FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

Library Management System

FACULTY OF ENGINEERING

Function 1

```c
//function 1 Number_Books
int Number_Books(const char *bookfile){
    FILE *file = fopen(bookfile, "r");
    if (file == NULL) {
        printf("Error opening file.\n");
        return -1;  // Indicate error
    }
    int count = 0;
    char line[MAX_LINE_LEN];
    while (fgets(line, sizeof(line), file)){  // Read each line
        char *token = strtok(line, ",");      // Get the first column
        if (token != NULL) {                  // If the first column exists
            count++;
        }
    }

    fclose(file);
    if (count == 0) {
        printf("none\n");
    }
    return count-1;  // Return the count
}
```

➔The function counts the number of book IDs in a CSV file where the first column contains the book IDs.

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

FACULTY OF ENGINEERING

Function 2

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```c
//fuction 2 Number_Members
int Number_members(const char *membersfile){
    FILE *file= fopen(membersfile,"r");
    if (file==NULL){
        printf("not found\n");
        return-1;
}
    int counter=0;
    char line[MAX_LINE_LEN];

   while (fgets(line, sizeof(line), file)){  // Read each line
        char *token = strtok(line, ",");      // Get the first column
        if (token != NULL) {                  // If the first column exists
            counter++;
        }
    }
     if (counter == 0) {
        printf("none\n");
    }

    fclose(file);
    return counter-1;

}
```

➔ The function counts the number of member IDs in a CSV file where the first column contains the member IDs.

FACULTY OF ENGINEERING

Function 3

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
212  //function 3 Book_ID_Min
213  void Book_ID_Min(const char *booksFile) {
214      FILE *file = fopen(booksFile, "r"); // Open the books file in read mode
215      if (!file) {
216          printf("Error: Unable to open file %s\n", booksFile);
217          return;}
218      char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
219      int bookID, copies;
220      int minBookID = -1; // Variable to track the minimum book ID (-1 indicates no books)
221
222      // Read the file line by line
223      while (fgets(line, sizeof(line), file)) {
224          // Parse the line to extract Book ID and Copies
225          if (sscanf(line, "%d,%d", &bookID, &copies) == 2){
226              if (minBookID == -1 || bookID < minBookID) {
227                  minBookID = bookID; // Update the minimum book ID
228                  }}}
229      fclose(file); // Close the file
230      // Print the minimum Book ID or "none" if no books were found
231      if (minBookID == -1) {
232          printf("none\n");
233      } else {
234          printf("%d\n", minBookID);
235      }
236  }
```

➔The function identifies the minimum book ID in a CSV file where the first column contains the book IDs.

FACULTY OF ENGINEERING

Function 4

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
237  //function 4 Books_Available
238  void Books_Available(const char *booksFile) {
239      FILE *books = fopen(booksFile, "r"); // Open the books file in read mode
240      if (!books) {
241          printf("Error: Unable to open file %s\n", booksFile);
242          return;
243      }
244
245      char line[MAX_LINE_LEN]; // Buffer to hold each line from the books file
246      int bookID, copies;
247      int found = 0; // Flag to check if at least one book is available
248
249      // Read the books file line by line
250      while (fgets(line, sizeof(line), books)){
251          // Parse each line to extract Book ID and Copies
252          if (sscanf(line, "%d,%d", &bookID, &copies) == 2) {
253              if (copies > 0) { // Check if the book has more than 0 copies
254                  printf("%d\n", bookID);
255                  found = 1;
256              }
257          }
258      }
259
260      fclose(books);
261
262      // If no books are available, print "none"
263      if (!found) {
264          printf("none\n");
265      }
266  }
```

➔The Books_Available function checks a file for books with available copies and prints their IDs. If no books are available, it outputs "none".

FACULTY OF ENGINEERING

Function 5

