FITNESS TRACKER

NAME: PARMIDA SAHNI
SAP ID: 590026308
BATCH: 37
SUBJECT: PROGRAMMING IN C

ABSTRACT

The project is a fitness tracker. It calculates the user's BMI. Moreover, it provides the user with a list of choices of activities like- running, walking, swimming, cycling and hiking. The user may choose one or more activities in order to get the output. The user inputs either distance covered or steps or time taken to complete the respective activity. It outputs the calories burnt, suggestions for warm up and cool down depending on the activity. This program displays the real world applications of C such as loops, structures, functions, file handling etc.

PROBLEM DEFINITION

Plenty of people don't have a simple way to check how fit they're getting or see how many calories they burn each day. Most fitness apps out there are either too hard to use or won't work without the internet. What's missing is a small, straightforward app that figures out BMI while giving tailored calorie numbers per workout and suggesting suitable exercises. To fix this, we built a basic C program featuring BMI checks, choices for different activities, rough calorie counts, plus advice on warming up and cooling down - all done through an easy-to-navigate terminal screen.

SYSTEM DESIGN

ALGORITHM FOR main.c

1. Start
2. Define a structure Activity
3. Print “welcome to the fitness tracker”
4. Call function bmi()
5. Input choice
6. Start an infinite loop while (1)
   Print “enter your choice”
   >1. Running
   >2. Walking
   >3. Swimming
   >4. Cycling
   >5. Hiking

7. Start a switch (choice)
8. Call functions
   >1. Running()
   >2. Walking()
   >3. Swimming()
   >4. Cycling()
   >5. Hiking()
   >6. exit
   > default: print “invalid input”

9. End while loop
10. Print activity name, calories burnt.
11. Stop

ALGORITHM FOR functions.c

1. Start
2. Define function bmi()
3. Input height(m) , weight(kg)
4. bmi = weight/ (height * height)
5. Print bmi
6. if (bmi < 18.5)
   Print "OH NO! You're underweight :(“
   else if (bmi < 24.9)
   Print “Well done! You're healthy :D”
   else if (bmi < 29.9)
   Print “OH NO! You're overweight :(“
   else
   Print “OH NO! You're obese :O”
7. Define function running()
8. Input distance
9. Calories = distance * 60
10. Print calories, suggestions
11. Define function walking()
12. Input steps
13. calories= steps * 0.04
14. Print calories, suggestions
15. Define function swimming()
16. Input time
17. calories= time * 8
18. Print calories, suggestions
19. Define function hiking()
20. Input distance
21. Calories = distance * 50
22. Print calories, suggestions
23. Define function cycling()
24. Input distance
25. Calories = distance * 35
26. Print calories, suggestions

