Bmr calculator:

**Scenario documentation:**

This program is designed to help users calculate their Basal Metabolic Rate (BMR) and Total Daily Energy Expenditure (TDEE) based on their basic information such as weight, height, age, and activity level. It also helps users set their daily calorie goals based on their weight goals, whether they want to lose weight, maintain weight, or gain weight.

The program has 5 classes: Bmrcalulator,UserInfo, Tdee, CalorieGoal, and CalorieTracker.

The BmrCalculator class calculates a user's BMR based on their basic information, using the Harris-Benedict equation. It has a method to calculate the BMR and a method to retrieve the calculated BMR.

The UserInfo class is responsible for getting the user's basic information such as name, gender, weight, height, and age. The information is obtained through user input using a Scanner object.

The Tdee class uses the user's BMR to calculate their TDEE based on their activity level. The user is prompted to select their activity level from a list of options ranging from sedentary to super active. The Tdee class then calculates the TDEE based on the selected activity level using a switch statement.

The CalorieGoal class helps users set their daily calorie goals based on their weight goals. The user is prompted to select their weight goal from a list of options, and the program calculates the total calories the user should consume daily based on their TDEE and weight goals.

The CalorieTracker class is responsible for tracking the user's daily calorie intake. The user is prompted to enter the number of calories they consume throughout the day, and the program keeps track of the user's total calorie intake. If the user exceeds their daily calorie goal, the program alerts them and displays the number of extra calories consumed. If the user has not yet reached their daily calorie goal, the program displays the number of calories remaining to reach their goal.

Overall, this program is designed to help users understand their daily calorie needs and help them reach their weight goals in a healthy and sustainable way.

**pseudo code algorithms:**

**user info:**

Create a class named UserInfo.

Declare private instance variables for name, gender, weight, height, and age.

Create a method named getBasicInfo that accepts a Scanner object as parameter.

Inside the getBasicInfo method, prompt the user to enter their name, gender, weight, height, and age.

Use the Scanner object to read the user input for each of these attributes.

Assign the input values to their corresponding instance variables.

Create getter methods for each of the instance variables.

End of class.

**bmr calculator:**

1.Start by creating a package named "basalmetabolicrate"

2.Create a public class named "BmrCalculator"

Declare two private variables, "bmr" of type double and "userInfo" of type "UserInfo"

Create a constructor for the "BmrCalculator" class that takes a "UserInfo" object as a parameter and assigns it to the "userInfo" variable

Create a public method named "calculateBMR" that has no parameters and returns void

Within the "calculateBMR" method, extract the user's weight, height, age, and gender from the "userInfo" object using getter methods and assign them to local variables.

Use an if statement to determine if the user's gender is male or female. If the gender is male, use the Harris-Benedict equation to calculate the user's BMR and assign it to the "bmr" variable. If the gender is female, use a different Harris-Benedict equation to calculate the user's BMR and assign it to the "bmr" variable.

Create a public method named "getBMR" that returns the value of the "bmr" variable.

End the class.

**Tdee class:**

1. 1.Create a class named Tdee that implements the TdeeInterface.
2. 2.Declare two private instance variables: tdee and bmr of type double.
3. 3.Create a constructor that takes a double bmr as a parameter and sets it to the instance variable.
4. 4.Implement the calculateTDEE method that takes a Scanner input as a parameter.
5. 5.In the calculateTDEE method, print out the activity level options to the console.
6. 6.Use the input.nextInt() method to read the activity level selected by the user and store it in a variable named activityLevel.
7. 7.Use a switch statement to calculate the tdee value based on the activity level selected by the user.
8. Store the calculated tdee value in the instance variable tdee.
9. Implement the getTDEE method that returns the value of tdee.

**Calorie goal:**

1. Define a public class named "CalorieGoal".
2. Declare two private instance variables: "totalCalories" and "tdee" of type double.
3. Create a constructor that takes in a double parameter "tdee" and assigns its value to the "tdee" instance variable.
4. Create an enum named "WeightGoal" with three values: "LOSE\_WEIGHT", "MAINTAIN\_WEIGHT", and "GAIN\_WEIGHT".
5. Create a public method named "calculateCalorieGoal" that takes in a Scanner object as a parameter.
6. Display a message asking the user to select their weight goal and provide three options to choose from.
7. Get the user's input for the weight goal and store it in an integer variable "goalOption".
8. Use a switch statement to assign the appropriate "WeightGoal" enum value based on the user's input. If the user enters an invalid option, display an error message and recursively call the "calculateCalorieGoal" method.
9. Use another switch statement to calculate the total calories based on the selected weight goal. If the user chooses to lose weight, ask them how many pounds they would like to lose per week and use the provided formula to calculate the total calories. If the user chooses to gain weight, ask them how many pounds they would like to gain per week and use the provided formula to calculate the total calories. If the user chooses to maintain weight, set the total calories equal to the TDEE.
10. Create a public method named "getTotalCalories" that returns the value of the "totalCalories" instance variable.

**Calorie tracker:**

1. Define a class named "CalorieTracker"
2. Declare private variables "totalCalories" and "currentCalories" of double data type.
3. Create a constructor with a parameter "totalCalories" of double data type and set the "totalCalories" variable.
4. Create a public method named "trackCalories" with a parameter "input" of Scanner data type.
5. Inside the "trackCalories" method, display a message "Please enter the number of calories you consumed today: " and read the input from the user.
6. Add the consumed calories to the "currentCalories" variable.
7. Check if "currentCalories" is greater than "totalCalories".
8. If it is greater, calculate the number of calories exceeded and display a message "You have exceeded your daily calorie goal by (number of calories exceeded) calories."
9. If it is not greater, calculate the number of calories remaining to reach the goal and display a message "You have (number of calories remaining) calories remaining to reach your goal."
10. Display a message "Would you like to track more calories? (Y/N)" and read the input from the user.
11. If the input is "Y", call the "trackCalories" method recursively.
12. If the input is "N", display a message "You have consumed (currentCalories) calories today."
13. End the "trackCalories" method.

**Save data class:**

Import necessary libraries

Create a class named "SaveData"

Define a method called "Data" that takes in the following parameters:

String name

String gender

double weight

double height

double bmr

double tdee

1. Try to create a FileWriter object with "bmr.txt" as the file name and "true" to indicate that data should be appended to the file.
2. Create a BufferedWriter object with the FileWriter object as the parameter.
3. Get the current date and time using LocalDateTime.now() method.
4. Format the date and time as "yyyy-MM-dd HH:mm:ss" using DateTimeFormatter class.
5. Write the data to the file in the following format:
6. "Name:" + name + ",Gender:" + gender + ",weight:" + weight + ",height:" + height + ",bmr:" + bmr + ",tdee:" + tdee + "," + dateTime
7. Add a new line to the file using the bufferedWriter.newLine() method.
8. Close the bufferedWriter using the bufferedWriter.close() method.
9. Catch IOException and print the stack trace if an error occurs.

**Features used:**

Enum

Collections

Interface

Exception handling (try – catch)

File writer

constructors