

Documentación Ahorcado

1r DAM M03



Programació

Sergio Rey García

Joan Dalmau

# Descripció

## De que tracta el projecte

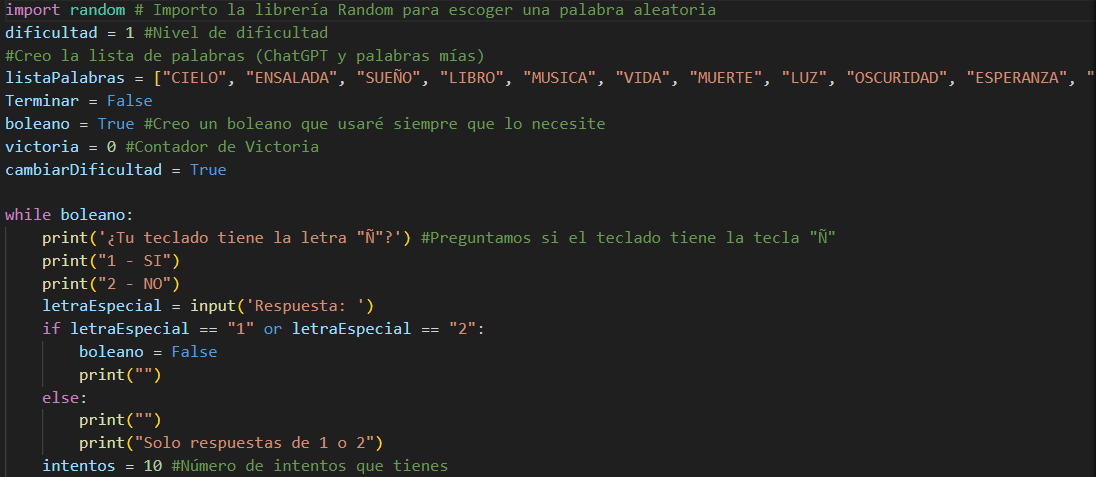
El projecte tracta de recrear el joc "El Ahorcado" amb el llenguatge de programació Python.

Les regles que havíem de seguir principalment eren les següents:

* La paraula secreta s’ha d’agarrar aleatòriament d’una llista.
* El jugador té 1 sola vida per resoldre les 3 paraules ocultes.
* Per cada fallada, el joc ha de mostrar un dibuix de l’estalvi.
* Hi ha d’haver tres nivells de dificultat. Els nivells de dificultat afecten els intents del jugador, reduint la possibilitat de trobar la paraula oculta.
* La paraula oculta s’ha d’anar mostrant per cada lletra que és encertada.
* No es poden fer servir funcions associades a cadenes de caràcter.

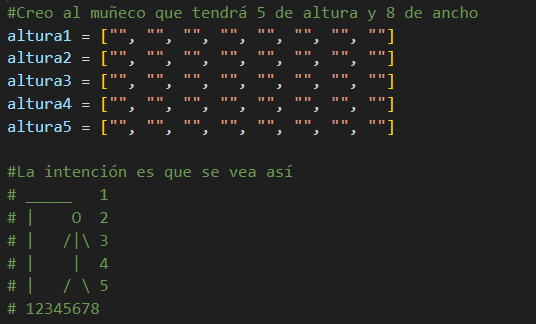
# Disseny i funcionament del programa

## Com està estructurat



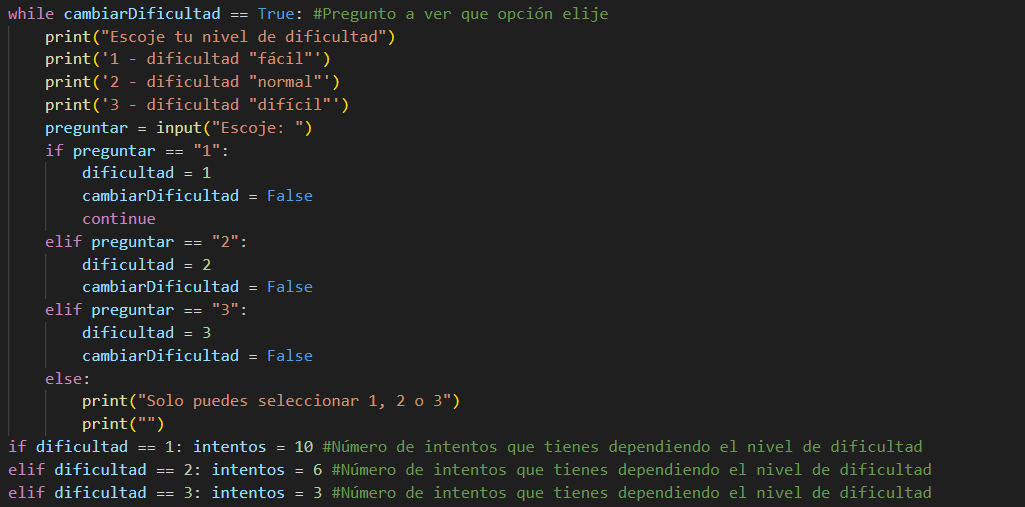
Començo important la llibreria “random” per escollir la paraula, ajustant el nivell de dificultat i creant les variables principals com la de victòria o terminar.

