Проект: модуль фитнес-трекера

Задача итогового проекта — создать программный модуль для фитнес-трекера, используя парадигму ООП. Этот модуль должен рассчитывать и отображать результаты тренировки.

Задание:

Разработайте программный модуль фитнес-трекера, который обрабатывает данные для трёх видов тренировок: бега, спортивной ходьбы и плавания.

Этот модуль должен выполнять следующие функции:

принимать от блока датчиков информацию о прошедшей тренировке,

определять вид тренировки,

рассчитывать результаты тренировки,

выводить информационное сообщение о результатах тренировки.

Информационное сообщение должно включать такие данные:

тип тренировки (бег, ходьба или плавание);

длительность тренировки;

дистанция, которую преодолел пользователь, в километрах;

среднюю скорость на дистанции, в км/ч;

расход энергии, в килокалориях.

❗ В ТЗ указаны формулы, которые нужно использовать для расчёта результатов тренировок. Хороший разработчик чётко следует всем требованиям ТЗ. Не меняйте формулы, используйте их в точно таком же виде, как они указаны в задании.

Структура программы:

**Базовый класс Training**

В классах, описывающих любой из видов тренировки, применяются одни и те же свойства и методы. Чтобы избежать дублирования кода, необходимо создать базовый класс Training. Он должен содержать все основные свойства и методы для тренировок. Каждый класс, описывающий определённый вид тренировки, будет дополнять и расширять базовый класс.

Методы базового класса

Метод get\_distance() - возвращает дистанцию (в километрах), которую преодолел пользователь за время тренировки.

Формула расчёта:

action \* LEN\_STEP / M\_IN\_KM

*action, тип int — количество совершённых действий (число шагов при ходьбе и беге либо гребков — при плавании).*

LEN\_STEP — расстояние, которое спортсмен преодолевает за один шаг или гребок. Один шаг — это 0.65 метра, один гребок при плавании — 1.38 метра.

M\_IN\_KM — константа для перевода значений из метров в километры. Её значение — 1000.

*Метод get\_mean\_speed()* возвращает значение средней скорости движения во время тренировки.

Формула расчёта:

преодолённая\_дистанция\_за\_тренировку / время\_тренировки

*Метод get\_spent\_calories()* возвращает количество килокалорий, израсходованных за время тренировки.

Логика подсчёта калорий для каждого вида тренировки будет своя, поэтому в базовом классе не нужно описывать поведение метода, в его теле останется ключевое слово pass.

*Метод show\_training\_info()* возвращает объект класса сообщения.

**class InfoMessage**

Это самостоятельный класс для создания объектов сообщений. У этого класса должен быть метод для вывода сообщений на экран.

Объекты этого класса создаются вызовом метода show\_training\_info() для классов тренировок.

Свойства класса InfoMessage:

training\_type — имя класса тренировки;

duration — длительность тренировки в часах;

distance — дистанция в километрах, которую преодолел пользователь за время тренировки;

speed — средняя скорость, с которой двигался пользователь;

calories — количество килокалорий, которое израсходовал пользователь за время тренировки.

У класса InfoMessage должен быть метод get\_message(), который возвращает строку сообщения:

Тип тренировки: {training\_type}; Длительность: {duration} ч.; Дистанция: {distance} км; Ср. скорость: {speed} км/ч; Потрачено ккал: {calories}.

❗ Числовые значения при выводе должны быть отформатированы до тысячных долей (до третьего знака после запятой).

**Дочерние классы**

Каждый вид спортивной активности в модуле должен быть описан соответствующим классом:

**Бег → class Running;**

**Спортивная ходьба → class SportsWalking;**

**Плавание → class Swimming.**

Конструктор каждого из классов должен получать информацию с датчиков:

action, тип int — количество совершённых действий (число шагов при ходьбе и беге либо гребков — при плавании);

duration, тип float — длительность тренировки;

weight, тип float — вес спортсмена.

Чтобы определить дистанцию, которую преодолел спортсмен, нужно число шагов или гребков, переданное в action, перевести в километры.

Для расчёта нужно добавить атрибуты класса:

**LEN\_STEP — расстояние, которое спортсмен преодолевает за один шаг или гребок. Один шаг — это 0.65 метра, один гребок при плавании — 1.38 метра.**

**M\_IN\_KM — константа для перевода значений из метров в километры. Её значение — 1000.**

Методы классов, которые отвечают за обработку данных:

расчёт дистанции, которую пользователь преодолел за тренировку: get\_distance();

расчёт средней скорости движения во время тренировки: get\_mean\_speed();

расчёт количества калорий, израсходованных за тренировку: get\_spent\_calories();

создание объекта сообщения о результатах тренировки: show\_training\_info().

Особенности реализации классов-наследников

**class Running**

Все свойства и методы этого класса без изменений наследуются от базового класса. Исключение составляет только метод расчёта калорий, его нужно переопределить.

Расход калорий для бега рассчитывается по такой формуле:

(18 \* средняя\_скорость + 1.79) \* вес\_спортсмена / M\_IN\_KM \* время\_тренировки\_в\_минутах

Важно вынести все неименованные значения в константы на уровне класса. Код станет понятнее для разработчиков, которые будут его читать.