```
267   // function 5 List Book Borrowers
268   void List_Book_Borrowers(const char *loansFile, int bookID) {
269       FILE *file = fopen(loansFile, "r"); // Open the loans file in read mode
270       if (!file) {
271           printf("Error: Unable to open file %s\n", loansFile);
272           return;
273       }
274
275       char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
276       int currentBookID, memberID;
277       int found = 0; // Flag to check if any member borrowed the book
278
279       // Read the file line by line
280       while (fgets(line, sizeof(line), file)){
281           // Parse the line into Book ID and Member ID
282           if (sscanf(line, "%d,%d", &currentBookID, &memberID) == 2) {
283               if (currentBookID == bookID) {
284                   printf("%d\n", memberID); // Print the Member ID
285                   found = 1;
286               }
287           }
288       }
289
290       fclose(file); // Close the file
291
292       if (!found) {
293           printf("none\n"); // Print "none" if no member borrowed the book
294       }
295   }
```

➔The function lists the IDs of members who borrowed a specific book based on the book ID.

FACULTY OF ENGINEERING

Function 6

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
296   //function 6 List_Member_Books
297   void List_Member_Books(const char *loansFile, int memberID) {
298       FILE *file = fopen(loansFile, "r"); // Open the loans file in read mode
299       if (!file) {
300           printf("Error: Unable to open file %s\n", loansFile);
301           return;
302       }
303
304       char line[MAX_LINE_LEN];  // Buffer to hold each line from the file
305       int currentBookID, currentMemberID;
306       int found = 0;            // Flag to check if any books were borrowed by the member
307         char date[11];
308       // Read the loans file line by line
309       while (fgets(line, sizeof(line), file)) {
310           // Parse the line to extract Book ID, Member ID, and Loan Date
311           if (sscanf(line, "%d,%d,%s", &currentBookID, &currentMemberID, date) == 3){
312               if (currentMemberID == memberID) {
313                   printf("%d\n", currentBookID); // Print the Book ID
314                   found = 1;
315               }
316           }
317       }
318
319       fclose(file); // Close the loans file
320
321       // If no books were borrowed by the member, print "none"
322       if (!found) {
323           printf("none\n");
324       }
325   }
```

➔The function is designed to list the books borrowed by a specific library member based on their ID.

FACULTY OF ENGINEERING

Function 7

```c
//function 7 Most Borrowed
void Most_Borrowed(const char *booksFile) {
    FILE *file = fopen(booksFile, "r"); // Open the books file in read mode
    if (!file) {
        printf("Error: Unable to open file %s\n", booksFile);
        return;
    }

    char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
    int bookID, totalCopies, borrowCount;
    int maxBorrowCount = -1; // To track the maximum borrow count
    int found = 0;           // Flag to check if at least one book exists

    // Step 1: Find the maximum borrow count
    while (fgets(line, sizeof(line), file)) {
        // Parse each line to extract Book ID, Total Copies, and Borrow Count
        if (sscanf(line, "%d,%d,%d", &bookID, &totalCopies, &borrowCount) == 3) {
            if (borrowCount > maxBorrowCount) {
                maxBorrowCount = borrowCount; // Update maximum borrow count
            }
        }
    }

    rewind(file); // Reset the file pointer to start for the second pass

    // Step 2: Find all books with the maximum borrow count
    while (fgets(line, sizeof(line), file)) {
        if (sscanf(line, "%d,%d,%d", &bookID, &totalCopies, &borrowCount) == 3) {
            if (borrowCount == maxBorrowCount) {
                printf("%d\n", bookID); // Print the Book ID
                found = 1;
            }
        }
    }

    fclose(file); // Close the books file

    // If no books are found, print "none"
    if (!found) {
        printf("none\n");
    }
}
```

➔The function identifies the book(s) with the highest borrow count in a CSV file where the columns contain book ID, total copies, and borrow count.

FACULTY OF ENGINEERING

Function 8

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```c
//function 8 Members_Less n
void Members_Less(const char *membersFile, int n) {
    FILE *file = fopen(membersFile, "r"); // Open the members file in read mode
    if (!file) {
        printf("Error: Unable to open file %s\n", membersFile);
        return;
    }

    char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
    int memberID, borrowCount;
    int found = 0; // Flag to check if any member meets the criteria

    // Read the file line by line
    while (fgets(line, sizeof(line), file)) {
        // Parse each line to extract Member ID and Borrow Count
        if (sscanf(line, "%d,%d", &memberID, &borrowCount) == 2) {
            if (borrowCount < n) {
                printf("%d\n", memberID); // Print the Member ID
                found = 1;
            }
        }
    }

    fclose(file); // Close the file

    if (!found) {
        printf("none\n"); // Print "none" if no members meet the criteria
    }
}
```

➔ The function is designed to list the IDs of members who have borrowed fewer books than a specified number (n)

FACULTY OF ENGINEERING

Function 9