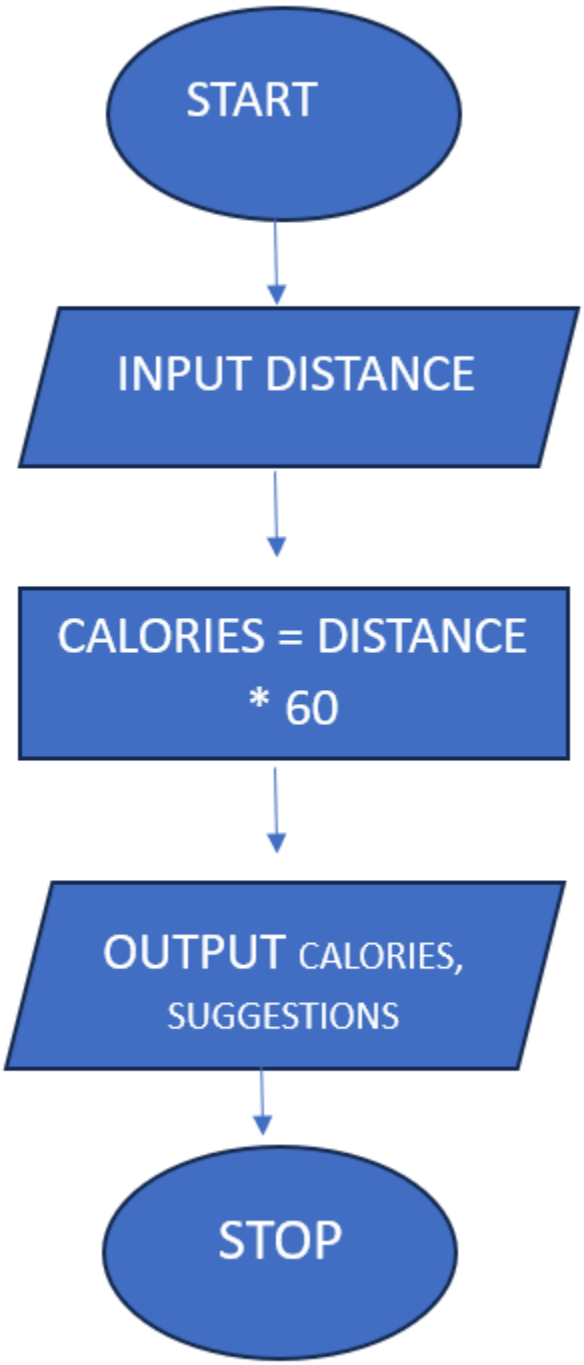
27. Stop

ALGORITHM FOR functions.h

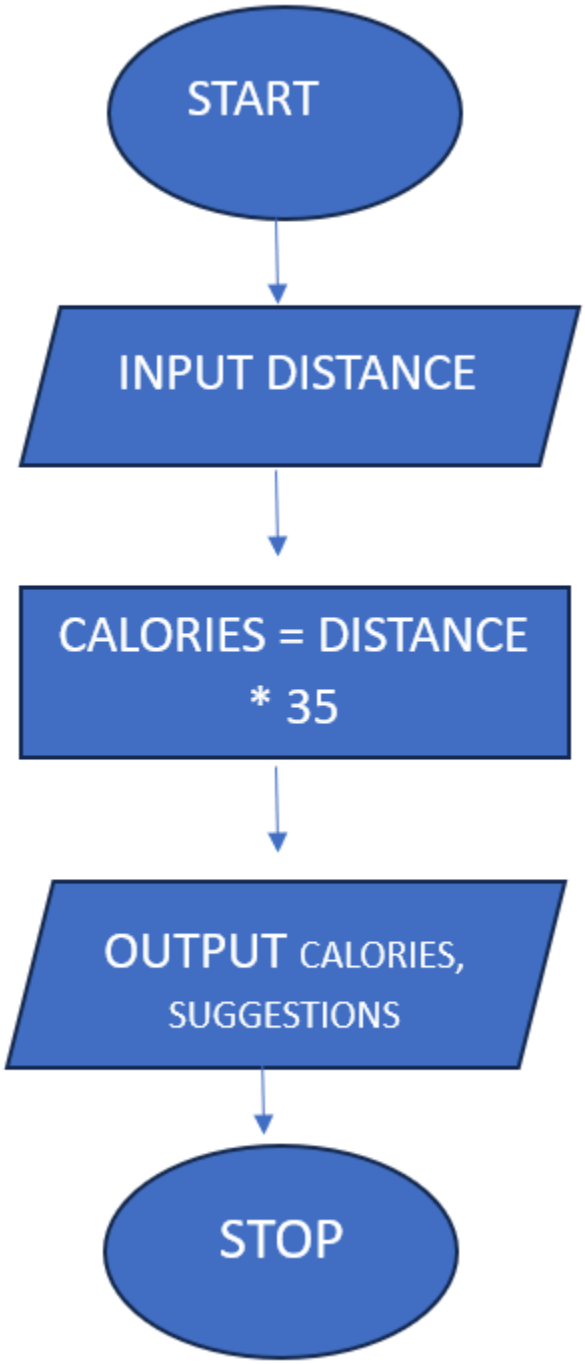
1. Start
2. Declare struct Activity
3. Declare functions
   >1. Running()
   >2. Walking()
   >3. Swimming()
   >4. Cycling()
   >5. Hiking()
   >6. bmi()
4. Stop