Имена констант должны отражать их смысл, для данной формулы имена переменных могут быть такими:

CALORIES\_MEAN\_SPEED\_MULTIPLIER = 18

CALORIES\_MEAN\_SPEED\_SHIFT = 1.79

Тогда формулу для подсчёта калорий можно записать так:

((CALORIES\_MEAN\_SPEED\_MULTIPLIER \* средняя\_скорость + CALORIES\_MEAN\_SPEED\_SHIFT)

\* вес\_спортсмена / M\_IN\_KM \* время тренировки\_в\_минутах)

class SportsWalking

Конструктор этого класса принимает дополнительный параметр height — рост спортсмена.

Расчёт калорий для этого класса должен проводиться по такой формуле:

((0.035 \* вес + (средняя\_скорость\_в\_метрах\_в\_секунду\*\*2 / рост\_в\_метрах)

\* 0.029 \* вес) \* время\_тренировки\_в\_минутах)

Числовым коэффициентам тоже нужны имена, не забывайте про это. Но это не касается, например, значения степени.

class Swimming

Конструктор класса Swimming, кроме свойств базового класса, принимает ещё два параметра:

length\_pool — длина бассейна в метрах;

count\_pool — сколько раз пользователь переплыл бассейн.

В классе Swimming нужно переопределить не только метод расчёта калорий get\_spent\_calories(), но и метод get\_mean\_speed(), который рассчитывает среднюю скорость.

Формула расчёта средней скорости при плавании:

длина\_бассейна \* count\_pool / M\_IN\_KM / время\_тренировки

Формула для расчёта израсходованных калорий:

(средняя\_скорость + 1.1) \* 2 \* вес \* время\_тренировки

Есть и ещё один параметр, который надо переопределить, ведь расстояние, преодолеваемое за один гребок, отличается от длины шага. Значит, необходимо переопределить атрибут LEN\_STEP базового класса.

Имитация получения данных от блока датчиков фитнес-трекера

Прежде чем отдавать модуль в релиз, его работоспособность нужно проверить на реальном фитнес-трекере. Но у разработчиков не всегда есть такая возможность, поэтому можно имитировать работу датчиков и передать в программу заранее подготовленные тестовые данные.

В исходном коде эти данные уже подготовлены:

if \_\_name\_\_ == '\_\_main\_\_':

packages = [

('SWM', [720, 1, 80, 25, 40]),

('RUN', [15000, 1, 75]),

('WLK', [9000, 1, 75, 180]),

]

for workout\_type, data in packages:

training = read\_package(workout\_type, data)

main(training)

Данные для тестов

Блок датчиков фитнес-трекера передаёт пакеты данных в виде кортежа, первый элемент которого — кодовое обозначение прошедшей тренировки, второй — список показателей, полученных от датчиков устройства. Для проверки были смоделированы пакеты для каждого вида тренировки и добавлены в список packages:

packages = [

('SWM', [720, 1, 80, 25, 40]),

('RUN', [15000, 1, 75]),

('WLK', [9000, 1, 75, 180]),

]

Последовательность данных в принимаемых пакетах:

Плавание

Код тренировки: 'SWM'.

Элементы списка: количество гребков, время в часах, вес пользователя, длина бассейна, сколько раз пользователь переплыл бассейн.

Бег

Код тренировки: 'RUN'.

Элементы списка: количество шагов, время тренировки в часах, вес пользователя.

Спортивная ходьба

Код тренировки: 'WLK'.

Элементы списка: количество шагов, время тренировки в часах, вес пользователя, рост пользователя.

Программа должна перебирать в цикле список пакетов, распаковывает каждый кортеж и передаёт данные в функцию read\_package().

Функция чтения принятых пакетов read\_package()

Функция read\_package() должна определить тип тренировки и создать объект соответствующего класса, передав ему на вход параметры, полученные во втором аргументе. Этот объект функция должна вернуть.

Функция read\_package() должна принимать на вход код тренировки и список её параметров.

read\_package(workout\_type, data)

В теле функции (или рядом с ней) должен быть словарь, в котором сопоставляются коды тренировок и классы, какие нужно вызвать для каждого типа тренировки.

Функция main()

Функция main() должна принимать на вход экземпляр класса Training.

При выполнении функции main() для этого экземпляра должен быть вызван метод show\_training\_info(); результатом выполнения метода должен быть объект класса InfoMessage, его нужно сохранить в переменную info.

Для объекта InfoMessage, сохранённого в переменной info, должен быть вызван метод, который вернёт строку сообщения с данными о тренировке; эту строку нужно передать в функцию print().

Задача описана, можно приступать к её выполнению. Удачи!  
  
  
  
**Когда все правильно сделаете – в терминале будет такое сообщение:**Тип тренировки: Swimming; Длительность: 1.000 ч.; Дистанция: 0.994 км; Ср. скорость: 1.000 км/ч; Потрачено ккал: 336.000.

Тип тренировки: Running; Длительность: 1.000 ч.; Дистанция: 9.750 км; Ср. скорость: 9.750 км/ч; Потрачено ккал: 699.750.

Тип тренировки: SportsWalking; Длительность: 1.000 ч.; Дистанция: 5.850 км; Ср. скорость: 5.850 км/ч; Потрачено ккал: 157.500.