"Gradanator"

Aquest programa interactiu se centra en instruccions if/else, entrada de dades i retorn de valors. Lliura un fitxer anomenat `gradanator.py`. El programa llegeix com a entrada les qualificacions d'un estudiant en els treballs i tres exàmens i les utilitza per calcular la nota final del curs de l'estudiant.

El programa comença amb un missatge d'introducció que explica breument el programa. A continuació, el programa llegeix les qualificacions en quatre categories: parcial 1, parcial 2, treballs i final. Cada categoria té un pes assignat: els punts es ponderen en una fracció del 100% de la nota final del curs. Quan el programa comença a llegir cada categoria, primer demana el pes de la categoria.

L'usuari comença introduint les puntuacions obtingudes al parcial 1. Les notes dels exàmens estan limitades a un màxim de 100; Es mostra la "nota ponderada" del parcial, que és igual a la puntuació de l'usuari multiplicada pel pes de l'examen.

A continuació, el programa demana informació sobre el parcial 2 i després sobre el final. El comportament per a cadascun és el mateix que per al parcial 1.

Després, el programa demanarà la nota de l'exàmen final i el seu pes. A continuació, l'usuari introdueix informació sobre els treballs, inclòs el pes i quants treballs es van assignar. Per a cada treball, l'usuari introdueix una nota i els punts possibles. Part de la nota dels treballs prové de les seccions a les quals s'ha assistit.

Un cop el programa ha llegit la informació de l'usuari sobre els dos exàmens i els treballs, imprimeix el percentatge global obtingut en el curs, que és la suma de les notes ponderades de les quatre categories, tal com es mostra a continuació:

$$Grade = WeightedMidterm1Score + WeightedMidterm2Score + WeightedFinalExamScore + WeightedHomeworkScore$$

$$Grade = \left(\frac{78}{100} x\ 10\right) + \left(\frac{84}{100} x\ 10\right) + \left(\frac{100}{100} x\ 30\right) + \left(\frac{14+17+19}{15+20+25} x\ 50\right)$$

$$Grade = 7.8 + 8.4 + 30 + 41.67$$

Grade = 87.87

El programa imprimeix la nota que l'estudiant obtindrà al curs,

La qualificació serà basada en l'escala següent.

- 90% o més: A
- 89.99% - 80%: B
- 79.99% - 70%: C
- 69.99% - 60%: D
- menys del 60%: F

Després d'imprimir la nota mínima garantida, imprimeix un missatge personalitzat sobre la qualificació. Aquest missatge ha de ser diferent per a cada rang de nota mostrat més amunt.

Aquest programa processa l'entrada de l'usuari amb `input`. Hauries de gestionar els següents dos casos especials:

1. Un estudiant pot rebre punts extres en una tasca individual, però la nota total dels treballs està limitada al màxim possible. Per exemple, un estudiant pot obtenir una nota de 22/20 en un treball, però si la nota total dels treballs és 63/60, aquesta es limitarà a 60/60..
2. Limita les puntuacions dels exàmens a 100. Si la puntuació original de l'examen supera 100, es fa servir una puntuació de 100.

En cas contrari, pots assumir que l'usuari introdueix una entrada vàlida. Quan es demani un valor, l'usuari introduirà un enter dins del rang adequat. L'usuari introduirà un nombre de treballs ≥ 1, i la suma dels quatre pesos serà exactament 100. El pes de cada categoria serà un número no negatiu. Els ajustos d'examen seran ≥ 0.

Estratègia de desenvolupament i consells:

- Aborda parts del programa (parcial 1, parcial 2, treballs, examen final) una a una, en lloc d'escriure tot el programa de cop. Escriu una mica de codi, fes-lo funcionar i prova el que tens fins ara. Si intentes escriure grans quantitats de codi sense executar-lo, podries trobar una llista llarga d'errors i/o bugs.
- Per calcular les notes dels treballs, hauràs de sumar acumulativament no només els punts totals que l'estudiant ha obtingut, sinó també els punts totals possibles de tots els treballs.
- Totes les notes ponderades i qualificacions es mostren amb com a màxim 1 decimal. Aconsegueix-ho amb una funció personalitzada o `round`. El següent codi imprimeix la variable `x` arrodonida a la desena més propera:

```
x = 1.2345
print("x arrodonit a la desena més propera és " + round(x, 1)) # 1.2
```

Directrius d'estil:

- Només fer servir les biblioteques que hem estudiat.
- No has de tenir declaracions `print` a la funció principal (`main`). A més, `main` ha de ser un resum concís del programa en general; `main` hauria de fer crides a diverses altres funcions que implementin la major part del comportament del programa. Les teves funcions hauran de fer un ús adequat de paràmetres i valors retornats. Cada funció ha de realitzar una tasca coherent i no fer una part massa gran del treball general. Evita l'"encadenament" extens de crides a funcions, on cada funció crida a la següent, no es retornen valors i el control no torna a `main`.
- Dona noms significatius a funcions i variables, i utilitza la indentació i els espais correctes. Segueix els estàndards de nomenclatura de Python especificats a la lliçó. Localitza variables sempre que sigui possible; declara-les dins del menor abast necessari. Inclou encapçalaments de comentaris significatius a la part superior del programa i a l'inici de cada funció. Limita la longitud de les línies a 100 caràcters.