```c
// function 9 Books_Unborrowed
void Books_Unborrowed(const char *booksFile) {
    FILE *books = fopen(booksFile, "r"); // Open the books file in read mode
    if (!books) {
        printf("Error: Unable to open file %s\n", booksFile);
        return;
    }

    char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
    int bookID, copies, borrowCount;
    int found = 0; // Flag to check if any unborrowed books exist

    // Iterate through each line in the books file
    while (fgets(line, sizeof(line), books)){
        // Parse each line to extract Book ID, Total Copies, and Borrow Count
        if (sscanf(line, "%d,%d,%d", &bookID, &copies, &borrowCount) == 3){
            if (borrowCount == 0) { // If the borrow count is zero
                printf("%d\n", bookID); // Print the Book ID
                found = 1;
            }
        }
    }

    fclose(books); // Close the books file

    // If no unborrowed books are found, print "none"
    if (!found) {
        printf("none\n");
    }
}
```

➔The function identifies and lists books that have not been borrowed based on their borrow count.

FACULTY OF ENGINEERING

Function 10

```c
//function 10 Books_Borrowed_Days
int Books_Borrowed_Days(FILE *loanfile){
    FILE *file = fopen(loanfile, "r");
    if (file == NULL)
    {
        printf("Error opening file.\n");
        return -1;  // Indicate error
    }


    char unique_dates[10000][11]; //2D array to store each date in each row
    int count = 0;
    char line[MAX_LINE_LEN];



    while (fgets(line, sizeof(line), file)) {
        char *token = strtok(line, ",");
        token = strtok(NULL, ",");
        token = strtok(NULL, "\n");      // Column 3: Date

        if (token != NULL) {
            int unique = 1; // This is a flag to know if it is a unique date or not

            // Check if the date is already in the unique_dates array
            for (int i = 0; i < count; i++)
            {
                if (strcmp(unique_dates[i], token) == 0) {
                    unique = 0;
                    break;
                }
            }

            // Add new unique date
            if (unique)
            {
                strcpy(unique_dates[count], token);
                count++;
                // to prevent errors
                if (count >= 10000 )
                {
                    printf("Error: Exceeded maximum unique dates.\n");
                    return count;
                }
            }
        }
    }
// Print "none" if no unique dates found
    if (count == 0) {
        printf("none\n");
    }
    return count;
}
```

➔The function calculates the total number of unique borrowing dates from a loan records CSV file. Each line contains a member ID, book ID, and borrowing date, and the function ensures that repeated dates are only counted once.

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

FACULTY OF ENGINEERING

Function 11

```c
//function 11 Books_Per_Member
void Books_Per_Member(const char *membersFile) {
    FILE *file = fopen(membersFile, "r"); // Open the members file in read mode
    if (!file) {
        printf("Error: Unable to open file %s\n", membersFile);
        return;
    }

    char line[MAX_LINE_LEN]; // Buffer to hold each line from the file
    int memberID, borrowCount;
    int found = 0; // Flag to check if there are any members

    // Read the file line by line
    while (fgets(line, sizeof(line), file)) {
        // Parse each line to extract Member ID and Borrow Count
        if (sscanf(line, "%d,%d", &memberID, &borrowCount) == 2) {
            printf("%d %d\n", memberID, borrowCount); // Print Member ID and Borrow Count
            found = 1;
        }
    }

    fclose(file);

    // If no members are found, print "none"
    if (!found) {
        printf("none\n");
    }
}
```

➔This function reads a file containing member information and prints the IDs of members along with the number of books they have borrowed. If no data is found, it prints "none".

FACULTY OF ENGINEERING

Function 12

