T1A3

Project : Terminal Application

By Chang-Ha LEE

2022

Walk - Through

What is the project about?

LionWolf Haus

- Restaurant Tycoon -

Walk - Through

What is the project about?

“Terminal application”

+

“Interactive with user input”

Walk - Through

What do you do in this game?

Business Simulation Game

Order stocks + Sell them and earn money + Give instructions to solve the problem

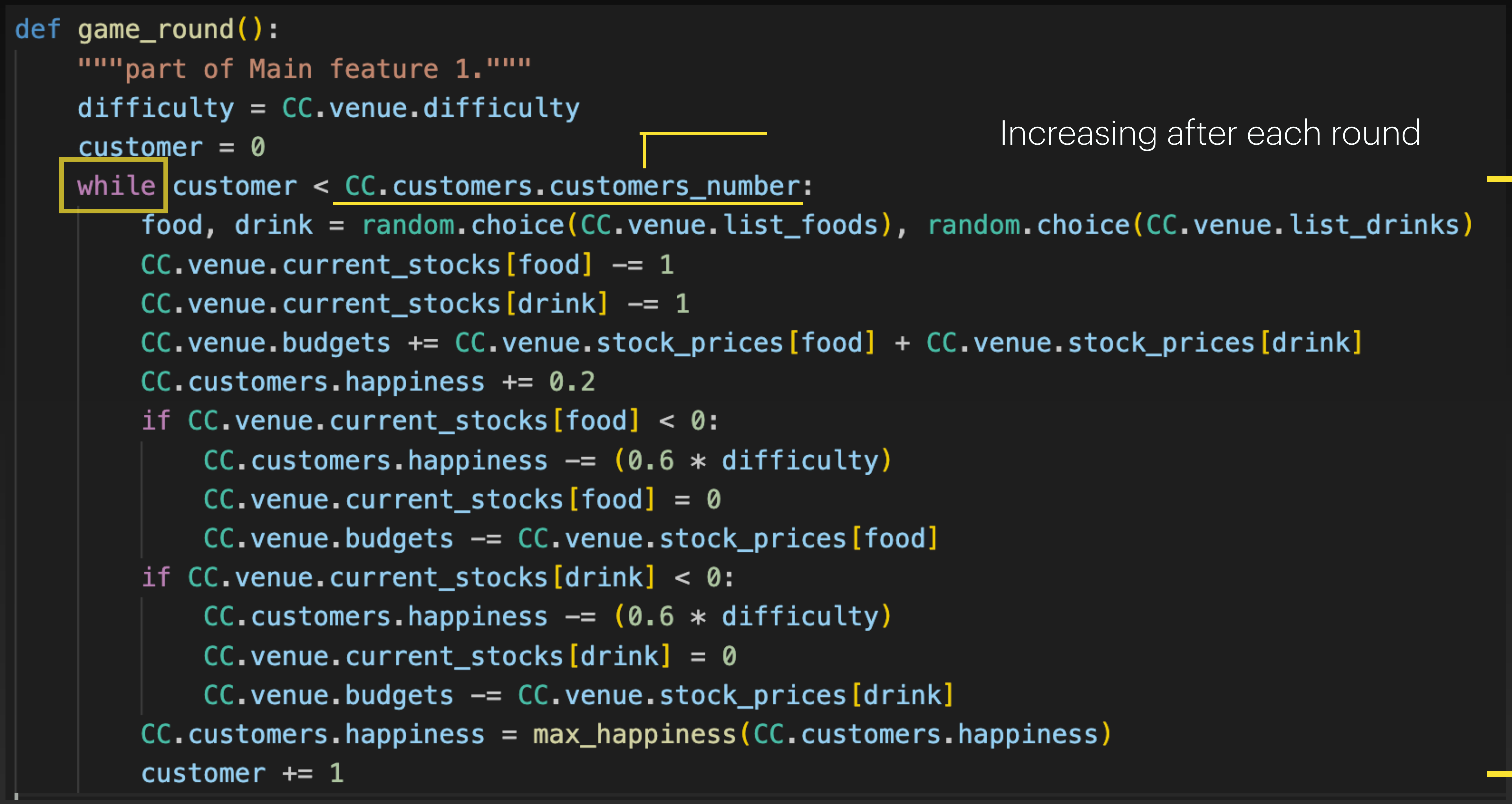
Walk - Through

Main Feature #1

Trading

Walk - Through

Main Feature #1



```
def game_round():
    """part of Main feature 1."""
    difficulty = CC.venue.difficulty
    customer = 0
    while customer < CC.customers.customers_number:
        food, drink = random.choice(CC.venue.list_foods), random.choice(CC.venue.list_drinks)
        CC.venue.current_stocks[food] -= 1
        CC.venue.current_stocks[drink] -= 1
        CC.venue.budgets += CC.venue.stock_prices[food] + CC.venue.stock_prices[drink]
        CC.customers.happiness += 0.2
        if CC.venue.current_stocks[food] < 0:
            CC.customers.happiness -= (0.6 * difficulty)
            CC.venue.current_stocks[food] = 0
            CC.venue.budgets -= CC.venue.stock_prices[food]
        if CC.venue.current_stocks[drink] < 0:
            CC.customers.happiness -= (0.6 * difficulty)
            CC.venue.current_stocks[drink] = 0
            CC.venue.budgets -= CC.venue.stock_prices[drink]
        CC.customers.happiness = max_happiness(CC.customers.happiness)
        customer += 1
```