FLOWCHARTS

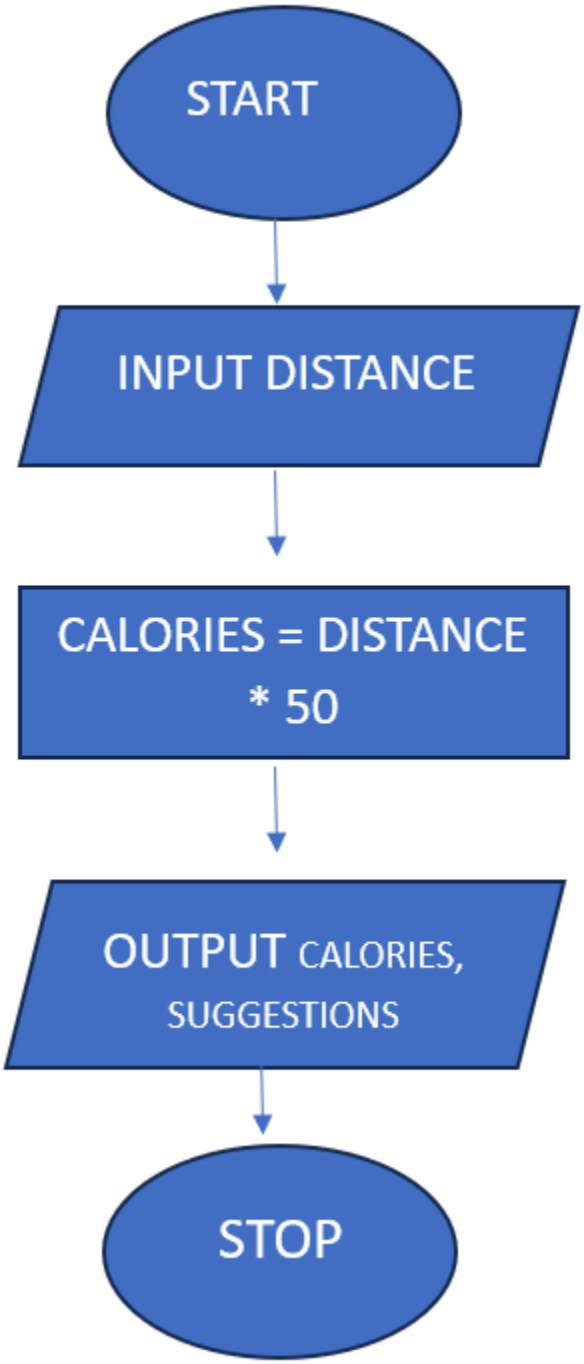
Void running()



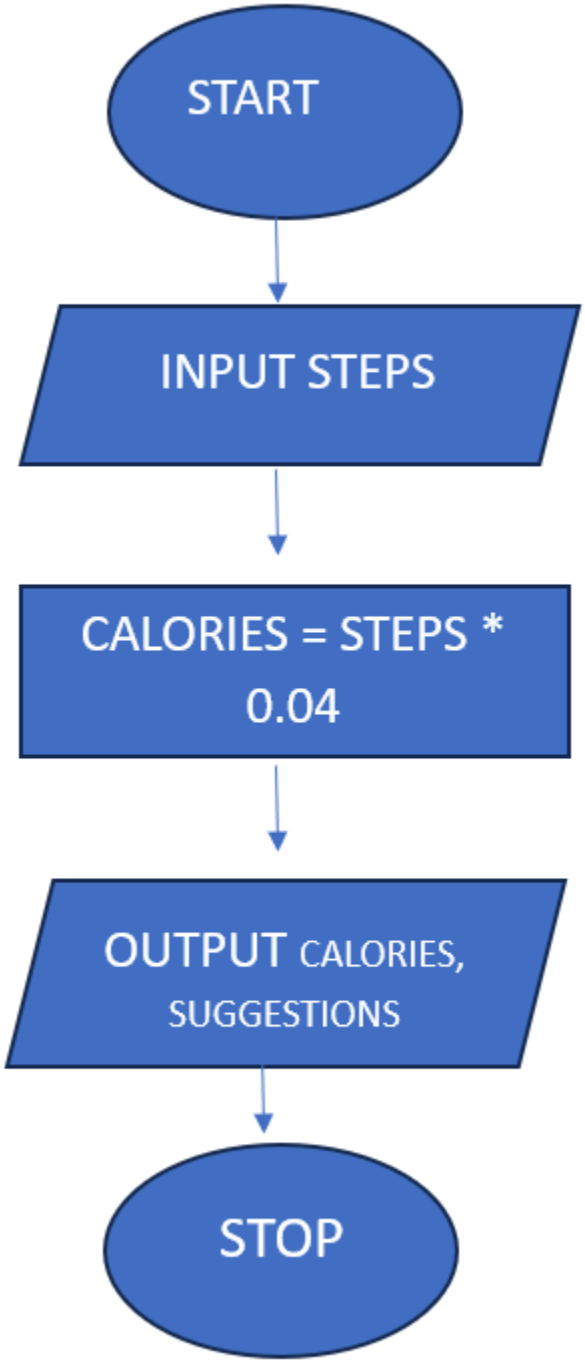
Void cycling()



