1. Inicializar vector global tipo package “storage data” para almacenar los paquetes de datos
2. Definir función “check\_correct\_data(package data)” :

* Si “data.segundos <= 0” o “data.steps <= 0” retornar false.
* Sino retornar true.

1. Definir función “check\_correct\_time(long seconds)”:

* Si “seconds < 0” o “seconds >= 86400” retornar false.
* Si “storage\_data.empty()” es true, retornartrue.
* Sino retornar “seconds > storage\_data.back().segundos”

1. Definir función “get\_step\_day(int steps)”:

* Iniciar variable “total” en 0.
* Bucle que recorre “storage\_data” y suma “total” con los pasos.
* Retorna “total”.

1. Definir función “get\_distance(int steps)”:

* Retornar “(steps \* STEP\_M) / 1000.0”

1. Definir función “get\_calories\_burned/double dist, times current\_time)”:

* Calcular hours, mean\_speed, minutes y calories con las fórmulas dadas y retornar calories.

1. Definir función “get\_achievement(double dist)”:

* Retornar mensaje de motivación según la distancia recorrida.
* Si “dist >= 6.5”, retornar "Gran entrenamiento! Objetivo cumplido.".
* Si “dist >= 3.9”, retornar "¡Nada mal! Hoy ha sido un dia productivo.".
* Si “dist >= 2.0”, retornar "Menos que el resultado deseado, ¡pero intenta alcanzarlo mañana!".
* Sino retornar "Esta bien tomarse el dia de descanso. No siempre se puede ganar.".

1. Definir función “get\_time\_hms (long seconds)”:

* Declaramos variable “time” tipo times.
* Convertimos “seconds” a structure times.
* time.hour = seconds / 3600.
* time.min = (seconds % 3600) / 60.
* time.sec = seconds % 60.
* Retornar time.

1. Definir función “format\_time (times t)”:

* Hacemos que horas, minutos y segundos se muestre formalmente.

1. Definir función “show\_message (times t, int steps, double dist, double calories, string achievement)”:

* Mostramos los mensajes con los datos indicados y el tiempo con la función de format\_time.

1. Definir función “accept\_package(package data)”:

* Verificamos datos y el tiempo, si están incorrectos devolvemos el storage\_data.
* Si están correctos se almacena el paquete, se calcula los datos con las funciones y se muestra el mensaje.
* Retorna “storage\_data”.

PRUEBA DE ENTRADA Y SALIDA:

**ENTRADA:** {7201, 505}

**SALIDA:** Tiempo: 02:00:01.

Pasos dados hoy: 505.

La distancia fue 0.33 km.

Has quemado 315.08 cal.

Esta bien tomarse el dia de descanso. No siempre se puede ganar.

**ENTRADA:** {0, 505}

**SALIDA:** Paquete inválido, por lo cual no se muestra el mensaje.

**ENTRADA:** {34562, 15000}

**SALIDA:** Tiempo: 09:36:02.

Pasos dados hoy: 15505.

La distancia fue 10.08 km.

Has quemado 1519.98 cal.

Gran entrenamiento! Objetivo cumplido.

**ENTRADA:** {34562, 9000}

**SALIDA:** Tiempo duplicado, por lo cual no se muestra el mensaje.

**ENTRADA:** {28862, 7600}

**SALIDA:** Tiempo menor que el último añadido, por lo cual no se muestra el mensaje.