Increasing after each round

While loop

Main part of this game

Starting from the first customer

Pick items, find them in price and stock dictionary

Each trading come with gain Happiness

- But if stock is under 0
- Money give back then lose Happiness

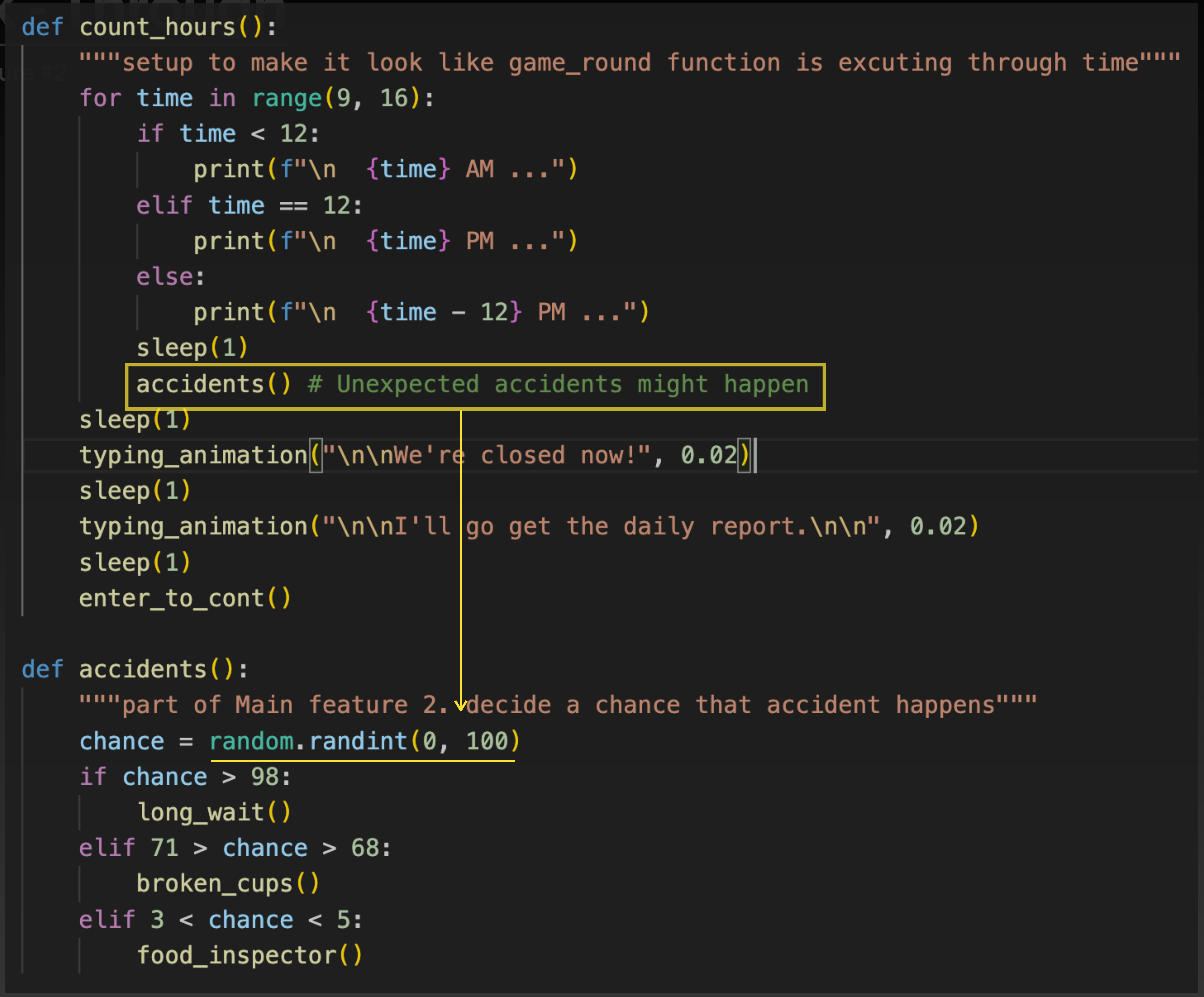
Walk - Through

Main Feature #2

Accidents

Walk Through

Main Featu



```python
def count_hours():
    """setup to make it look like game_round function is excuting through time"""
    for time in range(9, 16):
        if time < 12:
            print(f"\n  {time} AM ...")
        elif time == 12:
            print(f"\n  {time} PM ...")
        else:
            print(f"\n  {time - 12} PM ...")
        sleep(1)
        accidents() # Unexpected accidents might happen
    sleep(1)
    typing_animation("\n\nWe're closed now!", 0.02)
    sleep(1)
    typing_animation("\n\nI'll go get the daily report.\n\n", 0.02)
    sleep(1)
    enter_to_cont()

def accidents():
    """part of Main feature 2. decide a chance that accident happens"""
    chance = random.randint(0, 100)
    if chance > 98:
        long_wait()
    elif 71 > chance > 68:
        broken_cups()
    elif 3 < chance < 5:
        food_inspector()
```