```c
//function 12 Overlapping_Borrowers
void Overlapping_Borrowers(const char *loansFile, int bookID){
    FILE *file = fopen(loansFile, "r"); // Open the loans file in read mode
    if (!file) {
        printf("Error: Unable to open file %s\n", loansFile);
        return;
    }

    char line[MAX_LINE_LEN];  // Buffer to hold each line from the file
    int currentBookID, memberID;
    char date[11];            // To store the loan date (format: dd/mm/yyyy)

    int found = 0;            // Flag to check if any overlapping borrowers exist

    // Step 1: Read all loans for the given Book ID
    while (fgets(line, sizeof(line), file)) {
        // Parse each line to extract BookID, MemberID, and Date
        if (sscanf(line, "%d,%d,%s", &currentBookID, &memberID, date) == 3) {
            if (currentBookID == bookID) {
                // Step 2: Compare this loan with all other loans in the file
                FILE *innerFile = fopen(loansFile, "r"); // Open the loans file again for comparison
                char innerLine[MAX_LINE_LEN];
                int innerBookID, innerMemberID;
                char innerDate[11];
                while (fgets(innerLine, sizeof(innerLine), innerFile)) {
                    // Parse the inner loop line
                    if (sscanf(innerLine, "%d,%d,%s", &innerBookID, &innerMemberID, innerDate) == 3) {
                        // Check if the book ID and date match, but the member ID is different
                        if (innerBookID == bookID && strcmp(date, innerDate) == 0 && memberID != innerMemberID) {
                            printf("%d\n", memberID);
                            found = 1;
                            break;
                        }
                    }
                }
                fclose(innerFile); // Close the inner file after comparison
            }}}

fclose(file); // Close the loans file
// If no overlapping borrowers are found, print "none"
    if (!found) {
        printf("none\n");
    }
}
```

➔This function identifies members who borrowed the same book on the same date as another member.

FACULTY OF ENGINEERING

addMemberToFile function

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
 8  void addMemberToFile(const char *membersFile, int memberID) {
 9      FILE *file = fopen(membersFile, "a");
10      if (file == NULL) {
11          printf("Error: Unable to open file %s\n", membersFile);
12          return;
13      }
14      int memberBorrows =0;
15
16      if (memberID > 999999) {
17          printf("Error: Member ID must be no longer than 6 digits.\n");
18          fclose(file);
19          return;
20      }
21
22      fprintf(file, "%d,%d\n", memberID, memberBorrows);
23      fclose(file);
24  }
```

FILE* addMemberToFile::file

➔This function adds a new member to the members file with their ID and initial borrow count.

FACULTY OF ENGINEERING

addBookToFile function

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
25  void addBookToFile(const char *bookfile, int id, int copies) {
26      FILE *file = fopen(bookfile, "a");
27      if (file == NULL) {
28          printf("Error: Unable to open file %s\n", bookfile);
29          return;
30      }
31      int bookBorrows = 0; // Default borrow count
32
33      if (id > 999){
34          printf("Error: Book ID must be no longer than 3 digits.\n");
35          fclose(file);
36          return;
37      }
38
39      // Write the new book data to the file
40      fprintf(file, "%d,%d,%d\n", id, copies, bookBorrows);
41      fclose(file);
42  }
```

➔This function adds a new book to the books file with its ID, available copies, and borrow count.

FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

addLoan function

```
149 void addLoan(const char *loansFile, const char *membersFile, const char *booksFile, int bookID, int memberID, const char *date) {
150     int result = processloan(membersFile, booksFile, bookID, memberID);
151     if (result != 0) {
152         return;
153     }
154
155     // If processLoan succeeds, add the loan to the loans file
156     FILE *file = fopen(loansFile, "a");
157     if (file == NULL) {
158         printf("Error: Unable to open file %s\n", loansFile);
159         return;
160     }
161
162     fprintf(file, "%d,%d,%s\n", bookID, memberID, date);
163     fclose(file);
164 }
```

➔This function processes a loan request and adds it to the loans file. It validates the loan through the `processLoan` function before appending it.

FACULTY OF ENGINEERING

processloan function

```c
int processloan(const char *membersFile, const char *booksFile, int bookID, int memberID,const char *date) {
    // Open the members file to validate and update the borrow count
    FILE *members = fopen(membersFile, "r+");
    if (!members) {
        printf("Error: Unable to open file %s\n", membersFile);
        return;
        }
    char line[MAX_LINE_LEN];
    int currentMemberID, borrowCount = 0;
    int memberFound = 0;

