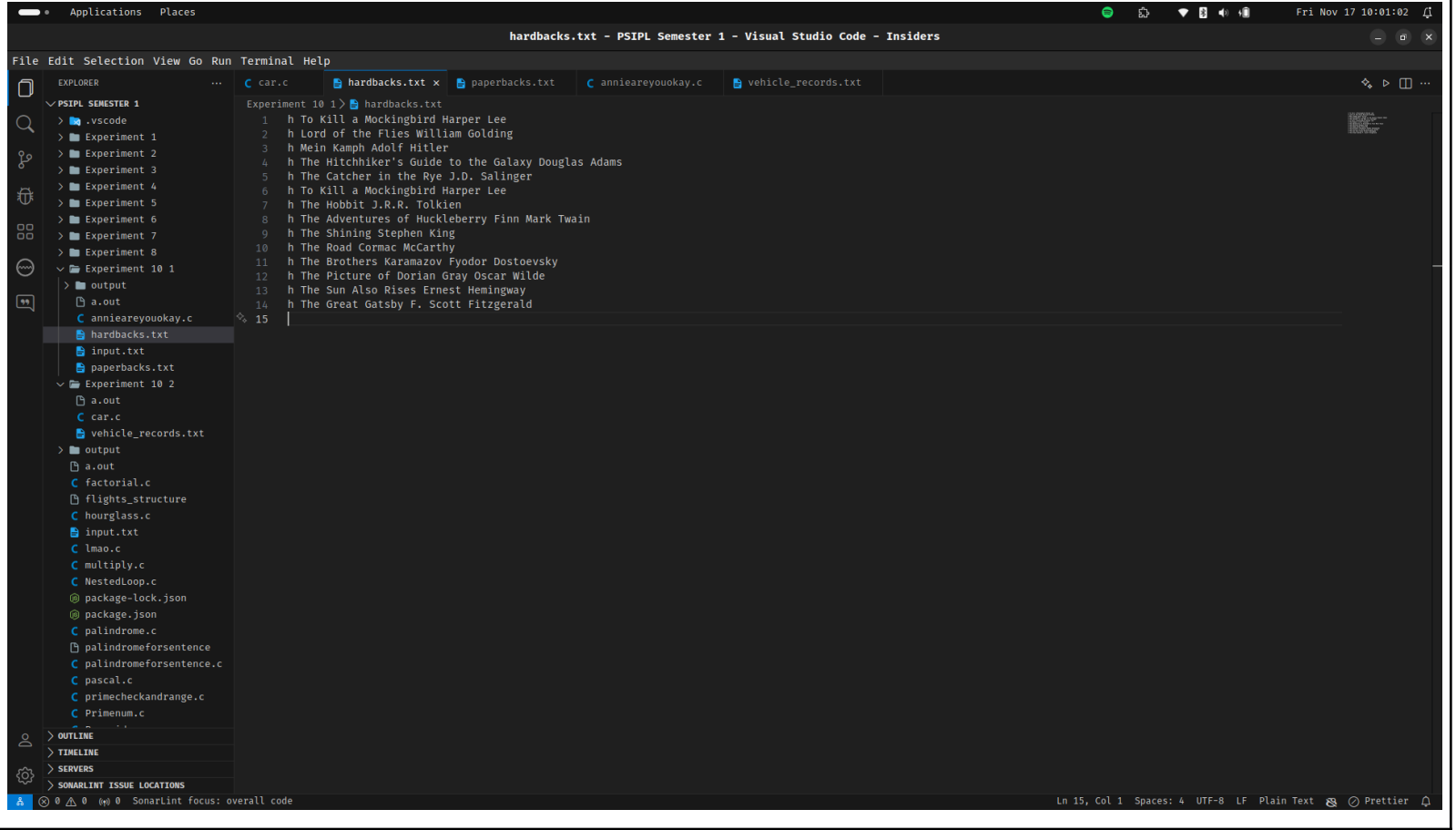
The screenshot shows the Visual Studio Code interface with the 'input.txt' file open. The file contains a list of books and authors, each preceded by a number and a letter (e.g., '1 h The Catcher in the Rye J.D. Salinger'). The Explorer sidebar on the left shows the project structure, including a folder named 'PSIPL SEMESTER 1' and various files and folders. The status bar at the bottom indicates the current line and column (Ln 1, Col 1) and the encoding (UTF-8).

