Semester: III

Group: I

COMPUTER PROGRAMMING 3

Author: Patrycja Cader

1. Task topic

Create the program consisting of five classes and 4 topics from the programming course. The program must be displayed as a window application consisting of Graphical User Interface. The idea of the project can be given by the student.

2. General description

The goal of the program is to allow user to lead healthy lifestyle. User is able to do calculations and track parameters in daily life. There are provided calculators for BMI, BMR and tracking systems. The interface is clear and visible for the user. The entire information concerning health is provided.

3. Project analysis

Program for tracking eating habits and workouts. Application has features: tracking results, BMI calculator and generating result. The program demanded to choose appropriate classes which consist of necessary topics from the lectures. The most interesting topics were exceptions, inheritance, templates and I/O streams. This procedure needed to revise information from the lectures. The names and tasks of the classes were chosen very simple because of bright perception of the topic.

Chosen topics:

- Exceptions - an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions. The runtime system searches the call stack for a method that contains a block of code that can handle the exception. This block of code is called an exception handler.
- I/O streams - contains all standard library functions required for writing a basic program.
- Exceptions - response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero. Exceptions provide a way to transfer control from one part of a program to another.
- Inheritance - is defined as the tendency of one class to derive properties and characteristics from other classes. It provides additional functionalities to extract features from the base class and imply it into other derived classes significantly.

Created classes:

- BMI.cpp
- Dish.cpp
- Water.cpp
- Workout.cpp
- Food.cpp

4. External specification

To compile program use commands:

make all

./health

At the beginning the user obtains a five main windows with “Calculate” and “Add” buttons.

The button must be pressed to get wanted value of calculator or to save data. Each of these options consist of inserting data. All of the windows and buttons are visible and inform the user properly how to proceed further. The user can calculate the Body Mass Index BMI by inserting his weight and height. He can also calculate Basal Metabolic Rate BMR by inserting his weight, height and age. The user can store information about amount of drank water by inserting volume of water. This gives water hydration. The user can track amount of consumed calories by inserting nutrition values as proteins, carbohydrates and fats. The user can track time spent on workout by inserting number of hours spent on training.

5. Internal specification

The program demanded to choose appropriate classes which should consist of necessary topics from the lecture.

The program contains of 5 classes: BMI, Dish, Water, Workout, Food. Each of the class is created as the heading and source class.

There is also class BMR that is inheriting from class BMI.

class	**BMI**
double weight	stores information about users weight
double height	stores information about users height
double calcBMI	method for calculating BMI index
constructor	set weigh and height to 0
destructor	deletes the object
inherited class	**BMR**
int age	stores information about users age
double calcBMR	method for calculating BMR index
constructor	set weigh, height and age to 0
destructor	deletes the object
class	**Dish**
int calories	returned information about calculated calories
int protein	stores information about dish amount of proteins
int carbs	stores information about dish amount of carbohydrates
int fat	stores information about dish amount of fats
void setDish	initialization list
int getDishCalories	returns number of calculated calories
class	**Water**
vector<int> glassesOfWaterVolume	container for storing drank water data
void addGlasses	adds new data to the container
int sumAllGlassesVolume	methode summing all inserted water data
class	**Workout**
vector<double> timeTrainedMinutes	container for storing training time data

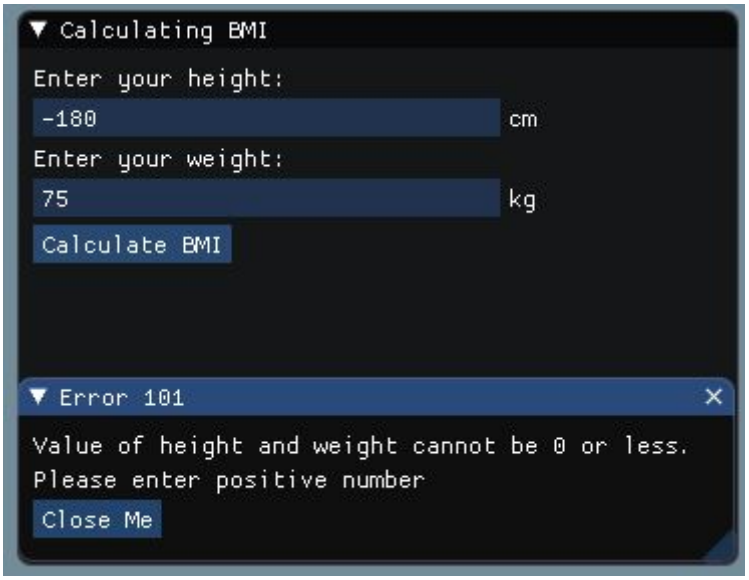
void addRecord	methode adding new data to the container
double sumAllTimeSpent	methode summing all inserted data
class	**Food**
vector Dishes	container for storing nutrition data
void addDish	method for inserting nutrition data to th container

6. *Source code*

Provided separately.

7. *Testing*

Inserting negative value of height



Inserting number 0 as weight

Inserting negative value of ml of water