Advanced C Lab DA-3

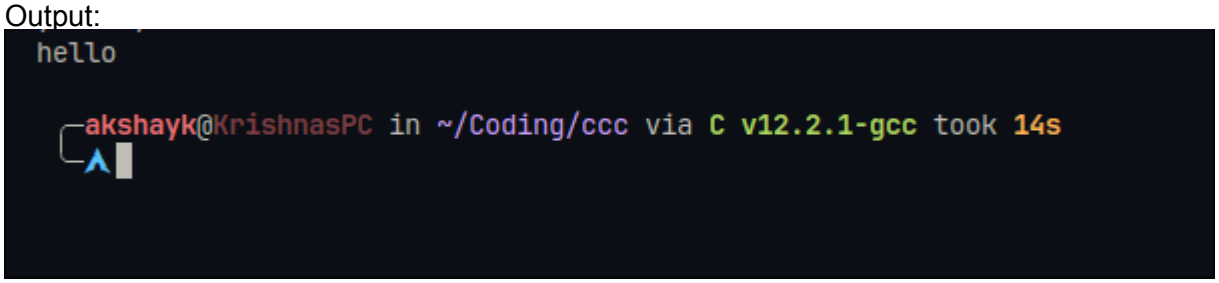
20MIC0023
Akshay Pranav Kalathil

Q1)write a c program to read 3 lines of text messages from the user and write into a binary file

```c
#include <stdio.h>
int main() {
    char messages[3][50];
    FILE *binary_file;
    int i;
    // Read 3 lines of text messages from user
    printf("Enter 3 lines of text messages:\n");
    for (i = 0; i < 3; i++) {
        fgets(messages[i], 50, stdin);
    }
    // Open the binary file for writing
    binary_file = fopen("messages.bin", "wb");
    if (binary_file == NULL) {
        printf("Error: Could not open file for writing.");
        return 1;
    }
    // Write the messages into the binary file
    fwrite(messages, sizeof(messages), 1, binary_file);
    // Close the binary file
    fclose(binary_file);
    // Open the binary file for reading
    binary_file = fopen("messages.bin", "rb");
    if (binary_file == NULL) {
        printf("Error: Could not open file for reading.");
        return 1;
    }
    // Read the messages from the binary file and display on the
screen
    printf("\nMessages from binary file:\n");
    fread(messages, sizeof(messages), 1, binary_file);
    for (i = 0; i < 3; i++) {
        printf("%s", messages[i]);
    }
    // Close the binary file
```

```
    fclose(binary_file);
    return 0;
```

Output:



