import random

# Create a list of strings and assign

# the list to a variable named words.

words = ["boy", "girl", "cat", "dog", "bird", "house"]

# Call the random.choice function which will choose

# one string from the words list. Store the chosen

# string in a variable named word.

word = random.choice(words)

def get\_determiner(quantity):

"""Return a randomly chosen determiner. A determiner is

a word like "the", "a", "one", "two", "some", "many".

If quantity == 1, this function will return either "a",

"one", or "the". Otherwise this function will return

either "two", "some", "many", or "the".

Parameter

quantity: an integer.

If quantity == 1, this function will return

a determiner for a single noun. Otherwise this

function will return a determiner for a plural noun.

Return: a randomly chosen determiner.

"""

if quantity == 1:

words = ["a", "one", "the"]

else:

words = ["two", "some", "many", "the"]

# Randomly choose and return a determiner.

word = random.choice(words)

return word

nouns = ["bird", "boy", "car", "cat", "child","dog", "girl", "man", "rabbit", "woman"]

noun = random.choice(nouns)

def get\_noun(quantity):

if quantity == 1:

nouns = ["bird", "boy", "car", "cat", "child","dog", "girl", "man", "rabbit", "woman"]

else:

nouns = ["birds", "boys", "cars", "cats", "children","dogs", "girls", "men", "rabbits", "women"]

noun = random.choice(nouns)

return noun

def get\_verb(quantity, tense):

if tense == 'past':

verbs = ["drank", "ate", "grew", "laughed", "thought", "ran", "slept", "talked", "walked", "wrote"]

elif tense == 'present':

verbs = ["drink", "eat", "grow", "laugh", "think", "run", "sleep", "talk", "walk", "write"]

elif tense == 'present' and quantity != 1:

verbs = ["drinks", "eats", "grows", "laughs", "thinks", "runs", "sleeps", "talks", "walks", "writes"]

elif tense == "future":

verbs = ["will drink", "will eat", "will grow", "will laugh", "will think", "will run", "will sleep", "will talk", "will walk", "will write"]

verb = random.choice(verbs)

return verb

def get\_preposition():

preposition = ["about", "above", "across", "after", "along", "around", "at", "before", "behind", "below", "beyond", "by", "despite", "except", "for",

"from", "in", "into", "near", "of", "off", "on", "onto", "out", "over", "past", "to", "under", "with", "without"]

prepositionrand = random.choice(preposition)

return prepositionrand

def get\_prepositional\_phrase(quantity):

"""Build and return a prepositional phrase composed of three

words: a preposition, a determiner, and a noun by calling the

get\_preposition, get\_determiner, and get\_noun functions.

Parameter

quantity: an integer that determines if the determiner

and noun in the prepositional phrase returned from

this function are single or pluaral.

Return: a prepositional phrase.

"""

prepositionphrase = f"{get\_preposition()} {get\_determiner(quantity)} {get\_noun(quantity)}"

return prepositionphrase

def get\_quantity():

return int(input("What is the quantity you want?:"))

#print(get\_prepositional\_phrase(get\_quantity()), get\_prepositional\_phrase(get\_quantity()))

def main(quantity):

sentence1 = print(f"{get\_determiner(quantity=1)} {get\_noun(quantity = 1)} {get\_verb(quantity,'past')} {get\_prepositional\_phrase(quantity)}")

sentence2 = print(f"{get\_determiner(quantity=1)} {get\_noun(quantity = 1)} {get\_verb(quantity,'present')} {get\_prepositional\_phrase(quantity)}")

sentence3 = print(f"{get\_determiner(quantity=1)} {get\_noun(quantity = 1)} {get\_verb(quantity,'future')} {get\_prepositional\_phrase(quantity)}")

sentence4 = print(f"{get\_determiner(quantity=2)} {get\_noun(quantity = 2)} {get\_verb(quantity,'past')} {get\_prepositional\_phrase(quantity)}")

sentence5 = print(f"{get\_determiner(quantity=3)} {get\_noun(quantity = 3)} {get\_verb(quantity,'present')} {get\_prepositional\_phrase(quantity)}")

sentence6 = print(f"{get\_determiner(quantity=4)} {get\_noun(quantity = 4)} {get\_verb(quantity,'future')} {get\_prepositional\_phrase(quantity)}")

return sentence1, sentence2, sentence3, sentence4, sentence5, sentence6

main(get\_quantity())