

Question 1: Declare a structure that represents the following hierarchical information: to Remember (a) Student (b) Roll Number (c) Name (Typedef) (i) First name (ii) Middle Name (iii) Last Name (d) Gender (e) Date of Birth (Typedef) (i) Day (ii) Month (iii) Year (f) Marks (typedef) (i) English (ii) Mathematics (iii) Computer Science

```
// Q.1 Declare a structure that represents the following hierarchical information:
// to Remember

// (a) Student
// (b) Roll Number
// (c) Name (Typedef)
// (i) First name
// (ii) Middle Name
// (iii) Last Name
// (d) Gender
// (e) Date of Birth (Typedef)
// (i) Day
// (ii) Month
// (iii) Year
// (f) Marks (typedef)
// (i) English
// (ii) Mathematics
// (iii) Computer Science

typedef struct Name{
    char *firstName;
    char *middleName;
    char *lastName;
} Name;

typedef struct Date{
    int day;
    int month;
    int year;
} Date;

typedef struct Marks{
    int english;
    int mathematics;
    int computerScience;
} Marks;

typedef struct Student{
    int rollNumber;
    Name name;
    char gender;
    Date DOB;
    Marks marks;
} Student;
```

Question 2: Using the above structure, write a program to display the details of the student whose name is entered by the user. Display the name of the students who have secured less than 40% of the aggregate. In addition, print each student's average marks.

Header.h

```
typedef struct Name{
    char *firstName;
    char *middleName;
    char *lastName;
} Name;

typedef struct Date{
    int day;
    int month;
    int year;
} Date;

typedef struct Marks{
    int english;
    int mathematics;
    int computerScience;
} Marks;

typedef struct Student{
    int rollNumber;
    Name name;
    char gender;
    Date DOB;
    Marks marks;
} Student;

float getAverageMarks(Marks marks);
void printDetails(Student Student);
```

Logic.c

```
#include "../header.h"

float getAverageMarks(Marks marks){
    return (float)(marks.english + marks.mathematics + marks.computerScience) / 3;
}

void printDetails(Student Student){
    printf("Rollnumber: %d\n", Student.rollNumber);
    printf("Name: %s %s %s\n", Student.name.firstName, Student.name.middleName,
Student.name.lastName);
    printf("Gender: %c\n", Student.gender);
    printf("DOB: %d/%d/%d\n", Student.DOB.day, Student.DOB.month, Student.DOB.year);
    printf("Marks: English: %d Mathematics: %d Computer Science: %d\n\n", Student.marks.english,
Student.marks.mathematics, Student.marks.computerScience);
}
```

Main.c

```
// 0.2 Using the structure from Question 1, write a program to display the details of the student
// whose
// name is entered by the user. Display the name of the students who have secured less than
// 40% of the aggregate. In addition, print each student's average marks.

#include <stdio.h>
#include <string.h>
#include "../logic.c"

#define MAX_STUDENTS 10
#define MAX_NAME_SIZE 20
int main(){
    Student Students[MAX_STUDENTS];
    int studentIndex = 0;
    char name[MAX_NAME_SIZE];

    Students[studentIndex].rollNumber = 1;
    Students[studentIndex].name = (Name){ "Mehmood", "Rehan", "Deshmukh" };
    Students[studentIndex].gender = 'M';
    Students[studentIndex].DOB = (Date){ 3, 9, 2005 };
    Students[studentIndex].marks = (Marks){ 85.7, 82.0, 94.3 };
    studentIndex++;

    Students[studentIndex].rollNumber = 2;
    Students[studentIndex].name = (Name){ "Yashwant", "Chandrakant", "Bhosale" };
    Students[studentIndex].gender = 'F';
    Students[studentIndex].DOB = (Date){ 1, 11, 2001 };
    Students[studentIndex].marks = (Marks){ 95.0, 89.5, 100 };
    studentIndex++;

    Students[studentIndex].rollNumber = 3;
    Students[studentIndex].name = (Name){ "Deven", "Arunn", "Shinde" };
    Students[studentIndex].gender = 'M';
    Students[studentIndex].DOB = (Date){ 14, 11, 2005 };
    Students[studentIndex].marks = (Marks){ 32.7, 65.3, 22.3 };
    studentIndex++;

    printf("Enter name of student to get Details:\n");
    scanf("%s", name);
    int found = 0;
    for(int i=0; i<studentIndex; i++){
        if(strcmp(Students[i].name.firstName, name) == 0){
            found = 1;
            printDetails(Students[i]);
        }
    }

    if(!found) printf("Student with name %s not found\n", name);

    printf("**Average marks for each student**\n");

    for (int i = 0; i < studentIndex; ++i) {
        float averageMarks = getAverageMarks(Students[i].marks);
        float aggregatePercentage = averageMarks;

        printf("%s %s %s has an average of %.2f%% marks\n",
            Students[i].name.firstName, Students[i].name.middleName, Students[i].name.lastName,
            averageMarks);

        if (aggregatePercentage < 40) {
            printf("\n%s %s %s having the Roll Number %d has an average of %.2f%% marks and has
            secured less than 40%% aggregate.\n",
                Students[i].name.firstName, Students[i].name.middleName, Students[i].name.lastName,
                Students[i].rollNumber, averageMarks);
        }
    }
    return 0;
}
```