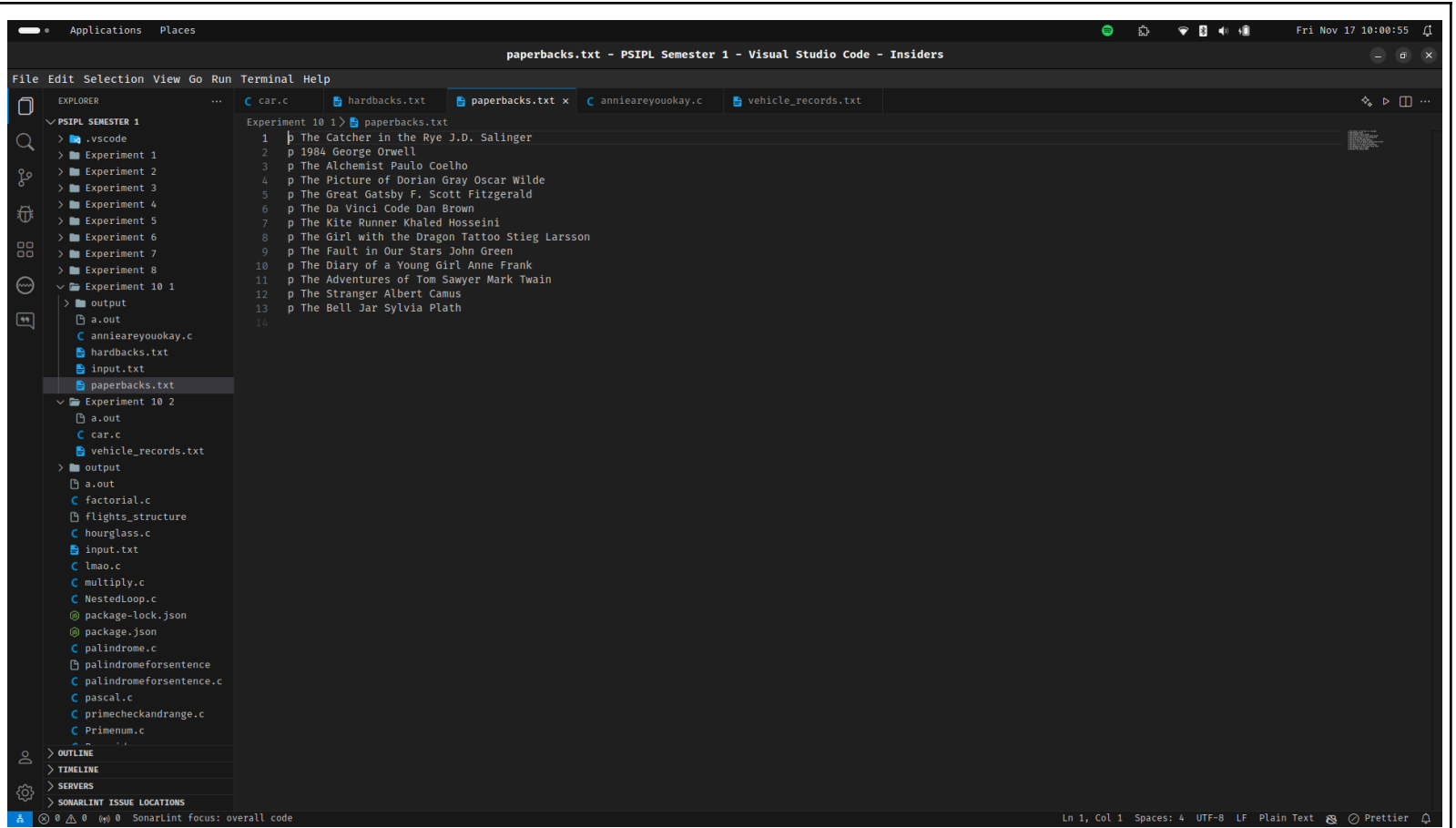
```

input.txt - PSIPL Semester 1 - Visual Studio Code - Insiders
File Edit Selection View Go Run Terminal Help
EXPLORER
PSIPL SEMESTER 1
  .vscode
  Experiment 1
  Experiment 2
  Experiment 3
  Experiment 4
  Experiment 5
  Experiment 6
  Experiment 7
  Experiment 8
  Experiment 10 1
    output
      a.out
      annieareyouokay.c
      hardbacks.txt
      input.txt
      paperbacks.txt
  Experiment 10 2
    a.out
    car.c
    vehicle_records.txt
  output
    a.out
    factorial.c
    flights_structure
    hourglass.c
    input.txt
    lmao.c
    multiply.c
    NestedLoop.c
    package-lock.json
    package.json
    palindrome.c
    palindromeforsentence
    palindromeforsentence.c
    pascal.c
    primecheckandrange.c
    Primenum.c
    ...
  OUTLINE
  TIMELINE
  SERVERS
  SONARLINT ISSUE LOCATIONS
Ln 1, Col 1 Spaces: 4 UTF-8 LF Plain Text Prettier

```

PAPERBACKS AND HARDBACKS.txt AFTER PROGRAM EXECUTION:





Program 2

PROBLEM STATEMENT :

Set up a file containing vehicle records which hold registration number and owner information (name and address). Write a program which, given a vehicle's registration number, will rapidly retrieve and print the owner information.

