

21102955 Programming Basics for Electronic Engineering Final Project

(Due date: 24.12.20 23:59)

Healthcare Management System 구현

1. 문제 정의

- 사용자가 하루의 식단과 운동 기록, 건강 상태를 기록하고 관리할 수 있는 프로그램 구현
- 식단의 후보군은 'diets.txt' 파일로부터 읽어와서 저장하며, 아래와 같은 형식으로 저장하고 있음.

(식단) (해당 식단 섭취 시 얻게 되는 kcal)

- 건강한 칼로리 섭취를 위해, 하루에 3끼를 먹음.
- 일일 권장 섭취 칼로리는 2,000kcal
- 운동의 후보군은 'exercises.txt' 파일로부터 읽어와서 저장하며, 아래와 같은 형식으로 저장하고 있음.

(운동 종류) (1분당 해당 운동 시, 소모하는 kcal)

- (사용자가 선택한 운동) * (운동 시간 [min]) = (총 소모 kcal)
- 구현하고자 하는 건강관리 프로그램은 아래와 같은 기능을 제공해야 함.
- 프로그램 실행 시, 아래와 같이 시스템이 구성되어 있어, 사용자의 needs에 따라서 선택 가능

```
=====
[Healthcare Management Systems]
1. Exercise
2. Diet
3. Show logged information
4. Exit
Select the desired number:
```

- 프로그램의 세부 사항은 아래와 같음.

0. 사용자의 needs에 따라, 운동/식사/칼로리 정보 확인 중 원하는 기능 사

용 가능

1. 출력된 식단/운동 후보 중 사용자가 원하는 옵션 선택 가능
2. 식사 옵션 선택 시, 저장된 모든 식단 후보 출력 (모든 정보 항목에 대해 출력)
3. 운동 옵션 선택 시, 저장된 모든 운동 후보 출력 (모든 정보 항목에 대해 출력) & 해당 운동을 진행할 시간 입력 가능
4. 사용자가 선택한 식사와 진행한 운동을 기준으로 현재 섭취/소모 /남은 칼로리 계산 및 해당 정보를 'health_data.txt'에 백업
5. 저장된 정보 확인 시, 현재 칼로리 세부 사항 출력과 함께 현재 기준으로 사용자에게 추천사항 출력
6. 시스템 종료 조건 도달 시, 시스템 종료 및 'health_data.txt' 파일 저장

- 프로그램 시작과 함께 'diets.txt'과 'exercises.txt'로부터 식단과 운동 관련 정보를 읽어와서 database에 저장함
 - 시작과 함께 사용자가 운동/식단 옵션 선택 시, 해당 정보들이 출력되어야 함.
- 한 번 동작을 할 때마다 'health_data.txt' 파일에 진행한 사항을 백업함
 - [Exercises] / [Diets] / [Total] 의 카테고리로 나누어 user가 선택한 항목에 맞게 백업함
 - [Example]

```
[Exercises]
running - 220 kcal
riding - 75 kcal
swimming - 240 kcal
pilates - 240 kcal
walking - 40 kcal
Total calories burned: 815 kcal

[Diets]
rice - 600 kcal
pizza - 900 kcal
tteokbokki - 615 kcal
Total calories intake: 2115 kcal

-

[Total]
Basal metabolic rate - 1300 kcal
The remaining calories - 0 kcal
```

= remain

- 'Show logged information' 옵션 선택 시, 현재까지 진행한 식사

와 운동을 기준으로 아래와 같이 출력

```
===== History of Exercise =====
Exercise: running, Calories burned : 220 kcal
Exercise: riding, Calories burned : 100 kcal
=====
===== History of Diet =====
Food: chicken, Calories intake: 925 kcal
Food: pizza, Calories intake: 900 kcal
=====
===== Total Calories =====
Basal Metabolic Rate: 1300 kcal
Total calories burned: 320 kcal
Total calories intake: 1825 kcal
The remaining calories: 205 kcal
=====

Please exercise for your health!
Your total calorie intake for today has not reached your goal, remember to eat more!
=====
```