Output:

```
Question 3>cd '..\Question 2\'
Question 2>gcc .\main.c -Wall -o main
Question 2>.\main.exe
Enter name of student to get Details:
Mehmood
Rollnumber: 1
Name: Mehmood Rehan Deshmukh
Gender: M
DOB: 3/9/2005
Marks: English: 85 Mathematics: 82 Computer Science: 94

**Average marks for each student**
Mehmood Rehan Deshmukh has an average of 87.00% marks
Yashwant Chandrakant Bhosale has an average of 94.67% marks
Deven Arunn Shinde has an average of 39.67% marks

Deven Arunn Shinde having the Roll Number 3 has an average of 39.67% marks and has secured less than 40% aggregate.
Question 2>
```

Question 3: Write a program to define a structure for a hotel that has members— name, address, grade, number of rooms, and room charges. Write a function to print the names of hotels in a particular grade. Also write a function to print names of hotels that have room charges less than the specified value.

Header.h

```
typedef struct Hotel{
    char name[50];
    char address[100];
    char grade;
    int numRooms;
    int roomCharges;
} Hotel;

void printHotelsByGrade(Hotel hotels[], int numHotels, char grade);
void printHotelsByRoomCharges(Hotel hotels[], int numHotels, int roomCharges);
```

Logic.c

```

#include "../header.h"

void printHotelsByGrade(Hotel hotels[], int numHotels, char grade){
    printf("\n**Hotels with Grade %c**\n", grade);
    for(int i = 0 ; i < numHotels; i++){
        if(hotels[i].grade == grade){
            printf("Hotel: %s\n", hotels[i].name);
        }
    }
}

void printHotelsByRoomCharges(Hotel hotels[], int numHotels, int roomCharges){
    printf("\n**Hotels with charges less than %d**\n", roomCharges);
    for(int i = 0 ; i < numHotels; i++){
        if(hotels[i].roomCharges <= roomCharges){
            printf("Hotel: %s\n", hotels[i].name);
        }
    }
}

```

Main.c

```

// Q.3 Write a program to define a structure for a hotel that has members- name, address,
// grade, number of rooms, and room charges. Write a function to print the names of hotels in
// a particular grade. Also write a function to print names of hotels that have room charges less
// than the specified value.

#include <stdio.h>
#include "../logic.c"
int main() {
    Hotel hotels[] = {
        {"Hotel 1", "Address 1", 'A', 20, 2000},
        {"Hotel 2", "Address 2", 'B', 45, 1200},
        {"Hotel 3", "Address 3", 'A', 25, 3000},
        {"Hotel 4", "Address 4", 'A', 60, 4500}
    };

    int numHotels = sizeof(hotels) / sizeof(hotels[0]);

    printHotelsByGrade(hotels, numHotels, 'A');
    printHotelsByRoomCharges(hotels, numHotels, 4000);
    return 0;
}