Method of approach

Important part to understand this app

Programming Language - Series structure

Real world - Parallel structure

To make it look like Trading executes in real time

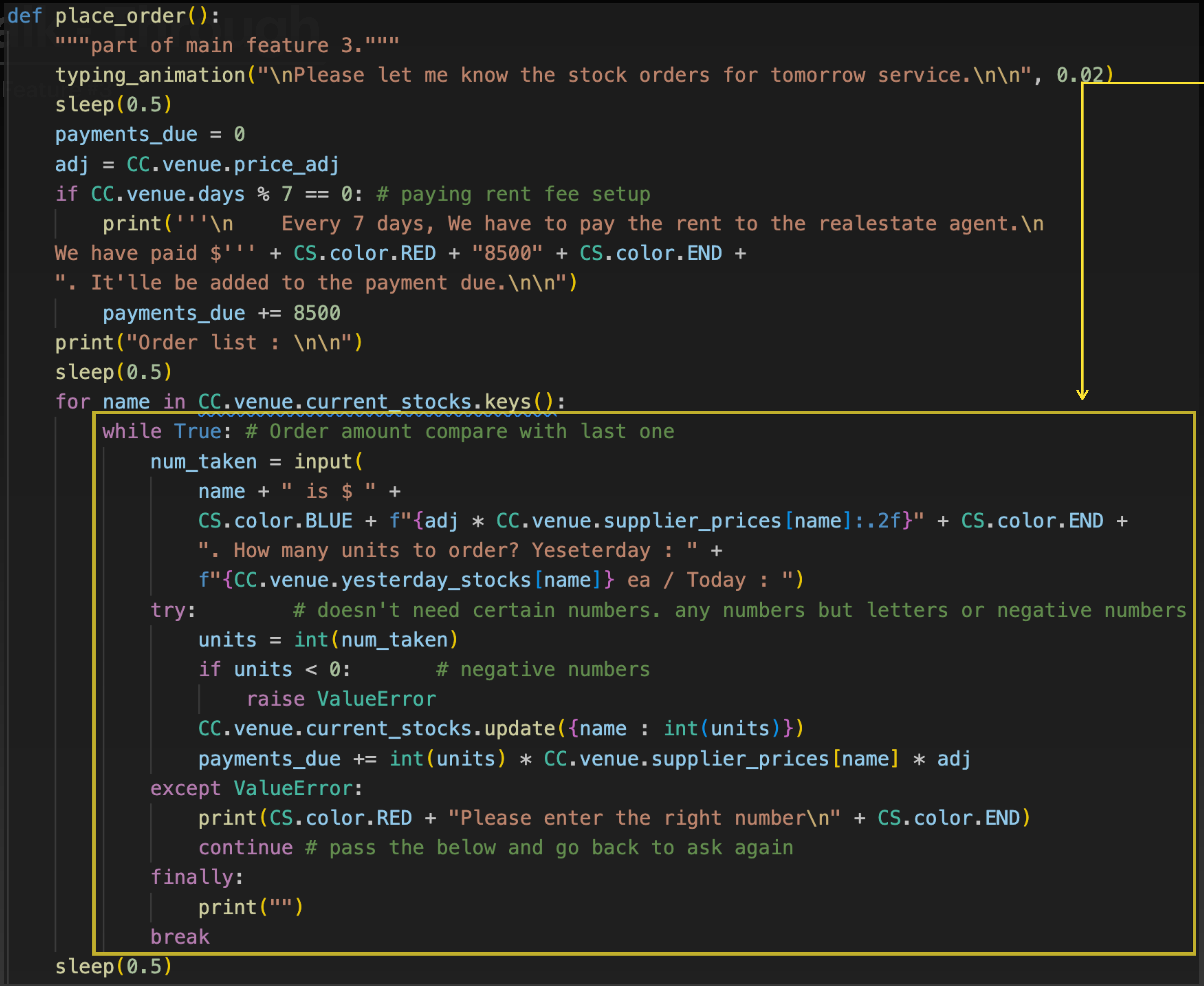
Although it’s already finished before we count

You can build it in Parallel way but *efficiency*

Walk - Through

Main Feature #3

Order Stocks



```python
def place_order():
    """part of main feature 3."""
    typing_animation("\nPlease let me know the stock orders for tomorrow service.\n\n", 0.02)
    sleep(0.5)
    payments_due = 0
    adj = CC.venue.price_adj
    if CC.venue.days % 7 == 0: # paying rent fee setup
        print('''\n    Every 7 days, We have to pay the rent to the realestate agent.\n
    We have paid $''' + CS.color.RED + "8500" + CS.color.END +
    ". It'lle be added to the payment due.\n\n")
        payments_due += 8500
    print("Order list : \n\n")
    sleep(0.5)
    for name in CC.venue.current_stocks.keys():
        while True: # Order amount compare with last one
            num_taken = input(
                name + " is $ " +
                CS.color.BLUE + f"{adj * CC.venue.supplier_prices[name]:.2f}" + CS.color.END +
                ". How many units to order? Yeseterday : " +
                f"{CC.venue.yesterday_stocks[name]} ea / Today : ")
            try:        # doesn't need certain numbers. any numbers but letters or negative numbers
                units = int(num_taken)
                if units < 0:       # negative numbers
                    raise ValueError
                CC.venue.current_stocks.update({name : int(units)})
                payments_due += int(units) * CC.venue.supplier_prices[name] * adj
            except ValueError:
                print(CS.color.RED + "Please enter the right number\n" + CS.color.END)
                continue # pass the below and go back to ask again
            finally:
                print("")
            break
    sleep(0.5)
```