    FILE *tempMembers = fopen("temp_members.csv", "w");
    if (!tempMembers) {
        printf("Error: Unable to create temporary file.\n");
        fclose(members);
        return;
        }
    while (fgets(line, sizeof(line), members)) {
        if (sscanf(line, "%d,%d", &currentMemberID, &borrowCount) == 2) {
            if (currentMemberID == memberID) {
                memberFound = 1;
                if (borrowCount >= 5) {
                    printf("Error: Member ID %d has already borrowed the maximum number of books (5).\n", memberID);
                    fclose(members);
                    fclose(tempMembers);
                    remove("temp_members.csv");
                    return 1; // Indicate failure
                }
                // Update the borrow count for the member
                fprintf(tempMembers, "%d,%d\n", currentMemberID, borrowCount + 1);
            }
             else {
                fprintf(tempMembers, "%s", line);
            }
        } else {
            fprintf(tempMembers, "%s", line); // Handle malformed lines
        }
    }

```

➔This function validates and processes a loan request, updating the member's borrow count and the book's borrow count in their respective files.

```
82      if (!memberFound) {
83          printf("Error: Member ID %d not found.\n", memberID);
84          fclose(members);
85          fclose(tempMembers);
86          remove("temp_members.csv");
87          return 1; // Indicate failure
88      }
89
90      fclose(members);
91      fclose(tempMembers);
92      remove(membersFile);
93      rename("temp_members.csv", membersFile);
94
95      // Open the books file to validate and update the borrow count
96      FILE *books = fopen(booksFile, "r+");
97      if (!books) {
98          printf("Error: Unable to open file %s\n", booksFile);
99          return;
100     }
101
102     int currentBookID, totalCopies, currentBorrowCount = 0;
103     int bookFound = 0;
104
105     FILE *tempBooks = fopen("temp_books.csv", "w");
106     if (!tempBooks) {
107         printf("Error: Unable to create temporary file for books.\n");
108         fclose(books);
109         return;
110     }
111
```

```c
    while (fgets(line, sizeof(line), books)) {
        if (sscanf(line, "%d,%d,%d", &currentBookID, &totalCopies, &currentBorrowCount) == 3) {
            if (currentBookID == bookID) {
                bookFound = 1;
                if (currentBorrowCount >= totalCopies) {
                    printf("Error: No copies available for Book ID %d.\n", bookID);
                    fclose(books);
                    fclose(tempBooks);
                    remove("temp_books.csv");
                     return 1; // Indicate failure
                }
                // Update the borrow count for the book
                fprintf(tempBooks, "%d,%d,%d\n", currentBookID, totalCopies, currentBorrowCount + 1);
            } else {
                fprintf(tempBooks, "%s", line);
            }
        } else {
            fprintf(tempBooks, "%s", line); // Handle malformed lines
        }
    }

    if (!bookFound) {
        printf("Error: Book ID %d not found.\n", bookID);
        fclose(books);
        fclose(tempBooks);
        remove("temp_books.csv");
        return 1; // Indicate failure
    }
    int day, month, year;
   if (sscanf(date, "%d/%d/%d", &day, &month, &year) != 3 || day < 1 || day > 31 || month < 1 || month > 12 || year < 1) {
    printf("Error: Invalid date format. Please use dd/mm/yyyy.\n");
    return 1;
}

    fclose(books);
    fclose(tempBooks);
    remove(booksFile);
    rename("temp_books.csv", booksFile);