Després, pregunto si el teclat té la lletra “Ñ”, per si algú té un teclat anglès.



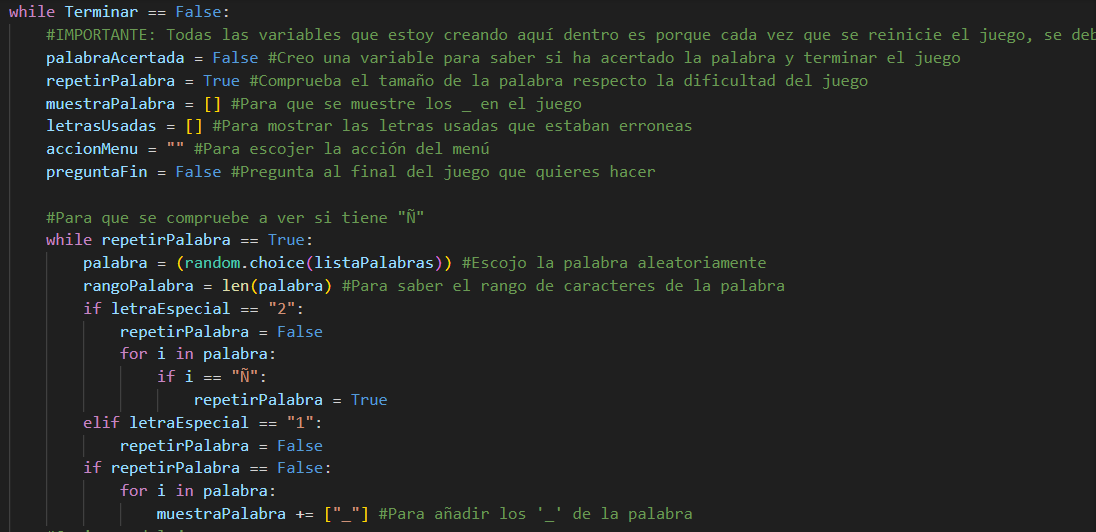
Després, creo el ninot.

Jo el ninot he decidit fer-ho amb llistes. Tinc companys que ho han fet amb “print()”, però al moment de crear el codi no sem va acudir i vaig continuar fer-ho amb llistes.



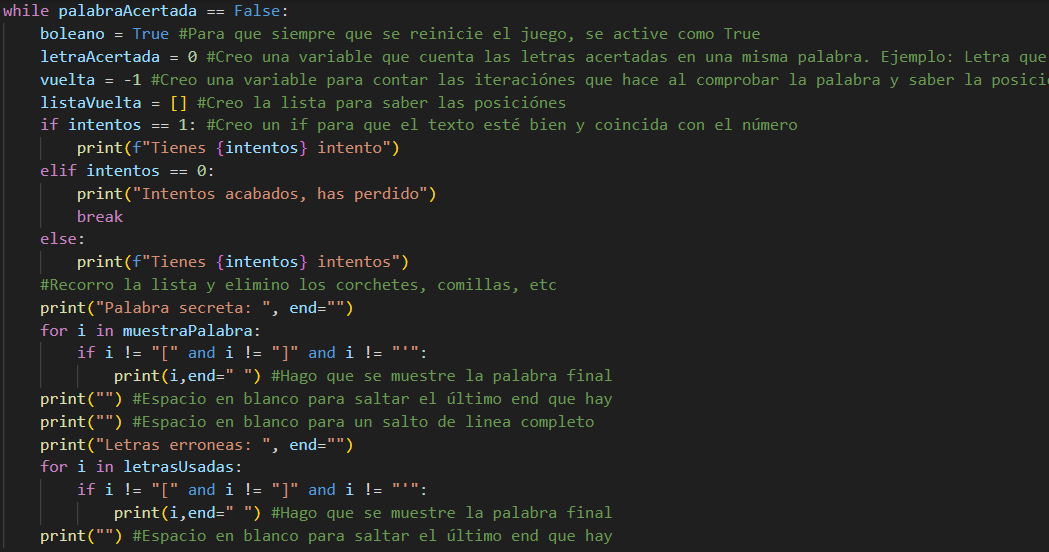
En aquesta part del codi, faig un bucle per preguntar la dificultat que la persona vol. Depenent de la dificultat, el número d’intents cambia.