2 ways of Error Handling

Depends on what value you need back from input

1. try/except statement

   Because we need lots of different numbers back

   Try casting the input to Integer

   If not possible which means ValueError occurs

   'continue' statement makes it ignore the rest of

   lines of codes until casting to Integer is possible

But what if I need certain numbers only back?

Walk - Through

Main Feature #3

2 ways of Error Handling

Depends on what value you need back from input

2. while false:

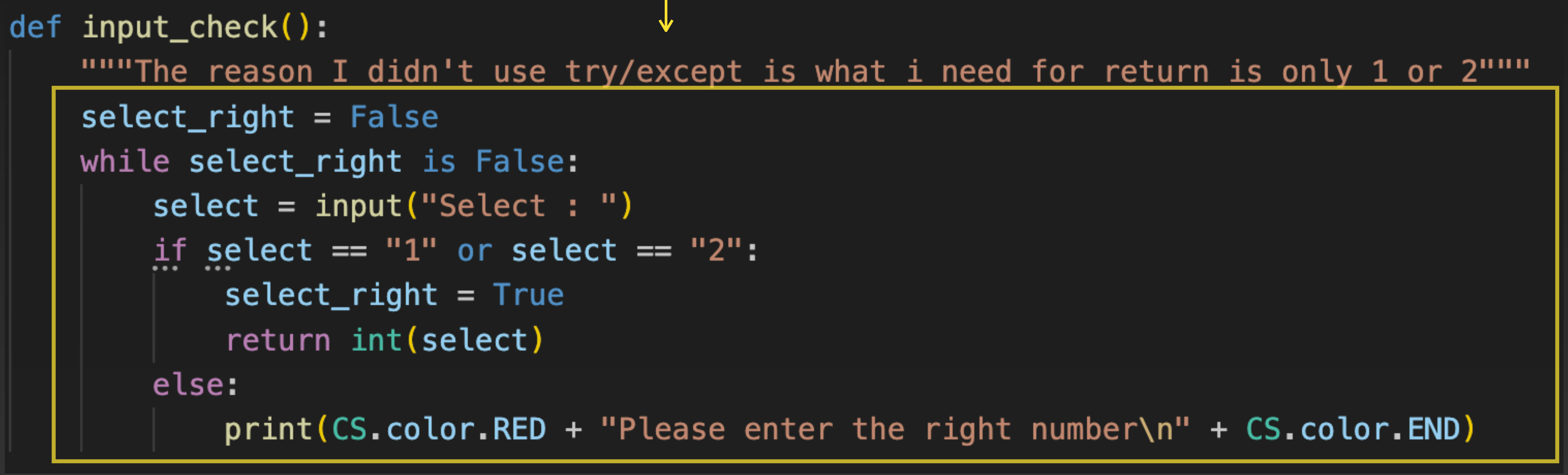
This case, we need only *Yes or No* answer.

Only if we receive what we want,

The loop will break by True

Anything else from input,

No matter what it is, We don’t need it



```
def input_check():
    """The reason I didn't use try/except is what i need for return is only 1 or 2"""
    select_right = False
    while select_right is False:
        select = input("Select : ")
        if select == "1" or select == "2":
            select_right = True
            return int(select)
        else:
            print(CS.color.RED + "Please enter the right number\n" + CS.color.END)
```

Demonstration

End