```

Output:

```
Question 3>gcc .\main.c -Wall -o main
Question 3>.\main.exe

**Hotels with Grade A**
Hotel: Hotel 1
Hotel: Hotel 3
Hotel: Hotel 4

**Hotels with charges less than 4000**
Hotel: Hotel 1
Hotel: Hotel 2
Hotel: Hotel 3
Question 3>
```

Question 4: Declare a structure time that has three fields—hr, min, sec. Create two variables start_time and end_time. Input their values from the user. Then while start_time does not reach the end_time, display GOOD DAY on the screen.

Header.h

```
typedef struct Time {
    int hours;
    int minutes;
    int seconds;
} Time;

void printGoodDay(Time startTime, Time endTime);
```

Logic.c

```

#include "../header.h"
#include <stdio.h>

int isValid(Time startTime, Time endTime){
    if(startTime.hours > endTime.hours){
        return 0;
    }else if(startTime.hours == endTime.hours && startTime.minutes > endTime.minutes){
        return 0;
    }else if(startTime.minutes == endTime.minutes && startTime.seconds > endTime.seconds){
        return 0;
    }

    return 1;
}

void printGoodDay(Time startTime, Time endTime){
    if(!isValid(startTime, endTime)) return;

    while(startTime.hours < endTime.hours || (startTime.hours == endTime.hours && startTime.minutes <
endTime.minutes) || (startTime.hours == endTime.hours && startTime.minutes == endTime.minutes &&
startTime.seconds < endTime.seconds)){
        printf("GOOD DAY!\n");
        startTime.seconds++;

        if(startTime.seconds == 60){
            startTime.seconds = 0;
            startTime.minutes++;
        }

        if(startTime.minutes == 60){
            startTime.minutes = 0;
            startTime.hours++;
        }
    }
}

```

Main.c

```

// Q.4 Declare a structure time that has three fields-hr, min, sec. Create two variables start_time
// and end_time. Input their values from the user. Then while start_time does not reach the
// end_time, display GOOD DAY on the screen.

#include "../logic.c"
#include <stdio.h>

int main() {
    Time startTime, endTime;
    printf("Enter start time in hours minutes and seconds :\n");
    scanf("%d %d %d", &startTime.hours, &startTime.minutes, &startTime.seconds);

    printf("Enter end time in hours minutes and seconds :\n");
    scanf("%d %d %d", &endTime.hours, &endTime.minutes, &endTime.seconds);

    printGoodDay(startTime, endTime);

    return 0;
}

```

Output:

```
Question 4>gcc .\main.c -Wall -o main
Question 4>.\main.exe
Enter start time in hours minutes and seconds :
12 12 20
Enter end time in hours minutes and seconds :
12 12 30
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
GOOD DAY!
Question 4>
```

Q.5 Declare a structure fraction that has two fields— numerator and denominator. Create two variables and compare them using function. Return 0 if the two fractions are equal, -1 if the first fraction is less than the second and 1 otherwise. You may convert a fraction into a floating point number for your convenience.

Header.h

```
typedef struct Fraction{
    int numerator;
    int denominator;
} Fraction;

int compare(Fraction x, Fraction y);
```


Logic.c

```
#include "../header.h"

int compare(Fraction x, Fraction y){
    double num1 = (double) x.numerator / x.denominator;
    double num2 = (double) y.numerator / y.denominator;

    if(num1 == num2){
        return 0;
    } else if(num1 < num2){
        return -1;
    } else{
        return 1;
    }
}
```

Main.c

```
#include <stdio.h>
#include "../logic.c"

int main(){
    Fraction num1 = (Fraction){10, 5};
    Fraction num2 = (Fraction){18, 6};

    printf("%d\n", compare(num1, num2));
    return 0;
}
```

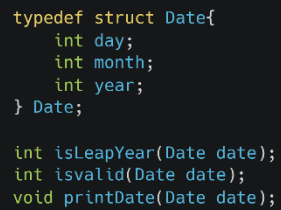
Output:

```
Question 5>gcc .\main.c -Wall -o main
Question 5>.\main.exe
-1
Question 5>
```