En aquesta part del codi i en moltes altres mes, no faria falta un bucle per el funcionament del codi, però jo volia fer un “codi sense errors”, es a dir, un codi que no es pugues petar de ninguna forma.



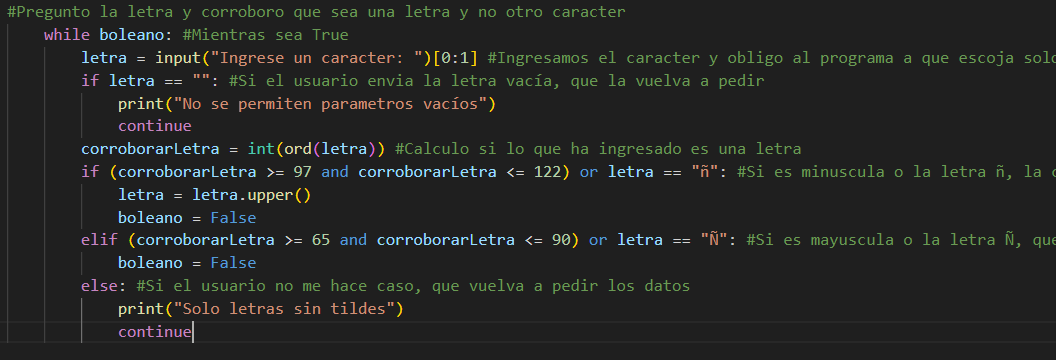
En aquesta part del codi, escollo la paraula aleatòria i corroboro si té la lletra “Ñ” per si la persona no té aquesta lletra.

Després, poso la llargada de la paraula en una llista ple de “\_” per ocultar la paraula i que es mostri oculta.

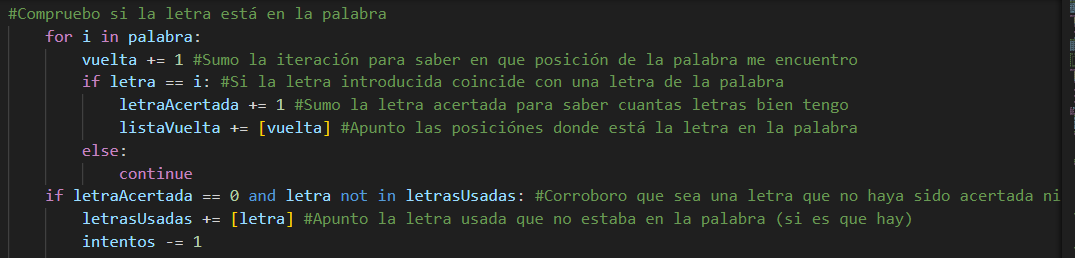


Aquí comença el lloc.

El primer que faig es mostrar els intents que tens, seguidament mostro la paraula secreta eliminant el “[”, “]” i “ ’ ”

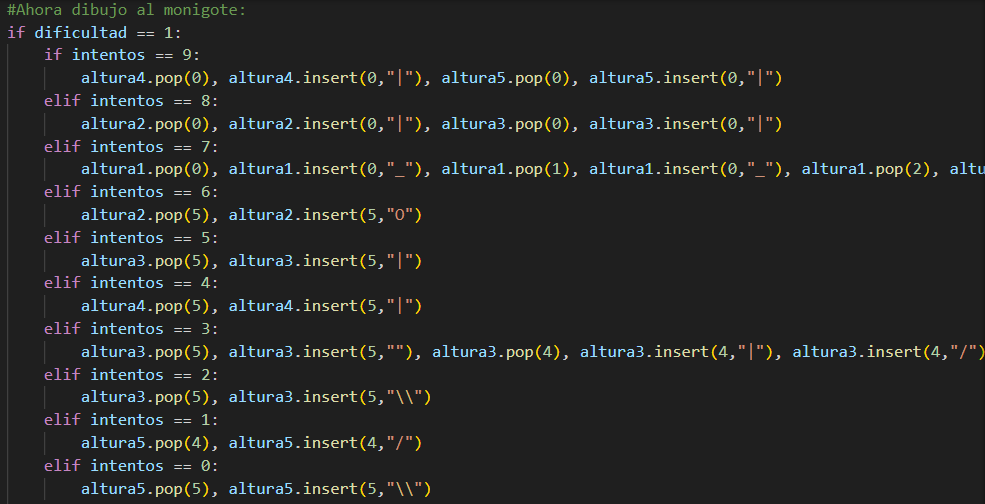


Pregunto la lletra que vol ingressar i nomes em quedo amb la primera lletra ingressada, per si l’usuari ha premut dues tecles, nomes emmagatzemar la primera. Corroboro que sigui una lletra i no un caràcter amb l’exercici que vam fer del codi ascii i continuo si està tot bé.