Q2) Check If Students Failed OR Passes And Write In File

```
#include <stdio.h>
#define MAX_STUDENTS 40
struct Student {
    int slno;
    char name[50];
    int marks[3];
    int average;
    char grade;
};
void write_student_data(struct Student students[], int
num_students) {
    FILE *fp = fopen("input.dat", "w");
    if (fp == NULL) {
        printf("Error: Could not open input.dat for writing.\
n");
        return;
    }
    fprintf(fp, "slno stname marks[0] marks[1] marks[2] average
grade\n");
    for (int i = 0; i < num_students; i++) {
        fprintf(fp, "%d %s %d %d %d %d %c\n",
                students[i].slno,
                students[i].name,
                students[i].marks[0],
                students[i].marks[1],
                students[i].marks[2],
                students[i].average,
```

```
                students[i].grade);
    }
    fclose(fp);
}
void write_pass_fail_data(struct Student students[], int
num_students) {
    FILE *input_fp = fopen("input.dat", "r");
    if (input_fp == NULL) {
        printf("Error: Could not open input.dat for reading.\
n");
        return;
    }
    FILE *pass_fp = fopen("pass.dat", "w");
    if (pass_fp == NULL) {
        printf("Error: Could not open pass.dat for writing.\n");
        fclose(input_fp);
        return;
    }
    FILE *fail_fp = fopen("fail.dat", "w");
    if (fail_fp == NULL) {
        printf("Error: Could not open fail.dat for writing.\n");
        fclose(input_fp);
        fclose(pass_fp);
        return;
    }
    // Read and discard the header line
    char header[100];
    fscanf(input_fp, "%[^\n]\n", header);
    for (int i = 0; i < num_students; i++) {
        int slno, marks[3], average;
        char name[50], grade;
        fscanf(input_fp, "%d %s %d %d %d %d %c", &slno, name,
&marks[0], &marks[1], &marks[2], &average, &grade);
        students[i].slno = slno;
        strcpy(students[i].name, name);
        students[i].marks[0] = marks[0];
        students[i].marks[1] = marks[1];
        students[i].marks[2] = marks[2];
        students[i].average = average;
        students[i].grade = grade;
        if (grade == 'F') {
            fprintf(fail_fp, "%d %s %d %d %d %d %c\n", slno,
name, marks[0], marks[1], marks[2], average, grade);
        } else {
```

```
            fprintf(pass_fp, "%d %s %d %d %d %d %c\n", slno,
name, marks[0], marks[1], marks[2], average, grade);
        }
    }
    fclose(input_fp);
    fclose(pass_fp);
    fclose(fail_fp);
}
int main() {
    struct Student students[MAX_STUDENTS];
    int num_students;
    printf("Enter the number of students to read (max %d): ",
MAX_STUDENTS);
    scanf("%d", &num_students);
    if (num_students > MAX_STUDENTS) {
        printf("Error: Too many students to read.\n");
    }


   if (num_students ⩽ 0) {
    printf("Error: Invalid number of students.\n");
    return 1;
}
for (int i = 0; i < num_students; i++) {
    printf("Enter details of student %d:\n", i+1);
    printf("  Serial number: ");
    scanf("%d", &students[i].slno);
    printf("  Name: ");
    scanf("%s", students[i].name);
    int total_marks = 0;
    for (int j = 0; j < 3; j++) {
        printf("  Mark %d: ", j+1);
        scanf("%d", &students[i].marks[j]);
        total_marks += students[i].marks[j];
    }
    students[i].average = total_marks / 3;
    if (students[i].average ⩾ 60) {
        students[i].grade = 'A';
    } else if (students[i].average ⩾ 50) {
        students[i].grade = 'B';
    } else if (students[i].average ⩾ 40) {
        students[i].grade = 'C';
    } else {
        students[i].grade = 'F';
```

```
        }
    }
    write_student_data(students, num_students);
    write_pass_fail_data(students, num_students);
    printf("Student data written to input.dat\n");
    printf("Passing students data written to pass.dat\n");
    printf("Failing students data written to fail.dat\n");
    return 0;


        }
```

OutPut:

```
  Mark 3: 45
Enter details of student 4:
  Serial number: 324
  Name: Aksha
  Mark 1: 45
  Mark 2: 32
  Mark 3: 12
Student data written to input.dat
Passing students data written to pass.dat
Failing students data written to fail.dat

akshayk@KrishnasPC in ~/Coding/ccc via C v12.2.1-gcc took 46s
```

```
789 SAFSG 67 56 45 56 B
324 Aksha 45 32 12 29 F
```

```
789 SAFSG 67 56 45 56 B
```

```
324 Aksha 45 32 12 29 F
```