ALGORITHM:

1. Start
2. Define the function FindOwnerInfo that takes a registration number (regNumber) as input and finds corresponding owner information.
 - a. Open the "vehicle_records.txt" file in read mode (r) and check for errors.
 - i. If the file is not found, print an error message and exit the program.
 - b. Declare variables:
 - i. FILE pointer 'record' to handle file operations.
 - ii. Character arrays 'line', 'fileRegNumber', 'ownerName', and 'ownerCountry' to store read data from the file.
 - iii. Integer variable 'found' to keep track of whether the registration number is found in the file (initialize to 0).

- c. Use a while loop to read each line from the file using fgets until the end of the file is reached.
 - i. Parse each line using sscanf to extract 'fileRegNumber', 'ownerName', and 'ownerCountry'.
 - ii. Compare 'fileRegNumber' with the provided 'regNumber'.
 - If a match is found:
 - a. Print the owner's name and country corresponding to the registration number.
 - b. Set 'found' to 1 to indicate the registration number was found and break out of the loop.
 - d. If 'found' is still 0 after checking all lines, print a message indicating the registration number was not found in the file.
 - e. Close the 'vehicle_records.txt' file.
- 3. Define the main function.
 - a. Declare a character array 'regNumber' to store the user-input registration number.
 - b. Use an infinite loop to continuously prompt the user for a registration number and find its owner information.
 - i. Prompt the user to enter the registration number.
 - ii. Read the registration number using scanf.
 - iii. Call the FindOwnerInfo function, passing the entered registration number.
 - iv. Prompt the user to continue by pressing 1 or exit by pressing 0 .
 - Read the user's choice using scanf.
 - If the choice is 0, break out of the loop and exit the program.
- 4. End.

PROGRAM:

```
#include <stdio.h>
#include <string.h>
```

```

#include <stdlib.h>

void FindOwnerInfo(char* regNumber)
{
    FILE* record = fopen("vehicle_records.txt", "r");
    if (record == NULL)
    {
        printf("Error: File not found.\n");
        exit(0);
    }
    char line[100];
    char fileRegNumber[20];
    char ownerName[50];
    char ownerCountry[50];
    int found = 0;

    while (fgets(line, sizeof(line), record))
    {
        sscanf(line, "%s %s %s", fileRegNumber, ownerName, ownerCountry);
        if (strcmp(fileRegNumber, regNumber) == 0)
        {
            printf("Information of Owner of vehicle: %s , %s\n",
ownerName, ownerCountry);
            found = 1;
            break;
        }
    }

    if (!found)
    {
        printf("Registration number not found.\n");
    }

    fclose(record);
}

int main() {
    char regNumber[20];

```

```

while(1)
{
    printf("Enter the registration number: ");
    scanf("%s", regNumber);
    FindOwnerInfo(regNumber);
    printf("Press 1 to continue, 0 to exit: ");
    int choice;
    scanf("%d", &choice);
    if (choice == 0)
    {
        break;
    }
}
return 0;
}

```

RESULT: VEHICLE_RECORDS.TXT :

The screenshot shows the Visual Studio Code editor with the file 'vehicle_records.txt' open. The file contains a list of 20 vehicle records, each with a registration number, owner name, and country. The records are as follows:

Line	Registration Number	Owner Name	Country
1	KA23YZ1234	John	Canada
2	KA67IJ5678	Charlie	Mexico
3	KA45WX2345	Jane	Brazil
4	KA89EF6789	Bob	Australia
5	KA34AB3456	Alice	France
6	KA56QR7890	John	Germany
7	KA12CD4567	Charlie	Japan
8	KA78MN8901	Eve	Russia
9	KA90UV2345	Bob	China
10	KA78OP5678	Alice	Italy
11	KA12KL9012	John	Spain
12	KA34GH3456	Charl	South Africa
13	KA56ST7890	Jane	Argentina
14	KA90IJ2345	Bob	India
15	KA78YZ5678	Alice	United Kingdom
16	KA12WX9012	John	Sweden
17	KA34EF3456	Charlie	Norway
18	KA56AB7890	Johnson	Egypt
19	KA90QR2345	Doe	Turkey
20	KA78CD5678	Lee	South Korea
21			

The Visual Studio Code interface shows the Explorer sidebar on the left with the file 'vehicle_records.txt' selected. The status bar at the bottom indicates 'Ln 17, Col 26' and 'Spaces: 4'.

OUTPUT:

```
cyclops@cyclops:~/Desktop/PSIPL Semester 1/Experiment 10 2$ ./a.out
Enter the registration number: KA34EF3456
Information of Owner of vehicle: Charlie , Norway
Press 1 to continue, 0 to exit: 1
Enter the registration number: KA89EF6789
Information of Owner of vehicle: Bob , Australia
Press 1 to continue, 0 to exit: 1
Enter the registration number: KA90QR2345
Information of Owner of vehicle: Doe , Turkey
Press 1 to continue, 0 to exit: 0
cyclops@cyclops:~/Desktop/PSIPL Semester 1/Experiment 10 2$ █
```

CONCLUSION:

I have learnt how to implement various operations on files to solve a given problem and understood the concept behind different file functions.