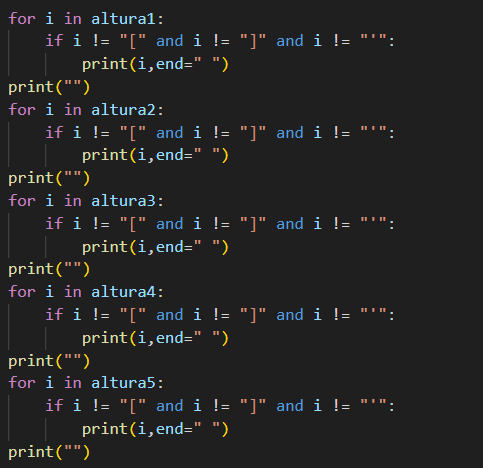


Comprovo si la lletra la lletra hi es en la paraula amb el bucle, recorrent la paraula lletra a lletra i apuntant la posició on hi es.

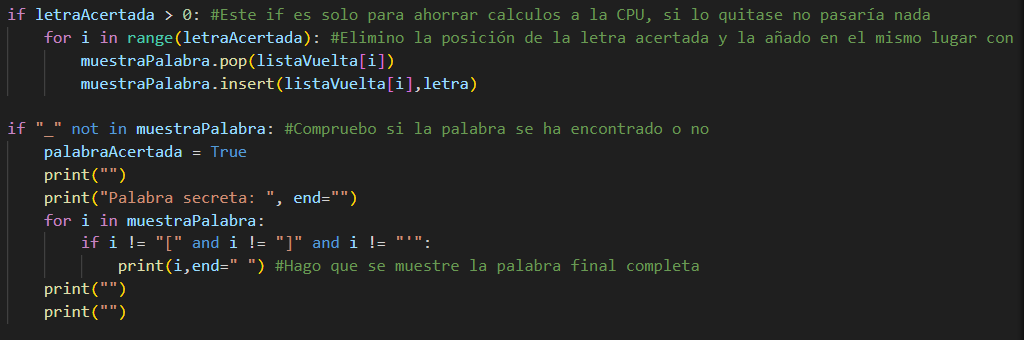
Si la lletra no hi es en la paraula i no hi es en la llista de lletres fallades, que s’apunti a la llista i et redueixi un intent per fallar.



Després dibuixo el ninot afegint i eliminant posicions a la llista que he creat abans de començar el joc. El ninot es construeix mes ràpid o mes lent depenent del nivell de dificultat.

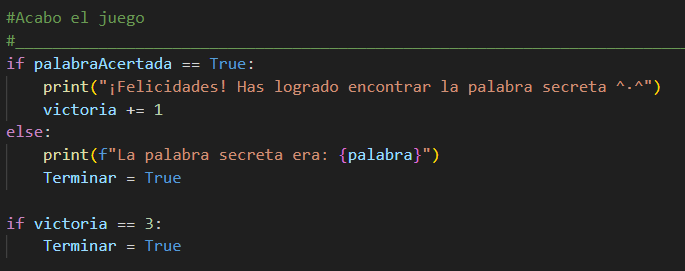


Després dibuixo el ninot amb bucles que recorren la llista i no afegeixen les claus o l’apòstrof.



I al final del joc elimino la posició de la llista de la paraula on hi es les “\_”, canviant-les per la lletra encertada. La “letraAcertada” es la llista que es guardava les posicions de la paraula encertada.

Després corroboro si la paraula s’ha encertat, fent un “if “\_” not in muestraPalabra:”. Si s’ha encertat, el joc continua, sinó, el bucle des de preguntar la lletra es repeteix.



Per últim, al final del codi, comprovo si a ha encertat o si les ha fallat totes. Per això, utilitzo “palabraAcertada” que activo nomes quan s’encerta la paraula. També faig que si has guanyat tres vegades, que el joc termini.