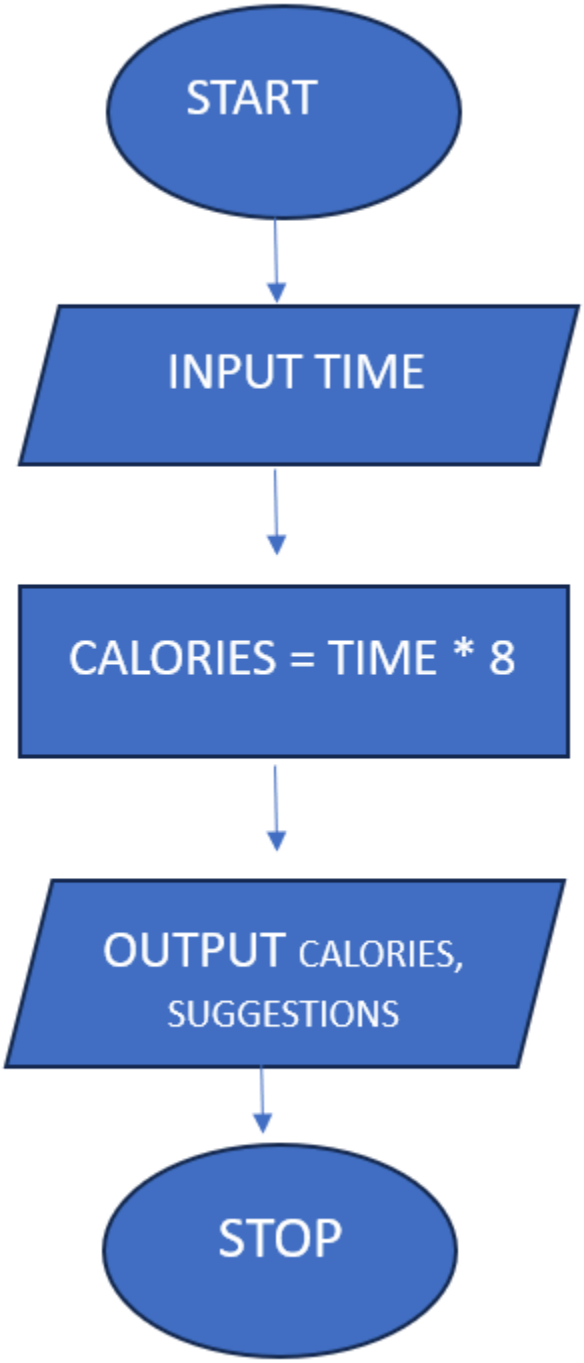
Void hiking()


START
INPUT DISTANCE
CALORIES = DISTANCE * 50
OUTPUT CALORIES, SUGGESTIONS
STOP

Void walking()



Void swimming ()



IMPLEMENTATION DETAILS

This section explains how the fitness tracker is built with the use of some snippets.

1. **Overall structure:**
   The program is divided into three parts
   >main.c - It contains the core of the program which handles the menu, input and overall flow.
   >functions.c - handles the logic of bmi calculation, calories and suggestions.
   >functions.h - a header file where all the functions are declared.

2. **BMI calculation:**
   The program first asks the user to input their height and weight and hence calculates their bmi.

```c
void bmi()
{
    float height, weight, bmi; //declaring local variables in bmi function.

    printf("\nEnter your height in meters.\n");
    scanf("%f", &height);
    printf("Enter your weight in Kg.\n");
    scanf("%f", &weight);

    bmi = weight / (pow(height, 2));

    printf("\nYour BMI is: %.2f\n", bmi); //we use %.2 so that it will print only upto 2 decimals.

    //bmi criterions.

    if (bmi < 18.5)
        printf("\nOH NO! You're underweight :(\n");
    else if (bmi < 24.9)
        printf("\nWell done! You're healthy :D\n");
    else if (bmi < 29.9)
        printf("\nOH NO! You're overweight :(\n");
    else
        printf("\nOH NO! You're obese :O\n");

    printf("\nNote: BMI is not a definitive measure of health for individuals because it doesn't distinguish between

}
```

3. **Activity selection menu:**
   Users may choose an activity from the following list of activities.

```
printf("\n===============FITNESS TRACKER===============\n");

printf("\n1. Running\n");
printf("\n2. Walking\n");
printf("\n3. Hiking\n");
printf("\n4. cycling\n");
printf("\n5. swimming\n");
printf("\n6. Exit\n");

printf("\nEnter the serial number of the activity you want to track: "); //the user inputs their choice.
scanf("%d", &choice);
```

4. **Calorie calculation:**
   Each activity has its own calorie calculation formula.
   For eg.
   For walking:

```
void walking (struct Activity *record)
{
    float steps;
    printf("\nEnter the number of steps: \n"); //taking input from the user.
    scanf("%f", &steps);

    record->calories = steps * 0.04;
```

5. **Suggestions for warm up and cool down:**
   The fitness tracker displays suggestions depending on the activity.
   For eg. for hiking

```
//function 6
void hiking (struct Activity *record)
{
    float distance;
    printf("\nEnter the distance covered in Km: \n"); //taking input from the user.
    scanf("%f", &distance);

    record->calories = distance * 50;
    strcpy(record->name, "Hiking");

    printf("\n================SUGGESTIONS================\n");
    printf("\nSome warm up tips:\n");
    printf("1. Dynamic stretches like leg swings, arm circles, and high knees to get your muscles and joints moving.\n");
    printf("2. Incorporate strength and mobility exercises such as squats, lunges, and ankle tilts to prepare for the physical demands of the trail.\n");

    printf("\nSome cool down tips:\n");
    printf("Do a light cardio like a 3-5 minute walk or jog to bring your heart rate down, followed by static stretches to improve flexibility and reduce muscle stiffness.\n");
}
```

6. **Program flow logic:**
   The program uses an infinite while loop in order to let the user use the program and select different activities multiple times in one run.

```
//infine loop for using the tracker multiple times.
while (1)
{
```

7. **Use of structures.**
   The program uses structures in a simple way to store the activity record.

```
5   //declaring a structure for the activity log.
6   struct Activity
7   {
8       char name[30];
9       float calories;
0   };
```

TESTING AND RESULTS

TEST METHODOLOGY:

CASE 1: BMI calculation:

```
Enter your height in meters.
1.65
Enter your weight in Kg.
60

Your BMI is: 22.04

Well done! You're healthy :D
```

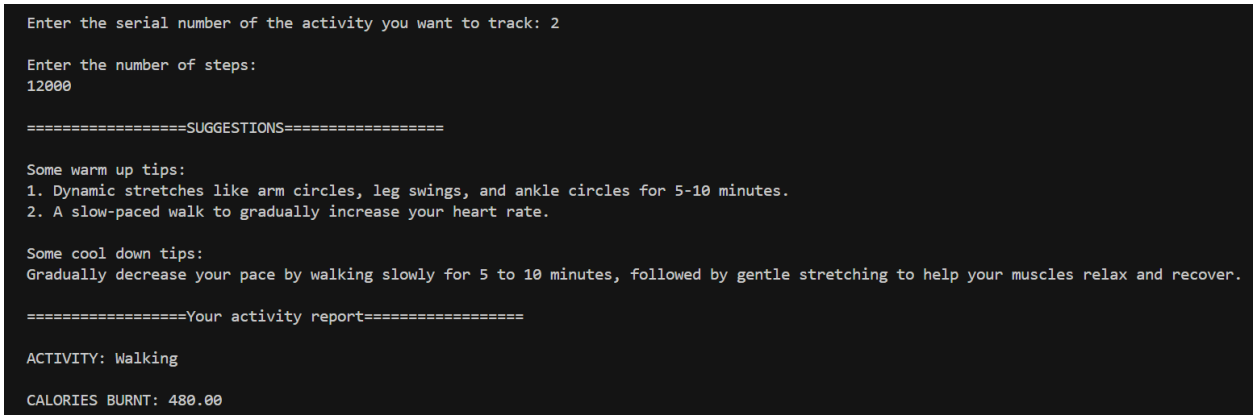
RESULT: It calculated the BMI properly and accurately.

