**Proyecto Courier Quest**

**Estructuras de datos**

**Bitacora de PROMPTS**

**Integrantes:** Jarod Bonilla Granados, Alex Quesada Bermudez, Oscar Gutiérrez Rosales.

**Consideraciones**

* El proyecto debe realizarse en grupos de máximo 3 personas.
* No se permite la entrega individual. Además, pueden trabajar con personas de otro horario
* El proyecto debe estar alojado en un repositorio de Github (a la hora de la entrega, solamente deben proporcionar el link donde está el código fuente del programa).
* Está permitido consultar documentación y pedir ayuda a un asistente de IA, pero deben registrar en un archivo de bitácora los prompts utilizados y las modificaciones realizadas.
* Se penalizará la copia directa sin explicación del código.
* El código debe cumplir las normas PEP8, y debe estar debidamente documentado.
* Fecha de entrega: Domingo 05 de septiembre a las 11:59pm.
* Deben incluir un archivo README.md con información general del proyecto, incluyendo las estructuras de datos que utilizaron y para qué partes del programa. Así como detalles de la complejidad algorítmica.

**Criterios de evaluación (100%)**

* Estructuras utilizadas (20%)
* Algoritmos y rendimiento (20%)
* API+caché (15%)
* Archivos (15%)
* Jugabilidad (20%)
* Código y Documentación (10%)

**La información del mundo de juego (ciudad, pedidos, clima, etc.) será proporcionada a través de un API alojado en:**

<https://tigerds-api.kindflower-ccaf48b6.eastus.azurecontainerapps.io>

**Documentación:**

ccaf48b6.eastus.azurecontainerapps.io/docs

GET /city/map → JSON de mapa

GET /city/jobs → JSON de pedidos

GET /city/weather → JSON de ráfagas de clima

**Normas PEP8:**

****

**Apartado de PROMPS**

(colocar parte del código modificada junto con el prompt y una breve explicación del uso)

PROMPT: Al principio el repartidor tiene pedidos regados por el mapa, y necesita ver los pedidos cercanos, pero después estos pedidos del mapa se eliminan y solo quedan en su inventario, necesito que el algoritmo sirva tanto para pedidos que se recogen por primera vez, hasta cuando ya no están en el mapa y se necesita saber la distancia del drop of, (Le pase el codigo que tenia).

Código: básicamente había hecho el algoritmo, pero no me servía para buscar el la lista general y de inventario, entonces le pedí ayuda para juntar la logica, asi como tambien previamente tome el método distance para calcular la distancia más cercana. De forma resumida se toman tanto la lista global de pedidos, como los del inventario, se itera sobre ellos y sacamos el pedido con distancia mínima entre el courier y el drop off o pickup para luego retornarlo. Distance saca la distance por pitágoras.

def job\_most\_nearly(self, curr\_position):

"""Busca el job más cercano a la posición del jugador."""

candidates = list(self.jobs) + list(self.courier.inventory.get\_all())

if not candidates:

return None

nearest: Job | None = None

min\_dist = float("inf")

for job in candidates:

if job in self.courier.inventory.get\_all():

dist = self.distance(job.dropoff, curr\_position)

else:

dist = self.distance(job.pickup, curr\_position)

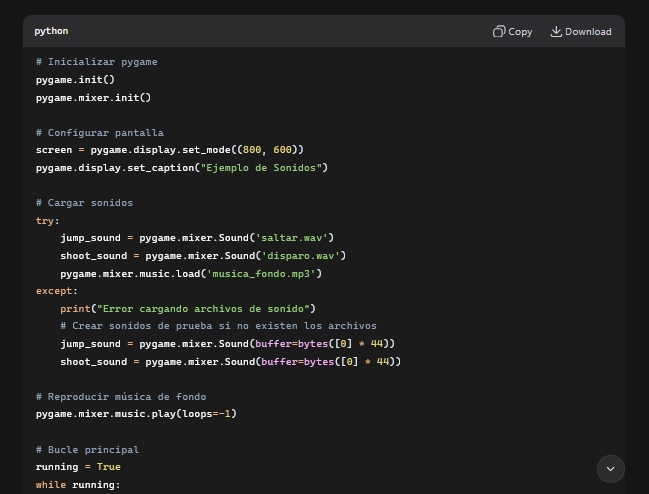
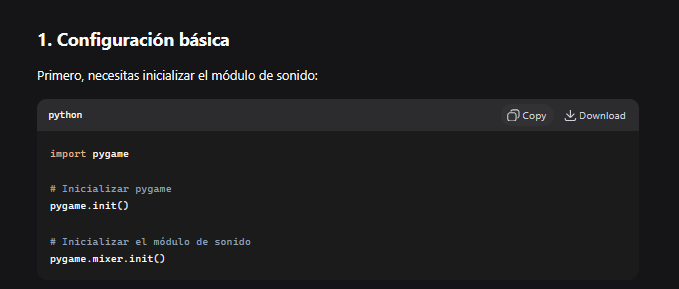
if dist <= 5 and dist < min\_dist:

nearest = job

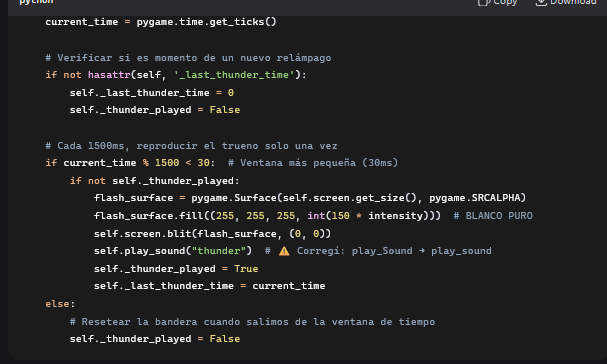
min\_dist = dist

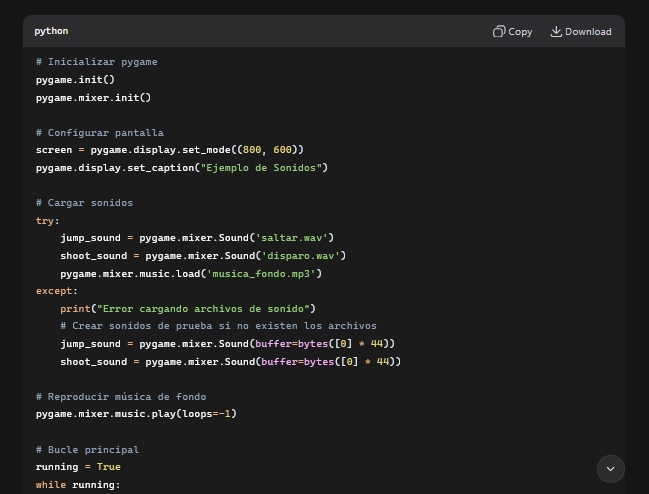
return nearest

Prompt: como puedo meter sonidos en python



Prompt: if pygame.time.get\_ticks() % 1500 < 100: flash\_surface = pygame.Surface(self.screen.get\_size(), pygame.SRCALPHA) flash\_surface.fill((255, 255, 255, int(150 \* intensity))) # BLANCO PURO self.screen.blit(flash\_surface, (0, 0)) self.play\_Sound("thunder") por que suena como lento el rayo

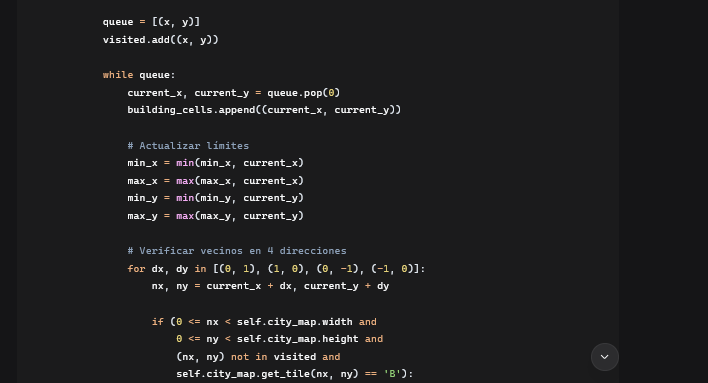


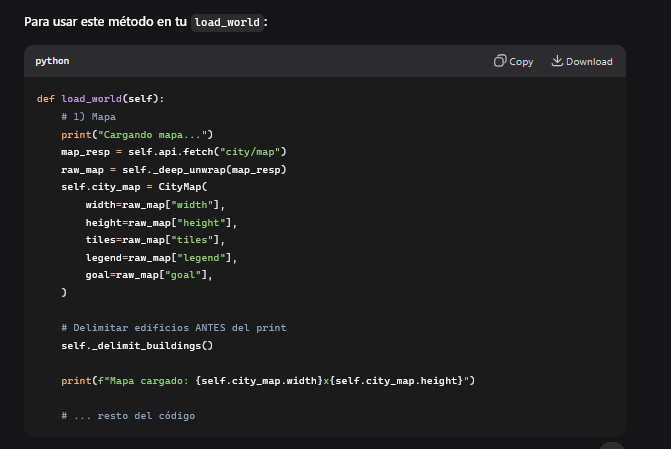


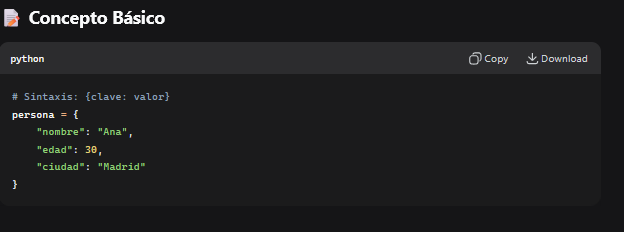
Prompt: Quiero que se ejecute una vez por 60 s

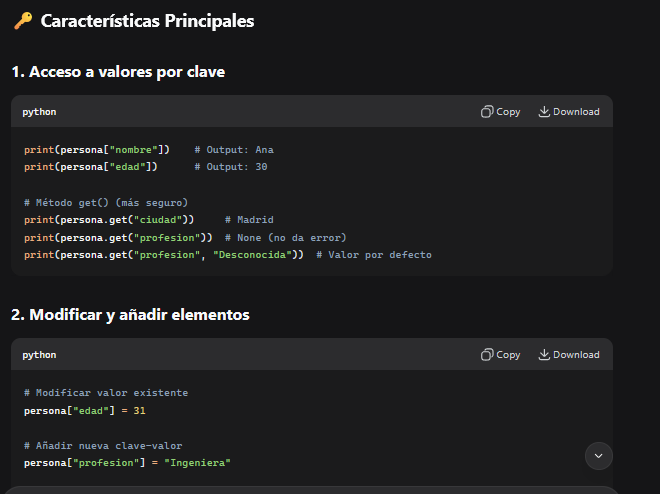
Prompt: def \_delimit\_buildings(self): for y in range(self.city\_map.height): for x in range (self.city\_map.width): completalo que delimite los edificios de la matriz que se identifican con una B



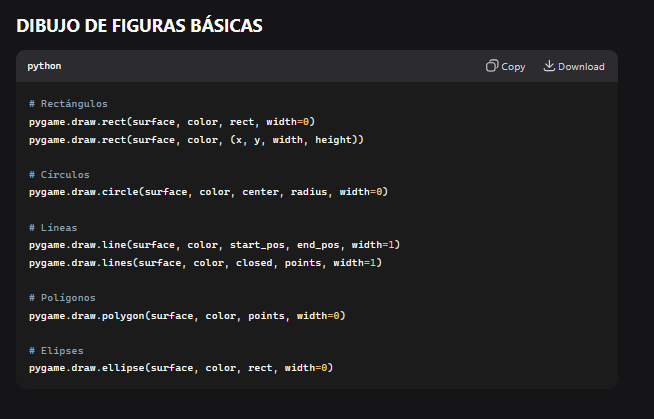


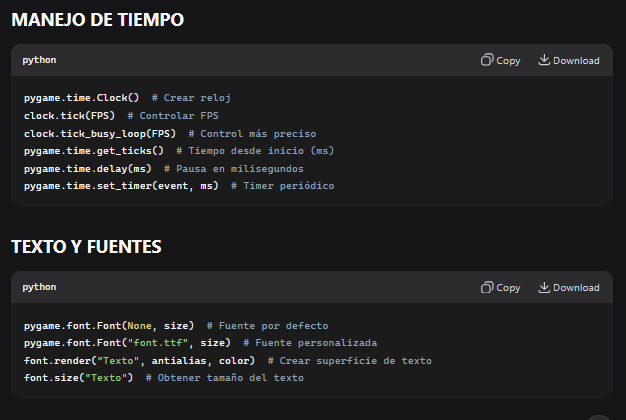
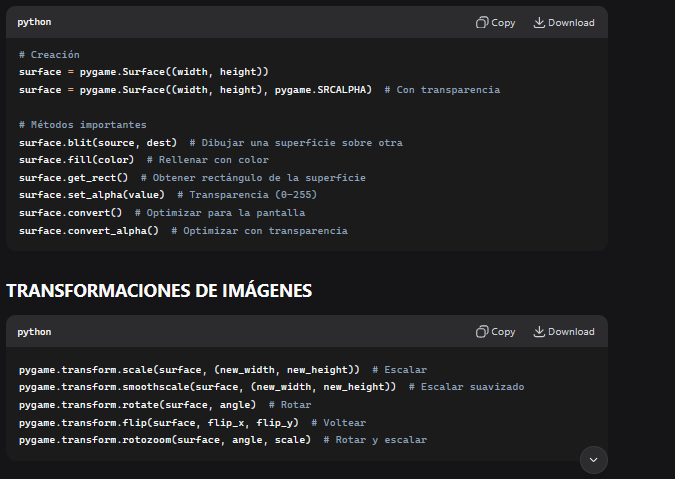


Promt:como funcionan los diccionarios

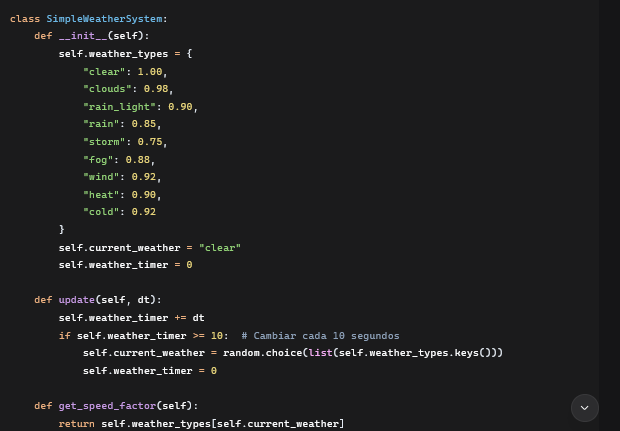


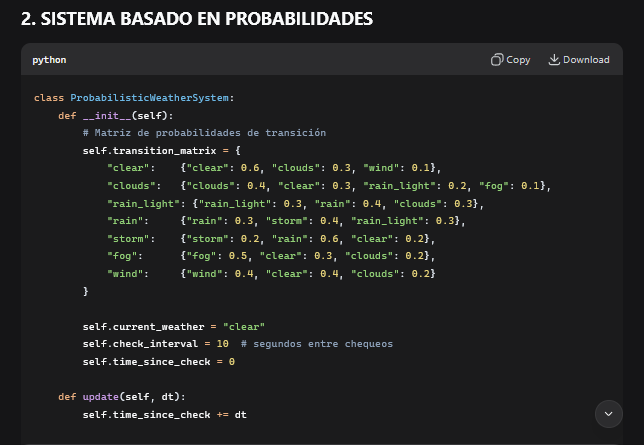


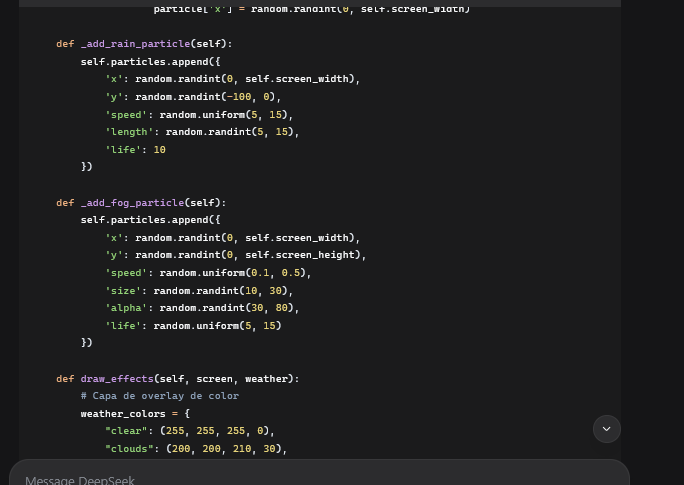
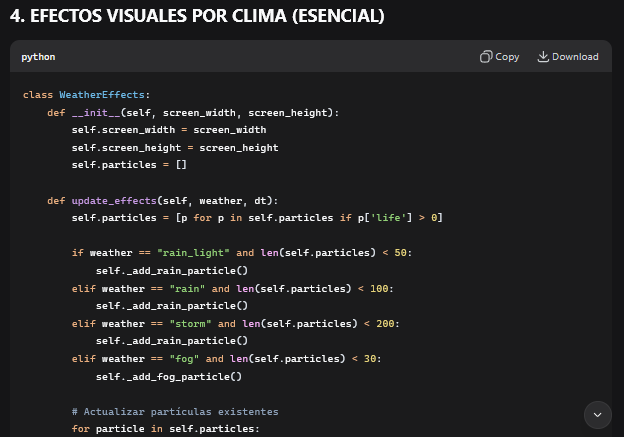
Prompt:metodos de pygame



Promt: como se puede simular el clima - clear: ×1.00 - clouds: ×0.98 - rain\_light: ×0.90 - rain: ×0.85 - storm: ×0.75 - fog: ×0.88 - wind: ×0.92 - heat: ×0.90 - cold: ×0.92 en estas condiciones en pygame y explicame cual es la mejor forma de simular cambios climaticos en pygame







Prompt: dame ejemplos de como se ven y se realizan peticiones de un api ademas de como se gurada la informacion