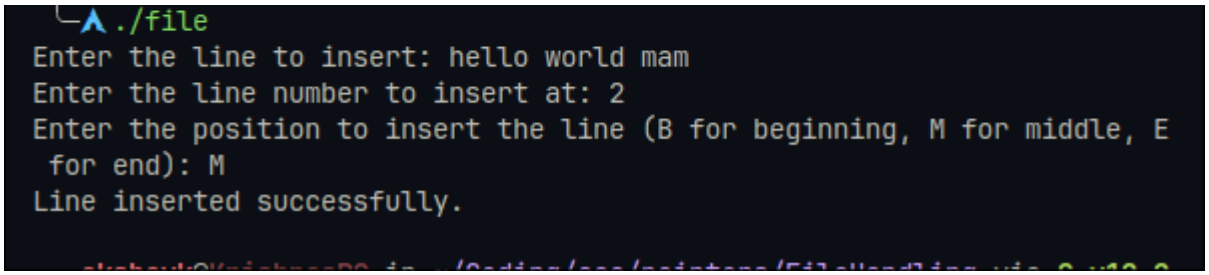
Q3)Insert Line In FIle

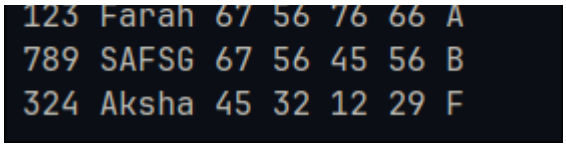
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_LINE_LENGTH 1000
int get_line_number() {
    int n;
    printf("Enter the line number to insert at: ");
    if (scanf("%d", &n) != 1) {
        printf("Invalid input. Please enter an integer.\n");
        exit(1);
    }
    getchar(); // remove the newline character from the input
buffer
    return n;
}
char get_insert_position() {
    char position;
    printf("Enter the position to insert the line (B for
beginning, M for middle, E for end): ");
    if (scanf(" %c", &position) != 1) {
        printf("Invalid input. Please enter B, M or E.\n");
        exit(1);
    }
    return position;
}
void insert_line(FILE *input_file, const char *line, int
line_number, char position) {
    // Create a temporary file
    FILE *temp_file = fopen("temp.txt", "w");
    if (temp_file == NULL) {
        printf("Error opening temporary file.\n");
        exit(1);
    }
    // Insert the line at the specified position
    char buffer[MAX_LINE_LENGTH];
    int current_line_number = 1;
    while (fgets(buffer, MAX_LINE_LENGTH, input_file) != NULL) {
        if (current_line_number == line_number && position ==
'M') {
            fprintf(temp_file, "%s\n", line);
        }
        fprintf(temp_file, "%s", buffer);
```

```
            current_line_number++;
        }
        if (line_number >= current_line_number && position != 'E') {
            printf("Invalid line number. Please enter a number
between 1 and %d.\n", current_line_number - 1);
            remove("temp.txt");
            exit(1);
        }
        if (position == 'B' || (position == 'M' && line_number ==
1)) {
            fprintf(temp_file, "%s\n", line);
        }
        if (position == 'E' || line_number == current_line_number) {
            fprintf(temp_file, "%s\n", line);
        }
        // Close the files
        fclose(input_file);
        fclose(temp_file);
        // Replace the input file with the temporary file
        if (remove("input.dat") != 0) {
            printf("Error deleting input file.\n");
            exit(1);
        }
        if (rename("temp.txt", "input.dat") != 0) {
            printf("Error renaming temporary file.\n");
            exit(1);
        }
}
int main() {
        // Open the input file
        FILE *input_file = fopen("input.dat", "r");
        if (input_file == NULL) {
            printf("Error opening input file.\n");
            return 1;
        }
        // Read the line to insert
        char line[MAX_LINE_LENGTH];
        printf("Enter the line to insert: ");
        if (fgets(line, MAX_LINE_LENGTH, stdin) == NULL) {
            printf("Error reading input.\n");
            fclose(input_file);
            return 1;
        }
```

```
    line[strcspn(line, "\n")] = '\0'; // remove the newline
character
    // Read the line number to insert at
    int line_number = get_line_number();
    // Read the insert position
    char position = get_insert_position();
    // Insert the line
    insert_line(input_file, line, line_number, position);
    printf("Line inserted successfully.\n");
    return 0;
}
```

Output:



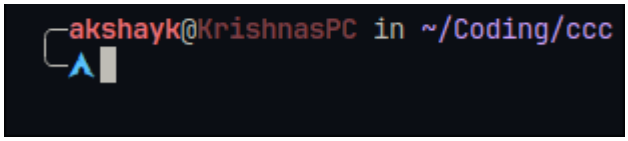


Q4)Macro To Find Even Number

```
#include<stdio.h>
#define even(x)({if(x%2==0){printf("It
isEven");}else{printf("Itis Odd");}})
int main(){
    int a;
    scanf("%d",&a);
    even(a);

    return 0;
}
```

OutPut:



Q5)Macro To Find Greatest Of 4 Numbers

```c
#include <stdio.h>
#define MAX(x, y) ((x) > (y) ? (x) : (y))
int main() {
    int a = 5, b = 8, c = 2, d = 3;
    int left_max = MAX(a, b);
    int right_max = MAX(c, d);
    int final_max = MAX(left_max, right_max);
    printf("Maximum number is: %d", final_max);
    return 0;
}
```

OutPut: