# DIET AND FITNESS PLANNER

A PROJECT REPORT

***Submitted by***

| | |
|---|---|
| BL.EN.U4AIE19007 | APOORVA.M |
| BL.EN.U4AIE19041 | TANUJ.M |
| BL.EN.U4AIE19068 | AISHWARYA.V |

***for the course***

***19AIE212- Design and Analysis of Algorithms***

***Guided and Evaluated by***

***DR. SUPRIYA M***

IN

ARTIFICIAL INTELLIGENCE ENGINEERING

AMRITA SCHOOL OF ENGINEERING, BANGALORE

**BANGALORE 560 035**

## ABSTRACT OF THE PROJECT

The application will focus in optimizing the diet and fitness plan for the user depending on his/her goal and customizing it according to his/her daily routine and habits. The user can select their choice of exercises and their choice of food category for breakfast, lunch and dinner like South Indian, North Indian, etc. Further we'll be taking to consideration other sub features such as availability of the fitness equipment, age, available time for the user etc. An optimised plan for exercise and diet plan for breakfast, lunch and dinner using Greedy Fractional Knapsack would be obtained for a week, thus making the user achieve their desired weight.

## INTRODUCTION

The key to successful weight loss is developing healthy diet and exercise habits. You may not like those words - diet and exercise. But don't get hung up on them. Diet just means eating healthy, lower calorie meals. Exercise means being more physically active.

The food we eat plays a vital role in how we look and feel. When it comes to eating foods to fuel your exercise performance, it’s not as simple as choosing vegetables over doughnuts. You need to eat the right types of food at the right times of the day. Although people appropriately focus on diet when they're trying to lose weight, being active also is an essential component of a weight-loss program. When you're active, your body uses energy (calories) to move, helping to burn the calories you take in with food you eat.

Cleaning the house, making the bed, shopping, mowing and gardening are all forms of physical activity. Exercise, on the other hand, is a structured and repetitive form of physical activity that you do on a regular basis.

This diet promotes weight loss in several ways, including:

- encourages people to eat more quantity of food, which are healthful, low-calorie foods.
- does not allow added sugars or processed foods
- lowers people’s daily calorie intake.

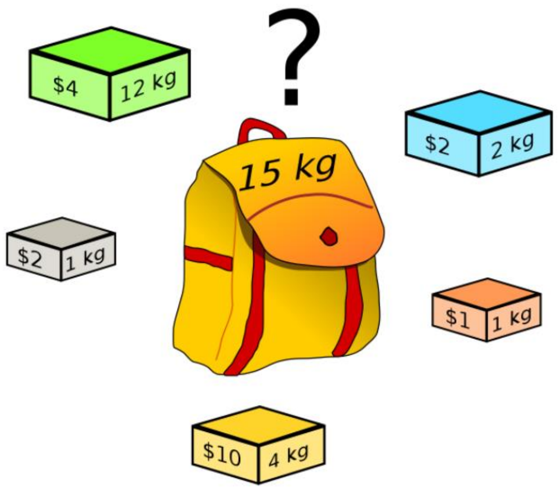
Obesity is preventable and is the result of a complex, multifactorial integration of environmental and social factors that influence our dietary and physical activity patterns. A healthy balanced diet accompanied by regular exercise is essential in maintaining physical and mental health and well-being. Not only are these effective in preventing excess weight gain or in maintaining weight loss, but healthier lifestyles are also associated with improved sleep and mood. Physical activity particularly improves brain-related function and outcomes.

## KNAPSACK PROBLEM

Given a set of items, each with a weight and a value, determine a subset of items to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible.
The knapsack problem is a combinatorial optimization problem. It appears as a subproblem in many, more complex mathematical models of real-world problems. One general approach to difficult problems is to identify the most restrictive constraint, ignore the others, solve a knapsack problem, and somehow adjust the solution to satisfy the ignored constraints.

### Problem Scenario

A thief is robbing a store and can carry a maximal weight of $W$ into his knapsack. There are n items available in the store and weight of $i^{th}$ item is $w_i$ and its profit is $p_i$. What items should the thief take?



In this context, the items should be selected in such a way that the thief will carry those items for which he will gain maximum profit. Hence, the objective of the thief is to maximize the profit.

Based on the nature of the items, Knapsack problems are categorized as

- Fractional Knapsack
- Knapsack

For our problem statement in order to plan fitness and diet for a person, we will choose fractional knapsack method cause fractional knapsack method can help in splitting the food can be split into how much ever quantity that person can eat.

## Fractional Knapsack

In this case, items can be broken into smaller pieces, hence the thief can select fractions of items.

According to the problem statement,

- There are **n** items in the store
- Weight of **i**<b>th</b> item wi>0
- Profit for **i**th item pi>0
- Capacity of the Knapsack is **W**

In this version of Knapsack problem, items can be broken into smaller pieces. So, the thief may take only a fraction $x_i$ of **i**th item.

$$0 \leqslant x_i \leqslant 1$$

The **i**th item contributes the weight xi.wi to the total weight in the knapsack and profit xi.pi to the total profit.

Hence, the objective of this algorithm is to

$$maximize \sum_{n=1}^{n} (x_i . pi)$$

subject to the constraint,

$$\sum_{n=1}^{n}(x_i . wi) \leqslant W$$

It is clear that an optimal solution must fill the knapsack exactly, otherwise we could add a fraction of one of the remaining items and increase the overall profit.

Thus, an optimal solution can be obtained by

$$\sum_{n=1}^{n}(x_i . wi) = W$$

In this context, first we need to sort those items according to the value of ***pi/wi***, so that ***pi+1/wi+1 ≤ piwi*** . Here, ***x*** is an array to store the fraction of items.

**Approaches to solve Fractional Knapsack Algorithm**

Fractional Knapsack problem can be solved by using the following two methods:

1. Dynamic Programming method
2. Greedy method

**DYNAMIC APPROACH**:

Dynamic algorithm is an algorithm design optimization method, which can be used when the problem breaks down into simpler sub-problems. Wherever there is a recursive solution that has repeated calls for the same inputs, it can be optimized by using dynamic programming. The idea is to simply store the results of sub-problems so that they do not have to be re-computed when needed later. This algorithm thus utilizes the fact that the optimal solution to the overall problem depends upon the optimal solution to its subproblems. This simple optimization reduces time complexities from exponential to polynomial.

**GREEDY APPROACH**:

Greedy is an algorithmic optimization method. Solving a problem using a greedy approach means solving the problem step-by-step. On each step, the algorithm makes a choice, based on some heuristic, that achieves the most obvious and beneficial profit. The algorithm hopes to achieve an optimal solution, even though it’s not always achievable. The greedy approach is suitable for problems where local optimality leads to an optimal global solution

Both dynamic and greedy approaches solve our fractional knapsack problem and give us an optimal solution. But the only difference is the time complexities. Greedy algorithms are generally faster. So, we go with greedy for our fractional knapsack problem.

**Steps to solve fractional knapsack problem using Greedy:**

- Find the value/weight ***(pi/wi)*** ratio for each given item.
- Sort the items based on the ratio of value/weight.
- Then start picking up the item with the highest ratio until that item is completely taken up. After that we will move to the next highest.

**Pseudo code:**

```
Fractional Knapsack (Array W, Array V, int M)
1. for i <- 1 to size (V)
2.      calculate cost[i] <- V[i] / W[i]
3. Sort-Descending (cost)
4. i ← 1
5. while (i <= size(V))
6.      if  W[i] <= M
7.              M ← M – W[i]
8.              total ← total + V[i];
9.      if  W[i] > M
10.             i ← i+1
```

**Analysis**

If the provided items are already sorted into a decreasing order of ***pi/wi*** in greedy, then the while loop takes a time in ***O(n)***; Therefore, the total time including the sort is in ***O(n logn)***.

## SYSTEM MODEL  &  IMPLEMENTATION

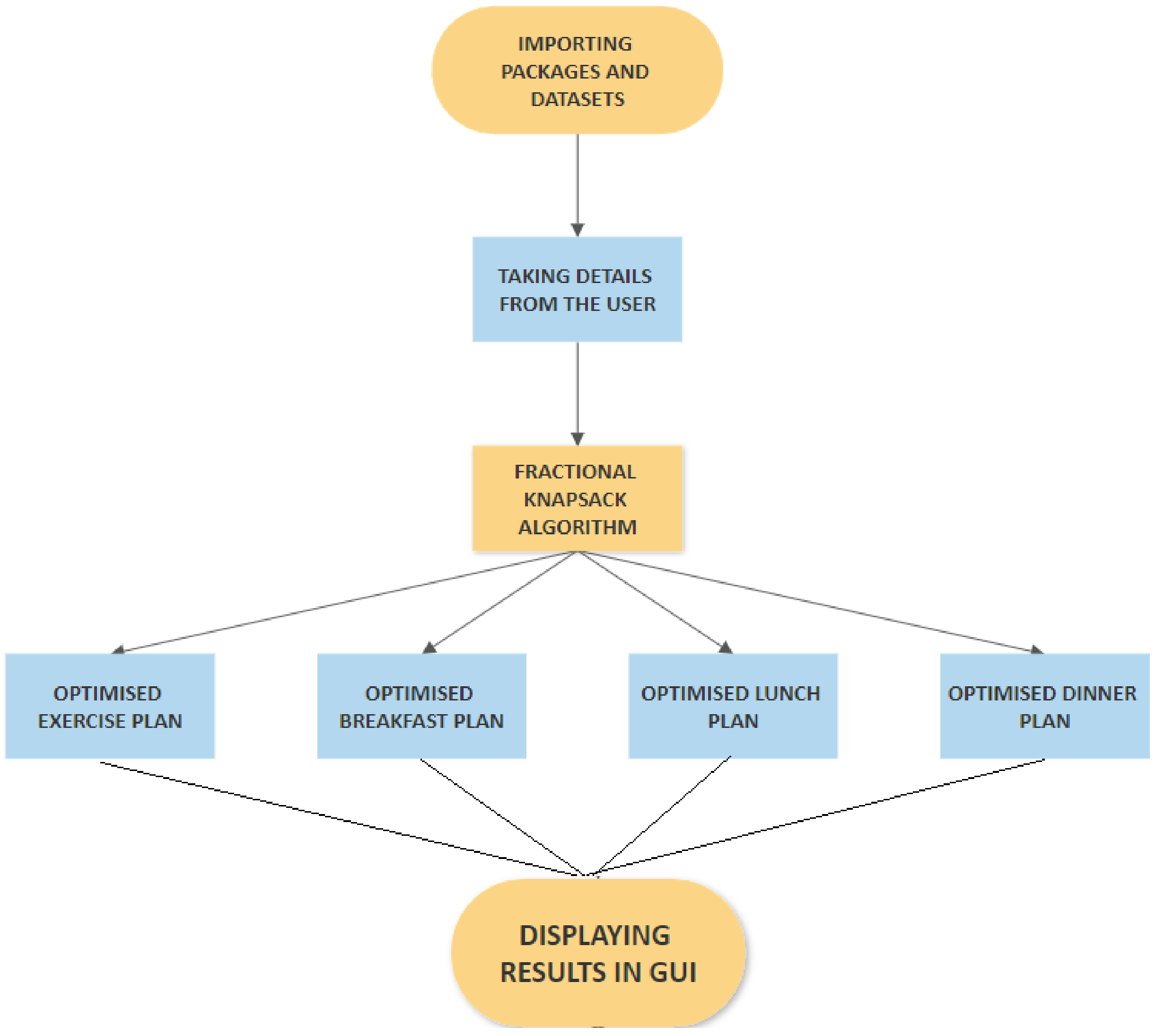
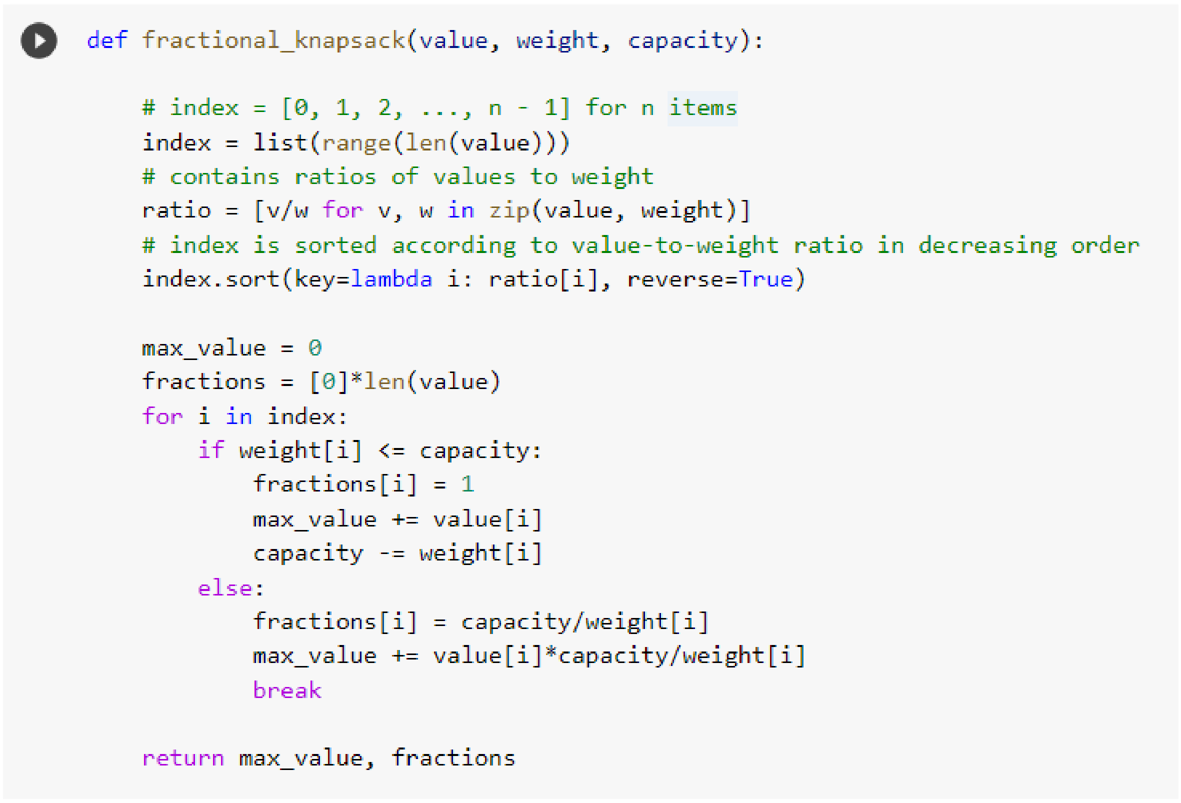

IMPORTING PACKAGES AND DATASETS
TAKING DETAILS FROM THE USER
FRACTIONAL KNAPSACK ALGORITHM
OPTIMISED EXERCISE PLAN
OPTIMISED BREAKFAST PLAN
OPTIMISED LUNCH PLAN
OPTIMISED DINNER PLAN
DISPLAYING RESULTS IN GUI

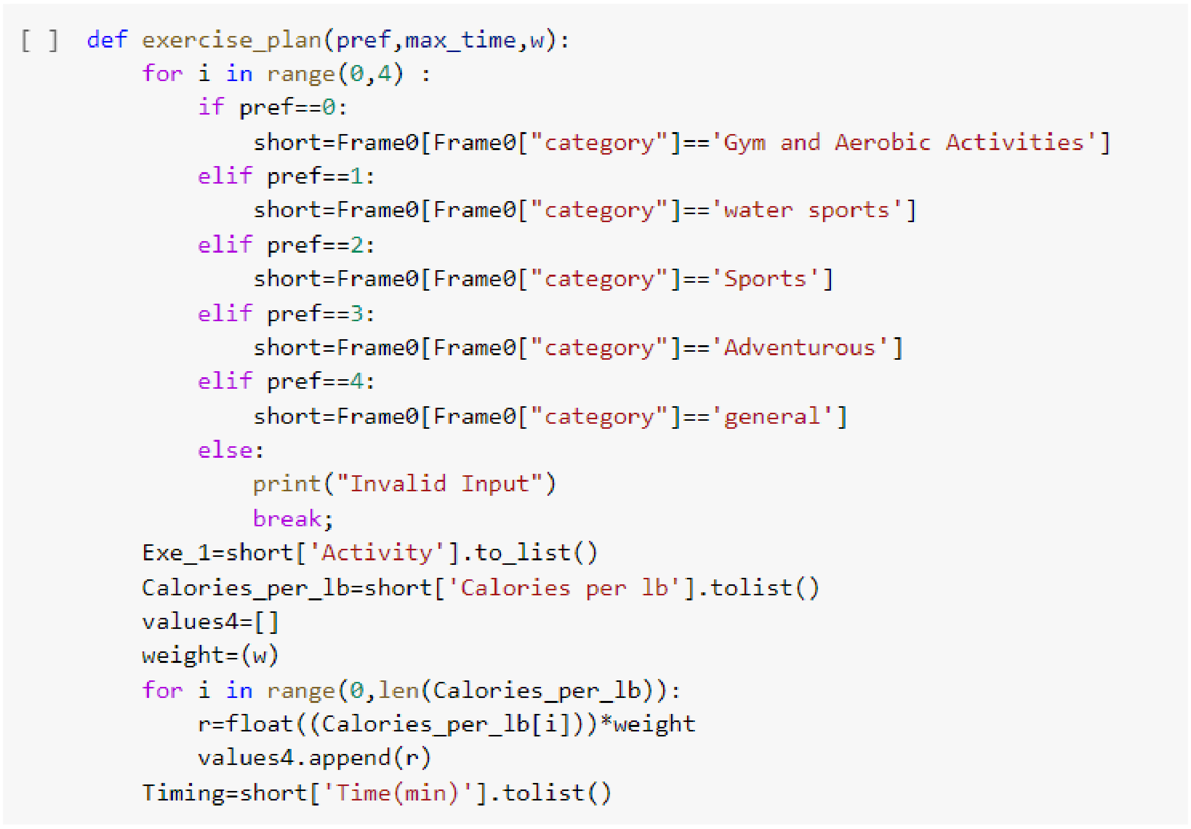
**Fig1**

We import the required packages and the datasets activity.csv, breakfast.csv, unch.csv, dinner.csv. Personal details and details of choices of activities and exercises from the user are taken using a user friendly – Graphical User Interface(GUI) called Tkinter. The given data when passes through fractional knapsack algorithm gives us an optimised exercise plan, optimised breakfast plan, optimised lunch plan and optimised dinner plan. All these obtained  plans will be displayed in the gui window for the user to follow for a week.

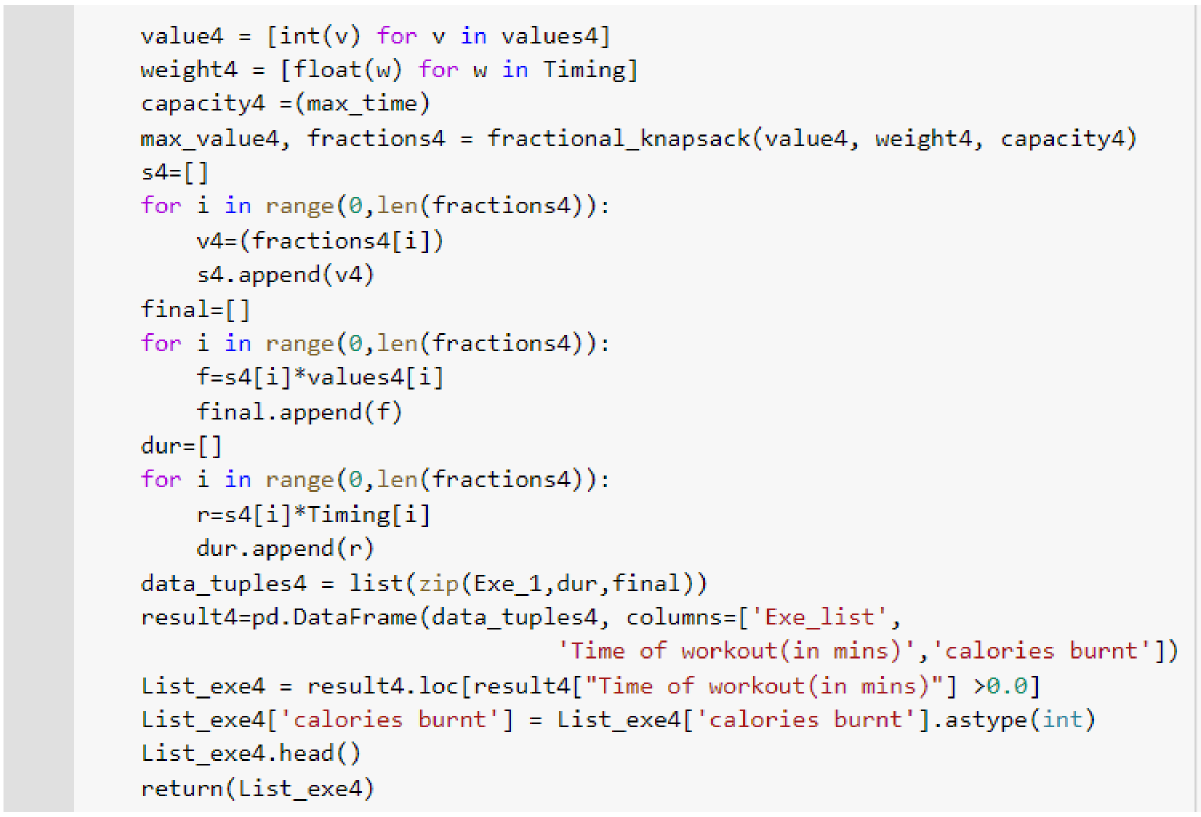
# SAMPLE CODE



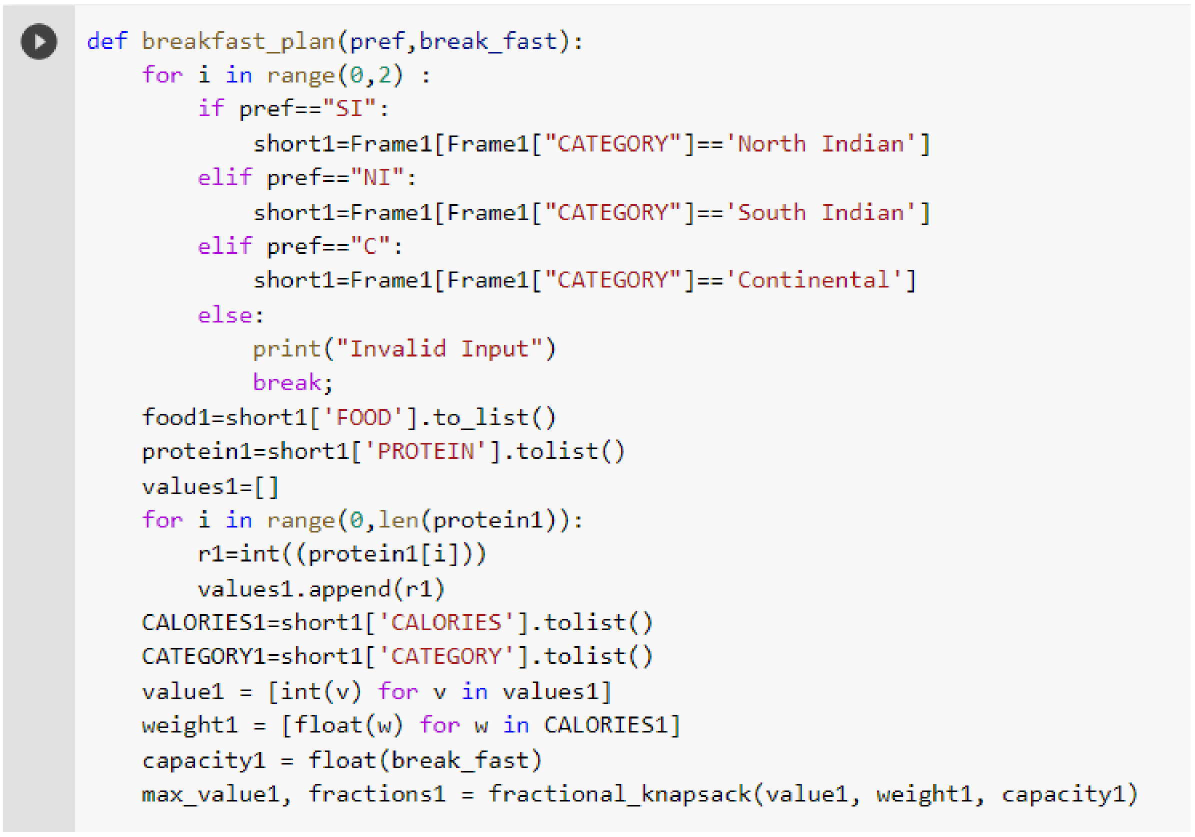
```python
def fractional_knapsack(value, weight, capacity):

    # index = [0, 1, 2, ..., n - 1] for n items
    index = list(range(len(value)))
    # contains ratios of values to weight
    ratio = [v/w for v, w in zip(value, weight)]
    # index is sorted according to value-to-weight ratio in decreasing order
    index.sort(key=lambda i: ratio[i], reverse=True)

    max_value = 0
    fractions = [0]*len(value)
    for i in index:
        if weight[i] <= capacity:
            fractions[i] = 1
            max_value += value[i]
            capacity -= weight[i]
        else:
            fractions[i] = capacity/weight[i]
            max_value += value[i]*capacity/weight[i]
            break

    return max_value, fractions
```



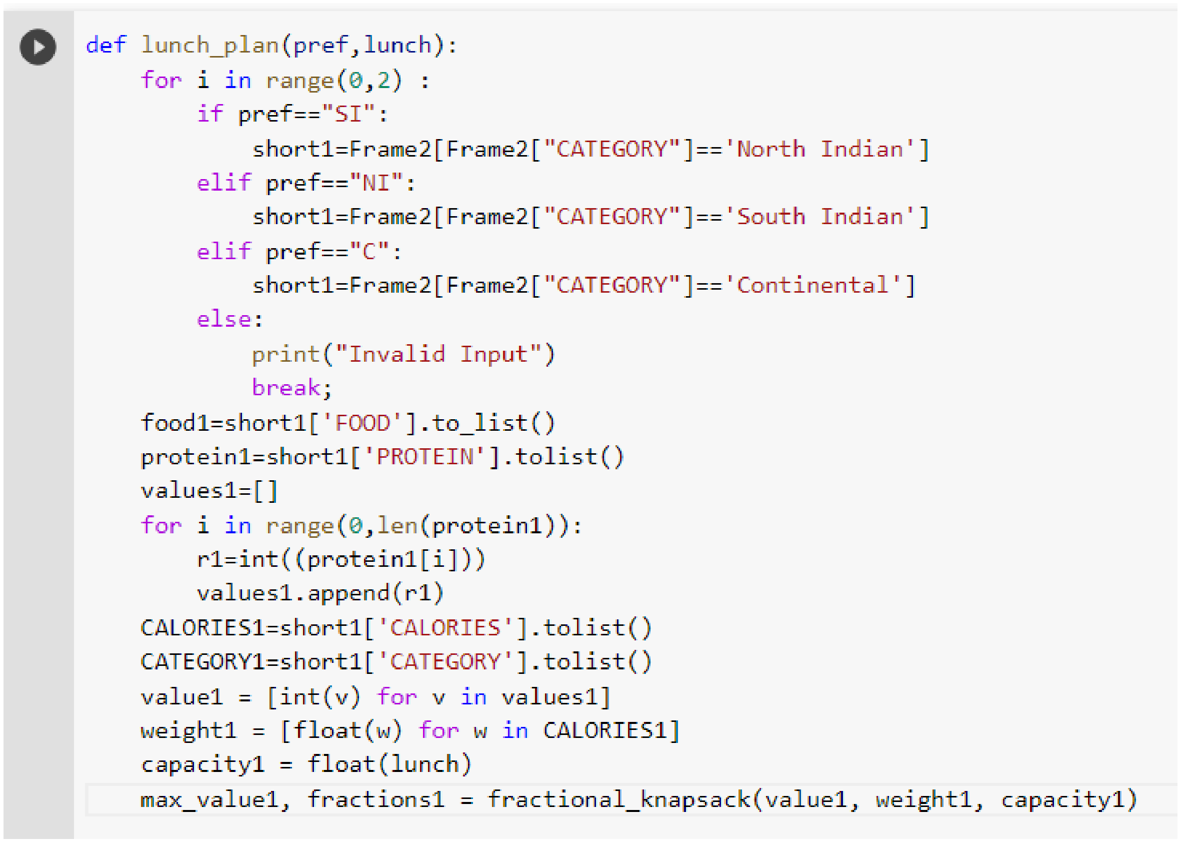
```python
[ ] def exercise_plan(pref,max_time,w):
        for i in range(0,4) :
            if pref==0:
                short=Frame0[Frame0["category"]=='Gym and Aerobic Activities']
            elif pref==1:
                short=Frame0[Frame0["category"]=='water sports']
            elif pref==2:
                short=Frame0[Frame0["category"]=='Sports']
            elif pref==3:
                short=Frame0[Frame0["category"]=='Adventurous']
            elif pref==4:
                short=Frame0[Frame0["category"]=='general']
            else:
                print("Invalid Input")
                break;
        Exe_1=short['Activity'].to_list()
        Calories_per_lb=short['Calories per lb'].tolist()
        values4=[]
        weight=(w)
        for i in range(0,len(Calories_per_lb)):
            r=float((Calories_per_lb[i]))*weight
            values4.append(r)
        Timing=short['Time(min)'].tolist()
```



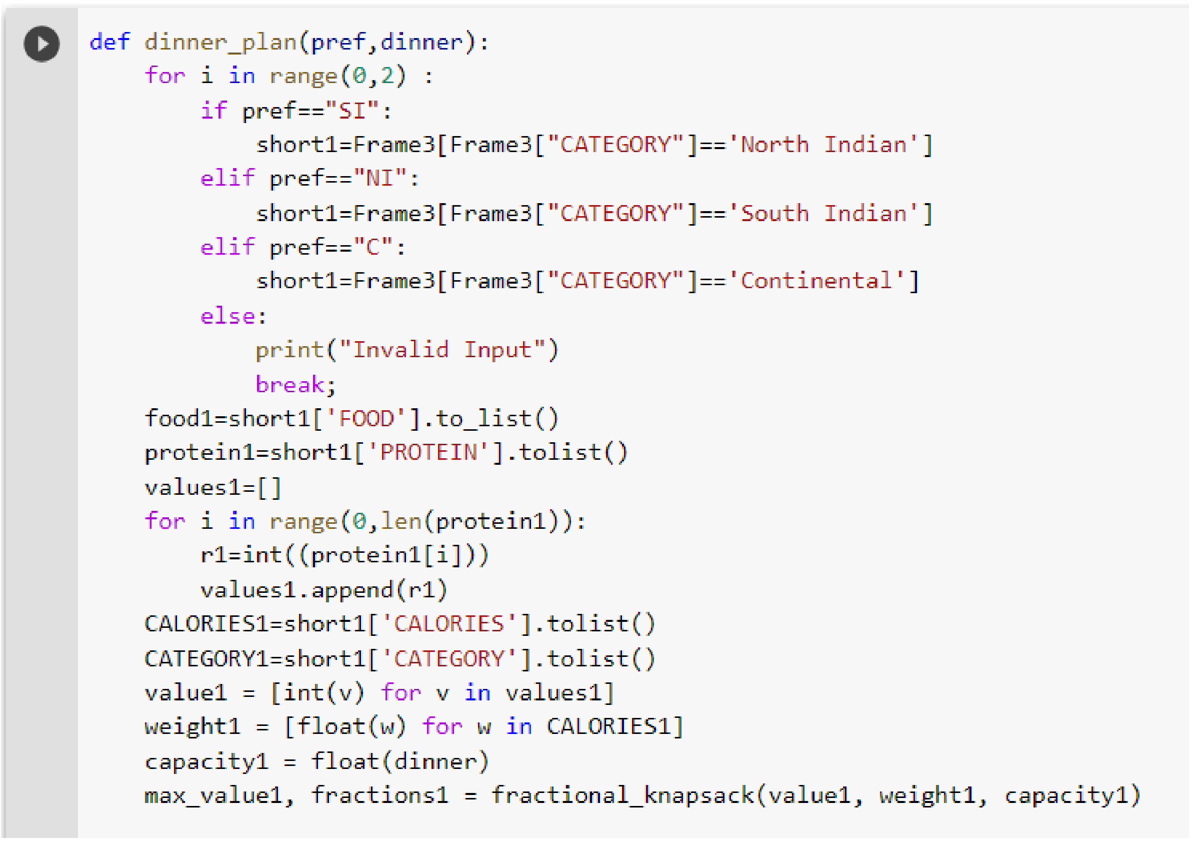
```python
    value4 = [int(v) for v in values4]
    weight4 = [float(w) for w in Timing]
    capacity4 =(max_time)
    max_value4, fractions4 = fractional_knapsack(value4, weight4, capacity4)
    s4=[]
    for i in range(0,len(fractions4)):
        v4=(fractions4[i])
        s4.append(v4)
    final=[]
    for i in range(0,len(fractions4)):
        f=s4[i]*values4[i]
        final.append(f)
    dur=[]
    for i in range(0,len(fractions4)):
        r=s4[i]*Timing[i]
        dur.append(r)
    data_tuples4 = list(zip(Exe_1,dur,final))
    result4=pd.DataFrame(data_tuples4, columns=['Exe_list',
                                  'Time of workout(in mins)','calories burnt'])
    List_exe4 = result4.loc[result4["Time of workout(in mins)"] >0.0]
    List_exe4['calories burnt'] = List_exe4['calories burnt'].astype(int)
    List_exe4.head()
    return(List_exe4)
```



```python
def breakfast_plan(pref,break_fast):
    for i in range(0,2) :
        if pref=="SI":
            short1=Frame1[Frame1["CATEGORY"]=='North Indian']
        elif pref=="NI":
            short1=Frame1[Frame1["CATEGORY"]=='South Indian']
        elif pref=="C":
            short1=Frame1[Frame1["CATEGORY"]=='Continental']
        else:
            print("Invalid Input")
            break;
    food1=short1['FOOD'].to_list()
    protein1=short1['PROTEIN'].tolist()
    values1=[]
    for i in range(0,len(protein1)):
        r1=int((protein1[i]))
        values1.append(r1)
    CALORIES1=short1['CALORIES'].tolist()
    CATEGORY1=short1['CATEGORY'].tolist()
    value1 = [int(v) for v in values1]
    weight1 = [float(w) for w in CALORIES1]
    capacity1 = float(break_fast)
    max_value1, fractions1 = fractional_knapsack(value1, weight1, capacity1)
```



```python
def lunch_plan(pref,lunch):
    for i in range(0,2) :
        if pref=="SI":
            short1=Frame2[Frame2["CATEGORY"]=='North Indian']
        elif pref=="NI":
            short1=Frame2[Frame2["CATEGORY"]=='South Indian']
        elif pref=="C":
            short1=Frame2[Frame2["CATEGORY"]=='Continental']
        else:
            print("Invalid Input")
            break;
    food1=short1['FOOD'].to_list()
    protein1=short1['PROTEIN'].tolist()
    values1=[]
    for i in range(0,len(protein1)):
        r1=int((protein1[i]))
        values1.append(r1)
    CALORIES1=short1['CALORIES'].tolist()
    CATEGORY1=short1['CATEGORY'].tolist()
    value1 = [int(v) for v in values1]
    weight1 = [float(w) for w in CALORIES1]
    capacity1 = float(lunch)
    max_value1, fractions1 = fractional_knapsack(value1, weight1, capacity1)
```

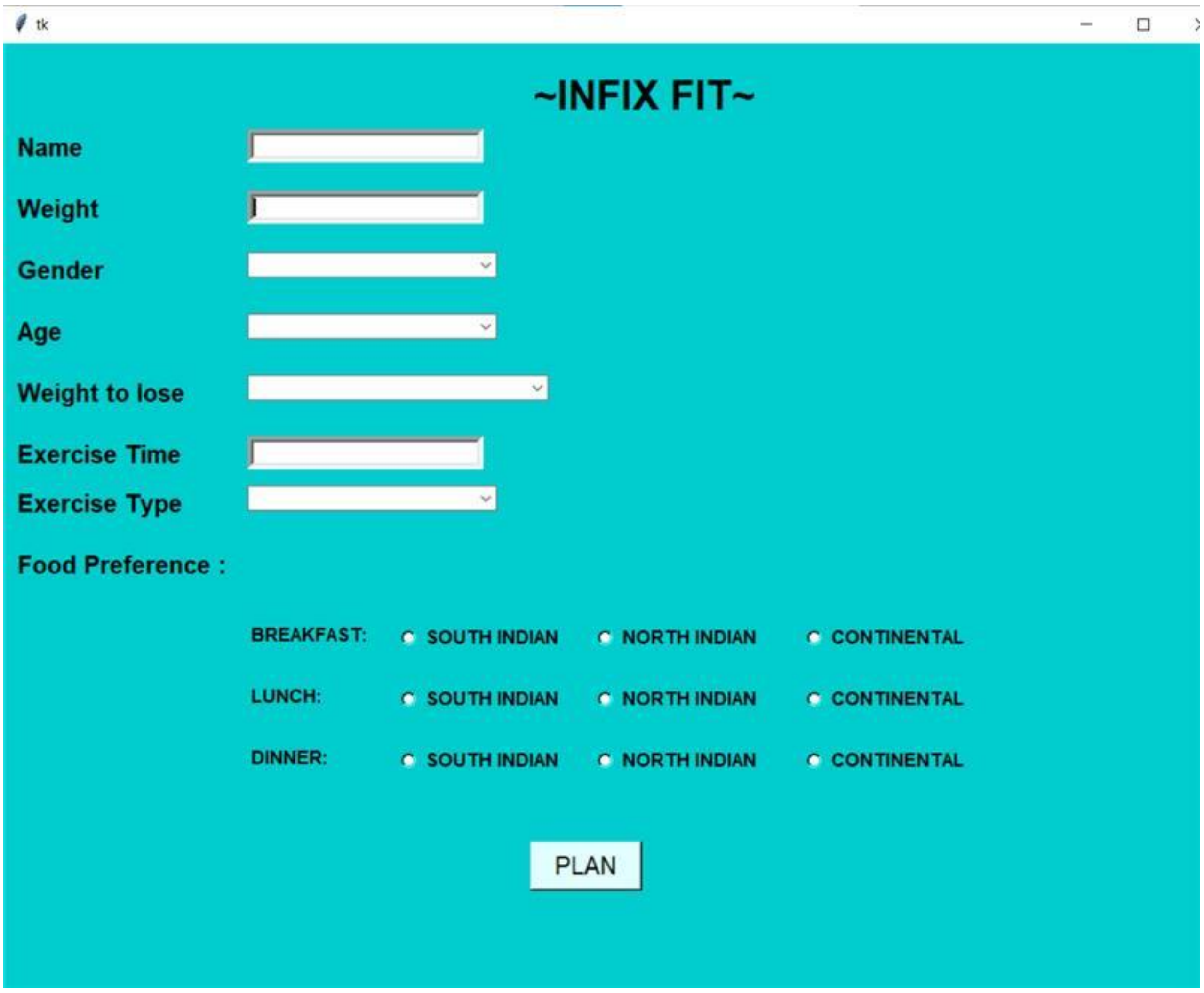


```python
def dinner_plan(pref,dinner):
    for i in range(0,2) :
        if pref=="SI":
            short1=Frame3[Frame3["CATEGORY"]=='North Indian']
        elif pref=="NI":
            short1=Frame3[Frame3["CATEGORY"]=='South Indian']
        elif pref=="C":
            short1=Frame3[Frame3["CATEGORY"]=='Continental']
        else:
            print("Invalid Input")
            break;
    food1=short1['FOOD'].to_list()
    protein1=short1['PROTEIN'].tolist()
    values1=[]
    for i in range(0,len(protein1)):
        r1=int((protein1[i]))
        values1.append(r1)
    CALORIES1=short1['CALORIES'].tolist()
    CATEGORY1=short1['CATEGORY'].tolist()
    value1 = [int(v) for v in values1]
    weight1 = [float(w) for w in CALORIES1]
    capacity1 = float(dinner)
    max_value1, fractions1 = fractional_knapsack(value1, weight1, capacity1)
    s1=[]
    for i in range(0,len(fractions1)):
        v1=(fractions1[i])
        s1.append(v1)
    final1=[]
    for i in range(0,len(fractions1)):
        f1=s1[i]*values1[i]
        final1.append(f1)
    dur1=[]
    for i in range(0,len(fractions1)):
        r1=s1[i]*CALORIES1[i]
        dur1.append(r1)
    data_tuples1 = list(zip(food1,s1,dur1,final1,CATEGORY1))
    result1=pd.DataFrame(data_tuples1, columns=['FOOD',
                                'QUANTITY','CALORIES','PROTEIN','CATEGORY'])
    list_food1 = result1.loc[result1["QUANTITY"] >0.0]
    return(list_food1)
```

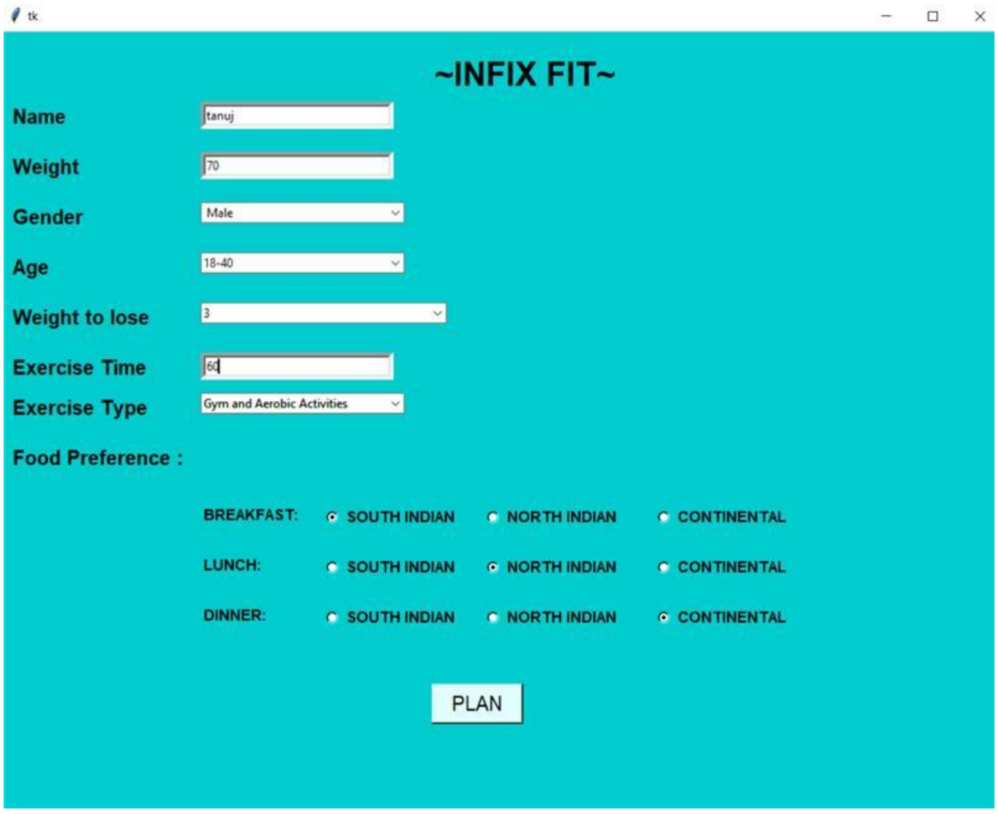
## SAMPLE OUTPUT

### GUI WINDOW

**Tkinter** is the most commonly used library for developing GUI (Graphical User Interface) in Python. It is a standard Python interface to the Tk GUI toolkit shipped with Python. As Tk and Tkinter are available on most of the Unix platforms as well as on the Windows system, developing GUI applications with Tkinter becomes the fastest and easiest. We have used buttons, text boxes, labels, etc to create the below GUI.



## CASE 1:



tk

~INFIX FIT~

Name: tanuj

Weight: 70

Gender: Male

Age: 18-40

Weight to lose: 3

Exercise Time: 60

Exercise Type: Gym and Aerobic Activities

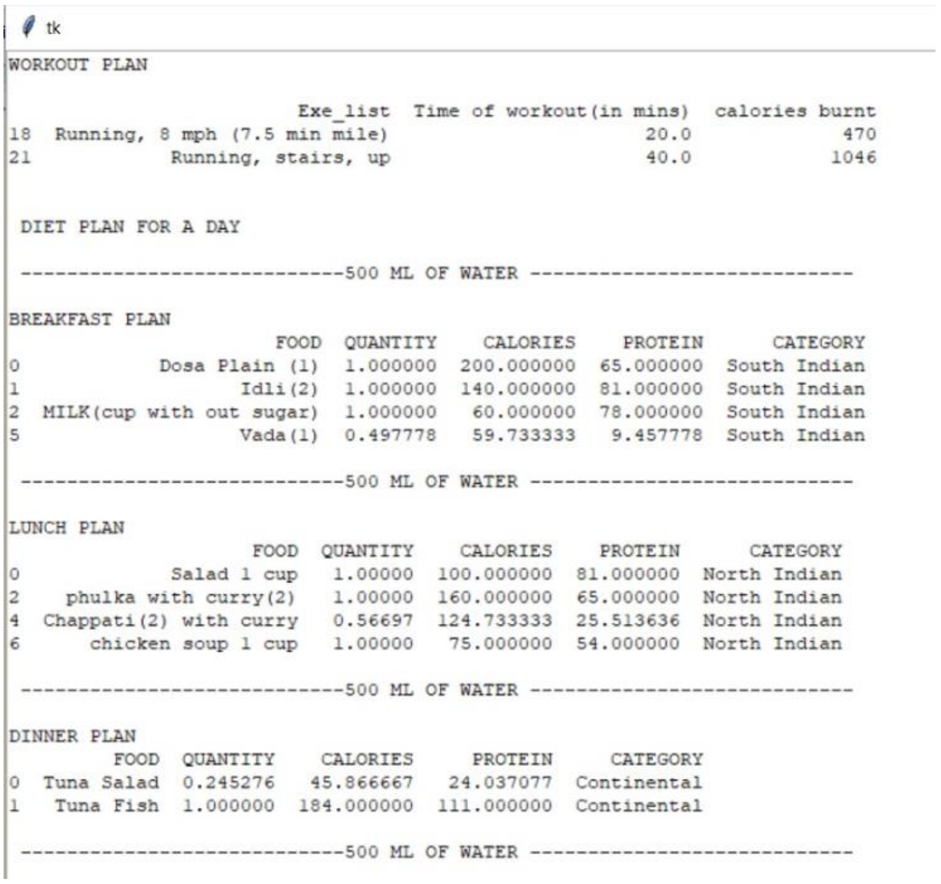
Food Preference :

BREAKFAST: [✓ SOUTH INDIAN] NORTH INDIAN CONTINENTAL

LUNCH: SOUTH INDIAN [✓ NORTH INDIAN] CONTINENTAL

DINNER: SOUTH INDIAN NORTH INDIAN [✓ CONTINENTAL]

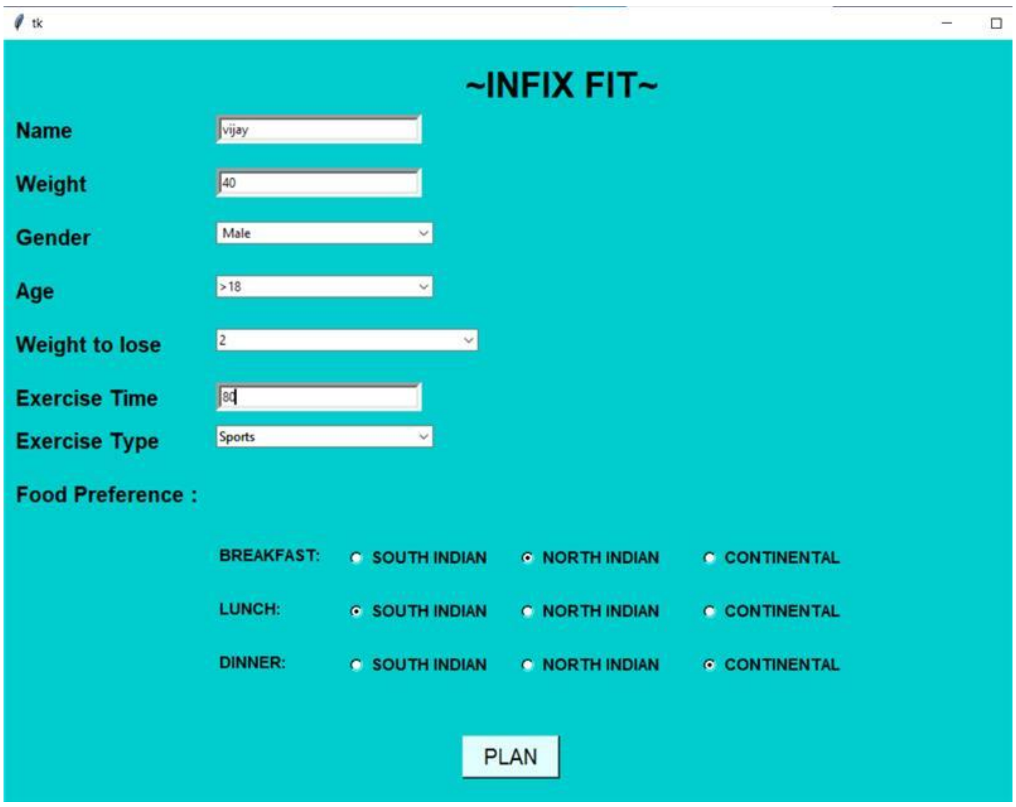
PLAN



tk

```
WORKOUT PLAN

                        Exe_list  Time of workout(in mins)  calories burnt
18  Running, 8 mph (7.5 min mile)                     20.0             470
21             Running, stairs, up                    40.0            1046


 DIET PLAN FOR A DAY

 ----------------------------500 ML OF WATER ---------------------------

BREAKFAST PLAN
                       FOOD  QUANTITY    CALORIES    PROTEIN     CATEGORY
0            Dosa Plain (1)  1.000000  200.000000  65.000000  South Indian
1                   Idli(2)  1.000000  140.000000  81.000000  South Indian
2  MILK(cup with out sugar)  1.000000   60.000000  78.000000  South Indian
5                   Vada(1)  0.497778   59.733333   9.457778  South Indian

 ----------------------------500 ML OF WATER ---------------------------

LUNCH PLAN
                     FOOD  QUANTITY    CALORIES    PROTEIN     CATEGORY
0            Salad 1 cup   1.00000  100.000000  81.000000  North Indian
2     phulka with curry(2)  1.00000  160.000000  65.000000  North Indian
4  Chappati(2) with curry   0.56697  124.733333  25.513636  North Indian
6         chicken soup 1 cup  1.00000   75.000000  54.000000  North Indian

 ----------------------------500 ML OF WATER ---------------------------

DINNER PLAN
         FOOD  QUANTITY    CALORIES     PROTEIN    CATEGORY
0  Tuna Salad  0.245276   45.866667   24.037077  Continental
1   Tuna Fish  1.000000  184.000000  111.000000  Continental

 ----------------------------500 ML OF WATER ---------------------------
```

## CASE 2:


tk

~INFIX FIT~

Name: vijay

Weight: 40

Gender: Male

Age: >18

Weight to lose: 2

Exercise Time: 80

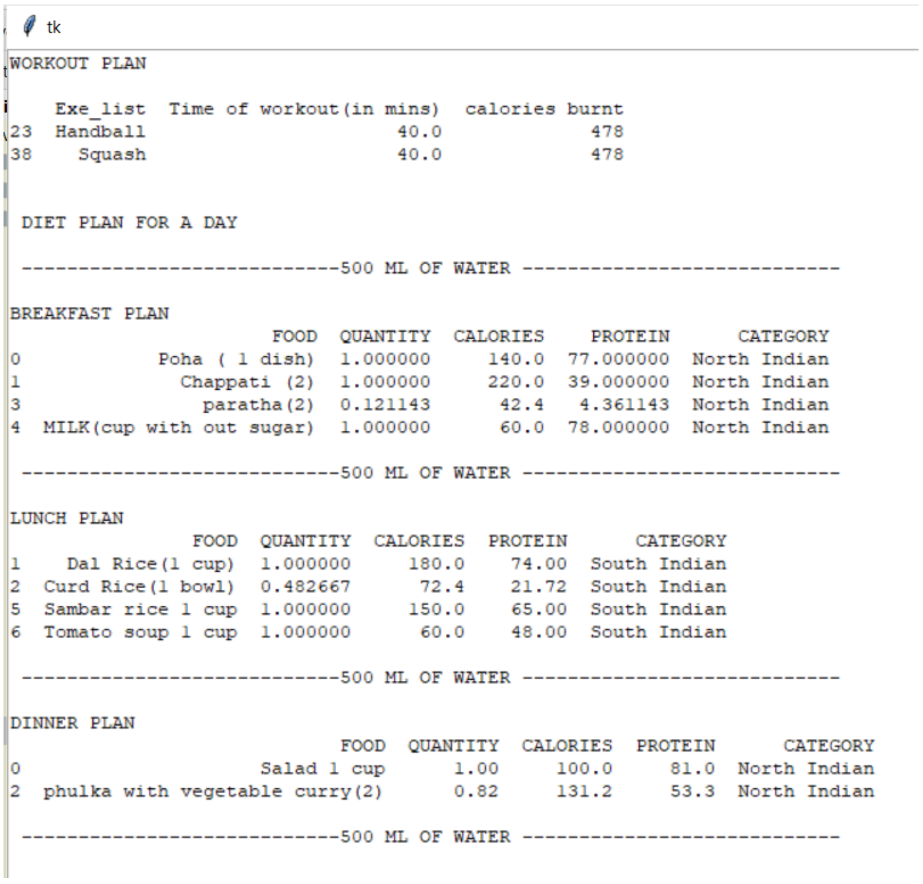
Exercise Type: Sports

Food Preference :

BREAKFAST: SOUTH INDIAN [✓ NORTH INDIAN] CONTINENTAL

LUNCH: [✓ SOUTH INDIAN] NORTH INDIAN CONTINENTAL

DINNER: SOUTH INDIAN NORTH INDIAN [✓ CONTINENTAL]


tk

```
WORKOUT PLAN

    Exe_list  Time of workout(in mins)  calories burnt
23  Handball                      40.0             478
38    Squash                      40.0             478


 DIET PLAN FOR A DAY

 ----------------------------500 ML OF WATER ----------------------------

BREAKFAST PLAN
                       FOOD  QUANTITY  CALORIES    PROTEIN      CATEGORY
0             Poha ( 1 dish)  1.000000     140.0  77.000000  North Indian
1               Chappati (2)  1.000000     220.0  39.000000  North Indian
3                 paratha(2)  0.121143      42.4   4.361143  North Indian
4   MILK(cup with out sugar)  1.000000      60.0  78.000000  North Indian

 ----------------------------500 ML OF WATER ----------------------------

LUNCH PLAN
                FOOD  QUANTITY  CALORIES  PROTEIN      CATEGORY
1     Dal Rice(1 cup)  1.000000     180.0    74.00  South Indian
2   Curd Rice(1 bowl)  0.482667      72.4    21.72  South Indian
5   Sambar rice 1 cup  1.000000     150.0    65.00  South Indian
6   Tomato soup 1 cup  1.000000      60.0    48.00  South Indian

 ----------------------------500 ML OF WATER ----------------------------

DINNER PLAN
                                FOOD  QUANTITY  CALORIES  PROTEIN      CATEGORY
0                       Salad 1 cup      1.00     100.0     81.0  North Indian
2   phulka with vegetable curry(2)      0.82     131.2     53.3  North Indian

 ----------------------------500 ML OF WATER ----------------------------
```

## **CONCLUSION**

In this project, we accomplished a proper weight loss scheme which can be followed by any user desiring to reduce their weight. A healthy balanced diet plan accompanied by regular exercise plan would be suggested to the user using Fractional Knapsack algorithm with a user friendly - graphical user interface. Following this plan for a week, will make the user achieve their desired weight.