CASE 2: Swimming activity:

```
Enter the serial number of the activity you want to track: 5

Enter time spent swimming in minutes:
60

==================SUGGESTIONS==================

Some warm up tips:
1. On land- Arm circles, shoulder rolls, and high knees.
2. In water- Light swimming, kicking, and drills that build up to more powerful strokes.

Some cool down tips:
light-intensity swimming, gentle dryland stretches like arm and shoulder circles, calf and hamstring stretches.

==================Your activity report==================

ACTIVITY: Swimming

CALORIES BURNT: 480.00
```

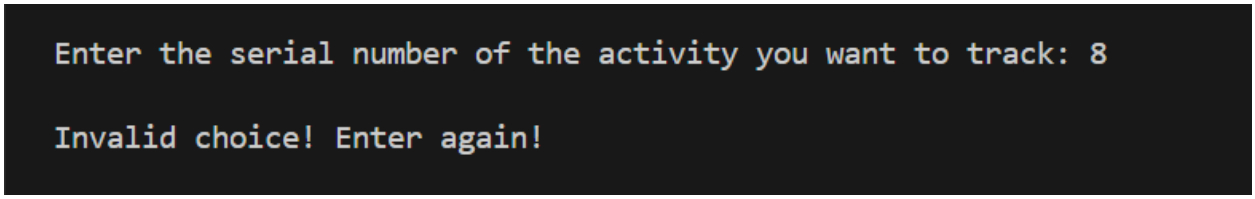
RESULT: Calculated the calories expectedly and displayed suggestions.

CASE 3: Walking activity:



RESULT: Calculated the calories expectedly and displayed suggestions.

CASE 4: Invalid menu choice



RESULT: Program handled invalid choice and re-displayed the menu.

SAMPLE OUTPUT:

```
~~~~Welcome to the fitness tracker~~~~

Enter your height in meters.
1.65
Enter your weight in Kg.
60

Your BMI is: 22.04

Well done! You're healthy :D

Note: BMI is not a definitive measure of health for individuals because it doesn't distinguish between muscle, bone, and fat.

==========FITNESS TRACKER==========

1. Running

2. Walking

3. Hiking

4. cycling

5. swimming

6. Exit

Enter the serial number of the activity you want to track: 3

Enter the distance covered in Km:
23

==========SUGGESTIONS==========
```

```
==========SUGGESTIONS==========

Some warm up tips:
1. Dynamic stretches like leg swings, arm circles, and high knees to get your muscles and joints moving.
2. Incorporate strength and mobility exercises such as squats, lunges, and ankle tilts to prepare for the physical demands of the trail.

Some cool down tips:
Do a light cardio like a 3-5 minute walk or jog to bring your heart rate down, followed by static stretches to improve flexibility and reduc
e muscle stiffness.

==========Your activity report==========

ACTIVITY: Hiking

CALORIES BURNT: 1150.00

==========FITNESS TRACKER==========

1. Running

2. Walking

3. Hiking

4. cycling

5. swimming

6. Exit

Enter the serial number of the activity you want to track: 6

Thank you for using the fitness tracker.
```

CONCLUSION

With this project, I gained skills in using C elements such as functions, loops, structs, also file operations to create a working app. It supports logging simple exercise data while offering a base that can grow over time.

FUTURE WORK

For future work, I can enhance it with the help of file handling which will store the user's data on a monthly or weekly basis. With this, the user can access and track their past activity logs. I can add a feature which will allow users to set a goal over a definite period of time and hence allow them to track their goals. The program will also generate a report monthly displaying the user's improvements, where they slacked and motivation to stay consistent.

REFERENCES

1. https://www.geeksforgeeks.org/
2. Let Us C - Yashavant Kanetkar
3. Class PPTs
4. Github