    return 0; // Indicate success

}
```

FACULTY OF ENGINEERING

main function

```
556  int main() {
557      const char *booksFile = "Books.csv";
558      const char *membersFile = "members.csv";
559      const char *loansFile = "loans.csv";
560      char header[MAX_LINE_LEN];
561       char line[MAX_LINE_LEN];
562      // Process Books
563      int bookCount = Number_Books(booksFile);
564      int memberCount = Number_members(membersFile);
565      // Read the first header
566      if (fgets(header, sizeof(header), stdin) && strncmp(header, "Books:", 6) == 0) {
567
568          // Read the first line after the header
569          while (fgets(line, sizeof(line), stdin)) {
570              // Check if the current line is "Members:"
571              if (strncmp(line, "Members:", 8) == 0) {
572                  break;
573              }
574               if (bookCount >= 50) {
575                  printf("Error: Maximum number of books (50) reached.\n");
576                  continue; // Skip further input
577              }
578
579              // Process the current book entry
580              int id, copies;
581              if (sscanf(line, "%d %d", &id, &copies) == 2) {
582                  addBookToFile(booksFile, id, copies);
583                    bookCount++;
584
585              } else {
586                  printf("Invalid book entry. Please try again.\n");
587              }
588          }
589      }
590
591      fflush(stdin);
```

➔This is the first part of the main function that asks the user for the input

FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
592         // Process Members
593     if (strncmp(line, "Members:", 8) == 0) {
594         while (fgets(line, sizeof(line), stdin)) {
595             if (strncmp(line, "Borrowed Books:", 15) == 0) {
596                 break;
597             }
598              if (memberCount >= 30) {
599                 printf("Error: Maximum number of members (30) reached.\n");
600                 continue; // Skip further input
601             }
602
603
604             int memberID;
605             if (sscanf(line, "%d", &memberID) == 1) {
606                 addMemberToFile(membersFile, memberID);
607                memberCount++;
608
609             } else {
610                 printf("Invalid member entry. Please try again.\n");
611             }
612         }
613     }
614
615     //process loans
616     if (strncmp(line, "Borrowed Books:", 15) == 0) {
617         while (fgets(line, sizeof(line), stdin)) {
618             int bookID, memberID;
619             char date[11];
620             if (sscanf(line, "%d %d %10s", &bookID, &memberID, date) == 3) {
621                 addLoan(loansFile,membersFile,booksFile, bookID, memberID, date);
622             }
623             else if (feof(stdin)) {
624                 break; // Stop if EOF is encountered
625             }
626              else {
627                 printf("Invalid borrowed book entry. Please try again.\n");
628             }
629         }
630     }
631
```

FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

```
632     char operation[MAX_LINE_LEN];
633     while (1)
634     {scanf("%s", operation);
635         if (strcmp(operation, "Number_Books") == 0) {
636             printf("%d\n", Number_Books(booksFile));
637         } else if (strcmp(operation, "Number_Members") == 0) {
638             printf("%d\n", Number_members(membersFile));
639         } else if (strcmp(operation, "Book_ID_Min") == 0) {
640             Book_ID_Min(booksFile);
641         } else if (strcmp(operation, "Books_Available") == 0) {
642             Books_Available(booksFile);
643         } else if (strcmp(operation, "List_Book_Borrowers") == 0) {
644             int bookID;
645             scanf("%d", &bookID);
646             List_Book_Borrowers(loansFile, bookID);
647         } else if (strcmp(operation, "List_Member_Books") == 0) {
648             int memberID;
649             scanf("%d", &memberID);
650             List_Member_Books(loansFile, memberID);
651         } else if (strcmp(operation, "Most_Borrowed") == 0) {
652             Most_Borrowed(booksFile);
653         } else if (strcmp(operation, "Members_Less") == 0) {
654             int n;
655             scanf("%d", &n);
656             Members_Less(membersFile, n);
657         } else if (strcmp(operation, "Books_Unborrowed") == 0) {
658             Books_Unborrowed(booksFile);
659         } else if (strcmp(operation, "Books_Borrowed_Days") == 0) {
660             printf("%d\n", Books_Borrowed_Days(loansFile));
661         }
662         else if (strcmp(operation, "Books_Per_Member") == 0) {
663             Books_Per_Member(membersFile);
664         }
665         else if (strcmp(operation, "Overlapping_Borrowers") == 0){
666             int bookID;
667             scanf("%d", &bookID);
668             Overlapping_Borrowers(loansFile, bookID);
669         }
670         else if (strcmp(operation, "Quit") == 0) {
671                 printf("Thanks!");
672             break;
673         }
674          else {
675             printf("Invalid operation name. Please try again.\n");
676         }
677     }
678
679 }
680
```

➔This is the second part of the main function that asks the member for operations

FACULTY OF ENGINEERING

جامعة مصر للمعلوماتية
EGYPT UNIVERSITY OF INFORMATICS

Thank You

Noureen Ibrahim mohamed hassan	22-101056
Salma Hamdy Abdel-Fattah Yassin Adam	24-101489
Mariam Ashraf Abdelrahman Ahmed	24-101500
Hams Said Hussein Mohamed	24-101485
Shahd Yasser Mohamed Elazab	24-101493
Nayera Abdeltawab Abdelghany Elnoby	24-101495