- [Exercises] 현재까지 진행한 운동과 해당 운동을 통해 소모된 칼로리 출력
- [Diet] 현재까지 먹은 식사 종류와 해당 식사를 통해 섭취한 칼로리 출력
- [Total] 기초 대사량 (고정, 1300 kcal), 현재까지 소모된 칼로리, 현재까지 섭취된 칼로리, 남은 칼로리 (섭취 칼로리 - 기초 대사량 - 소모된 칼로리) 출력
- 사용자에게 추천사항 제공
 - ◆ 남은 칼로리 = 0 인 경우)
 - "You have consumed all your calories for today!"
 - 시스템 종료
 - ◆ 남은 칼로리 < 0 인 경우)
 - "[Warning] Too few calories!"
 - 섭취 칼로리가 일일 권장 칼로리에 도달한 경우)
 - ✓ "Your total calorie intake for today has reached your goal!"
 - 섭취 칼로리가 일일 권장 칼로리보다 적은 경우)
 - ✓ "Your total calorie intake for today has not reached your goal, remember to eat more!!"
 - 섭취 칼로리가 일일 권장 칼로리보다 많은 경우)
 - ✓ "You have eaten more calories than planned today, but you have exercised too much!"
 - ◆ 남은 칼로리 > 0 인 경우)
 - "Please exercise for your health!"
 - 섭취 칼로리가 일일 권장 칼로리에 도달한 경우)
 - ✓ "Your total calorie intake for today has reached your

goal!"

- 섭취 칼로리가 일일 권장 칼로리보다 적은 경우)
 - ✓ "Your total calorie intake for today has not reached your goal, remember to eat more!!"
-

2. 코딩 방향

- header 파일들은 수정이 필요 없음.
- 주어진 base code의 구조 하에서 구현하며, 각 소스파일은 아래와 같은 역할을 가지도록 코딩
 - main.c
 - ◆ 건강관리 프로그램의 주요 동작 흐름 실현
 - cal_healthdata.h
 - ◆ exercise 구조체, diet 구조체, healthdata 구조체 정의 (수정 필요 없음)
 - cal_healthdata.c
 - ◆ printHealthData(healthdata 구조체 포인터) - healthdata 구조체에 저장된 정보를 기반으로 사용자에게 건강 관리 히스토리 및 추천사항 제공
 - ◆ saveData(파일 포인터, healthdata 구조체 포인터) - 시스템 종료 시, healthdata 구조체에 저장된 정보를 'health_data' 파일로 백업
 - cal_exercise.c
 - ◆ loadExercises(파일 포인터) - 'exercises.txt' 파일을 읽고 저장
 - ◆ inputExercise(healthdata 구조체 포인터) - 사용자가 운동 옵션 선택 시, 운동 종목 및 해당 운동을 통해 소모하는 칼로리를 계산하여 healthdata 구조체에 저장
 - cal_diets.c
 - ◆ loadDiets(파일 포인터) - 'diets.txt' 파일을 읽고 저장
 - ◆ inputDiet(healthdata 구조체 포인터) - 사용자가 식사 옵션 선택 시, 해당 식사 및 섭취하는 칼로리를 healthdata 구조체에 저장
- 주어진 파일 내의 함수들을 채우면 동작이 됨
 - 주석에 언급된 부분에 대해서 코드 구현
- 코드를 보기 쉽게 들여쓰기를 적절히 삽입
- 함수나 변수 정의, 코드 흐름에 대해 다른 사람들이 알아볼 수 있도록

주석 삽입

- 기능 혹은 일정 코드 부분을 구현하거나 디버깅을 통해 코드를 수정할 때마다 github에 올려서 변화를 추적할 수 있도록 함
-

3. 배점 및 기준

- 프로그램 동작 점수: 70점
 - 문제 정의에 명시된 기능들이 충실하게 잘 동작하는지 여부
 - 부분적으로 동작하면 그에 맞게 부분 점수 부여 예정
 - 코드 관리 점수: 30점
 - 주석을 충실하게 달았는지 여부 (함수 및 변수 정의에 대한 설명 및 코드 흐름에 대한 설명)
 - 들여쓰기를 잘했는지 여부 [github](#)
 - Github에 코드를 구현한 이력이 잘 남아있는지 여부
 - **Copy 여부에 대해 집중적으로 볼 예정이며, 적발 시 관련자 모두 0 점 처리**
-

4. 결과물 제출 방법

- Github 계정에 별도의 repository로 올림
 - SMHealthcare 라는 이름으로 repository 생성
 - 제출 기한 시간 직전에 올라간 소스코드 파일들을 기준으로 채점
 - Github에서 SMHealthcare라는 repository가 없으면 제출하지 않은 것으로 간주할 예정이므로, 제출 후 웹 페이지 상에서 해당 repository가 보이는지 반드시 확인 요망
 - 가급적 미리 올리거나 지속적으로 여러 번 submit을 하는 형태로 진행 요망