# Problemes al codi

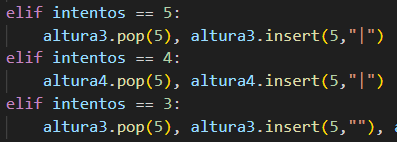
## Els problemes que m’he trobat

A l’hora de fer el programa, he tingut la sort de tindre pocs problemes, però el que podria dir que es el més complex va ser crear el ninot.

Com a l’hora de crear el ninot, jo ho vaig fer amb tables, vaig tindre un error de que el cos no estigues a la seva posició. Es a dir, a partir de “x” fallo, el cos del ninot es desplaçava a la dreta o a l’esquerra, i això feia que no sortís el ninot bé.

Per solucionar-ho, vaig tenir que fer proba i error, intent rere intent per veure l’evolució del ninot. Al final ho vaig solucionar eliminant la posició on hi havia l’error i posant-la a la dreta o esquerra

Exemple:

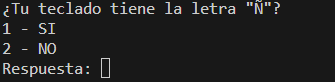


Al “elif intentos == 5:” elimino la posició 5 de l’altura 3 i insereixo la columna vertebral del cos del ninot, i a l’”elif intentos == 3:” elimino la mateixa posició de la mateixa taula per inserir espai. Mes endavant en el codi, insereixo la columna al ninot a la posició 4

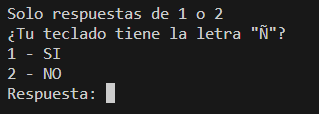


# Joc de proves

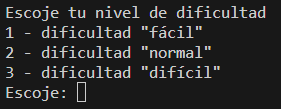
## Com jugar i provocar errors



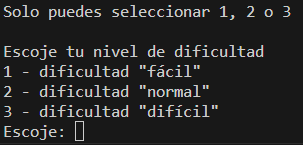
El joc comença preguntant si el teu teclat te la lletra “Ñ”. Si poses alguna resposta que no deus posar, com per exemple un caràcter, altre número o un espai, succeeix això



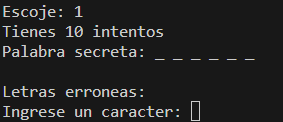
I et torna a preguntar, ja que està dins d’un bucle i nomes es permet el valor “1” o “2” que està en format string.



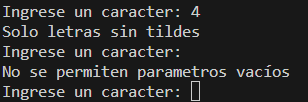
Desprès ens pregunta el nivell de dificultat. Si tornem a posar altre cosa que no es valida, surt aquet missatge



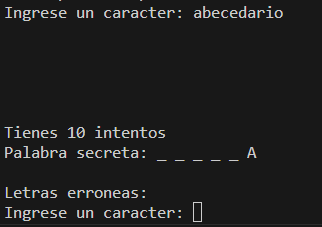
Un altre vegada, això son valors en strings i nomes es pot permetre una d’aquestes opcions.



Continua el lloc, si tornem a intentar petar-ho surt això

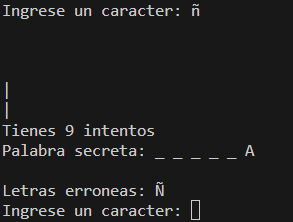


I si provem a posar una paraula sencera, passa això

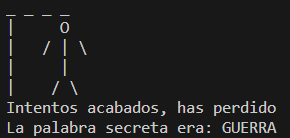


Nomes es queda amb la primera lletra de tota la paraula. En cas de posar una paraula que comenci amb un caràcter especial, em sortirà “Solo letras sin tildes”

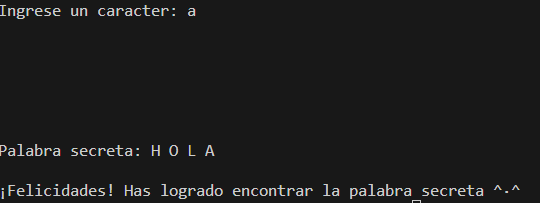
Si fallo una lletra, passa això



Així fins perdre



En cas de guanyar, passa això (he quitat l’aleatori per poder mostrar-ho)



# Conclusions

## Millores que posaria i que he après

El treball m’ha agradat molt i em sembla molt divertit i útil per aprendre altres formes de fer les coses, a mes a mes de estructurar el codi i aprendre a comentar les coses. Sobre tot, ha sigut útil per refrescar codi i aprendre al cent per cent com utilitzar les diferents opcions que he posat, com per exemple el “.pop” i “.insert”, o el “not in”, entre altres.

Hi han moltes coses per millorar, per exemple els comentaris del codi, la seva estructuració o la seva optimització. També podria millorar la seva interfície o el seu us.