Question 6: Define a structure date containing three integers— day, month, and year. Write a program using functions to read data, to validate the date entered by the user and then print the date on the screen. For example, if you enter 29/2/2010 then that is an invalid

date as 2010 is not a leap year. Similarly 31/6/2007 is invalid as June does not have 31 days.

header.h



```
typedef struct Date{
    int day;
    int month;
    int year;
} Date;

int isLeapYear(Date date);
int isvalid(Date date);
void printDate(Date date);
```

logic.c

```
#include "../header.h"

int isLeapYear(Date date){
    int year = date.year;

    return (year % 400 == 0) || (year % 4 == 0 && year % 100 != 0);
}

int isValid(Date date){
    int day = date.day;
    int month = date.month;
    int isLeap = isLeapYear(date);

    if (month < 1 || month > 12) {
        return 0;
    }

    if (day < 1){
        return 0;
    }

    switch (month){
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:
            return (day <= 31);
        case 4:
        case 6:
        case 9:
        case 11:
            return (day <= 30);
        case 2:
            return (isLeap ? day <= 29 : day <= 28);
        default:
            return 0;
    }
}

void printDate(Date date){
    printf("Date: %02d/%02d/%04d\n", date.day, date.month, date.year);
}
```

main.c

```

#include <stdio.h>
#include "../logic.c"
int main() {
    Date date;

    printf("Enter the date in (dd/mm/yyyy) format : \n");
    scanf("%d %d %d", &date.day, &date.month, &date.year);

    if(isvalid(date)){
        printf("Valid date entered!\n");
        printDate(date);
    }else{
        printf("Invalid date entered!\n");
    }

    return 0;
}

```

Output:

```

Question 6>gcc ../main.c -Wall -o main
Question 6>../main.exe
Enter the date in (dd/mm/yyyy) format :
29 2 2016
Valid date entered!
Date: 29/02/2016
Question 6>

```

Question 7: Write a program to add and subtract 10hrs 20mins 50sec and 5hrs 30min 40sec

header.h

```

typedef struct Time{
    int hours;
    int minutes;
    int seconds;
} Time;

Time add(Time t1, Time t2);
Time subtract(Time t1, Time t2);

```

logic.c

```
#include "../header.h"

Time add(Time t1, Time t2){
    Time result;
    int totalSeconds1 = t1.hours * 3600 + t1.minutes * 60 + t1.seconds;
    int totalSeconds2 = t2.hours * 3600 + t2.minutes * 60 + t2.seconds;

    int totalSeconds = totalSeconds1 + totalSeconds2;
    result.hours = totalSeconds / 3600;
    result.minutes = (totalSeconds % 3600) / 60;
    result.seconds = totalSeconds % 60;
    return result;
}

Time subtract(Time t1, Time t2){
    Time result;
    int totalSeconds1 = t1.hours * 3600 + t1.minutes * 60 + t1.seconds;
    int totalSeconds2 = t2.hours * 3600 + t2.minutes * 60 + t2.seconds;
    int totalSeconds = totalSeconds1 - totalSeconds2;
    result.hours = totalSeconds / 3600;
    result.minutes = (totalSeconds % 3600) / 60;
    result.seconds = totalSeconds % 60;
    return result;
}
```

main.c

```
#include "../logic.c"
#include <stdio.h>

int main(){
    Time t1 = (Time){10, 20, 50};
    Time t2 = (Time){5, 30, 40};

    Time added = add(t1, t2);
    Time subtracted = subtract(t1, t2);
    printf("Sum: %d hrs %d mins %d s\n", added.hours, added.minutes, added.seconds);
    printf("Difference: %d hrs %d mins %d s\n", subtracted.hours, subtracted.minutes,
    subtracted.seconds);
    return 0;
}
```

Output:

```
Question 7>gcc .\main.c -Wall -o main
Question 7>.\main.exe
Sum: 15 hrs 51 mins 30 s
Difference: 4 hrs 50 mins 10 s
Question 7>
```