Task 2 – Example Answer with Python Code

*#Task 2*

**from** datetime **import** date

**import** math

**def** price\_contract(in\_dates, in\_prices, out\_dates, out\_prices, rate, storage\_cost\_rate, total\_vol, injection\_withdrawal\_cost\_rate):

volume **=** 0

buy\_cost **=** 0

cash\_in **=** 0

last\_date **=** min(min(in\_dates), min(out\_dates))

*# Ensure dates are in sequence*

all\_dates **=** sorted(set(in\_dates **+** out\_dates))

**for** i **in** range(len(all\_dates)):

*# processing code for each date*

start\_date **=** all\_dates[i]

**if** start\_date **in** in\_dates:

*# Inject on these dates and sum up cash flows*

**if** volume **<=** total\_vol **-** rate:

volume **+=** rate

*# Cost to purchase gas*

buy\_cost **+=** rate **\*** in\_prices[in\_dates**.**index(start\_date)]

*# Injection cost*

injection\_cost **=** rate **\*** injection\_withdrawal\_cost\_rate

buy\_cost **+=** injection\_cost

print('Injected gas on %s at a price of %s'**%**(start\_date, in\_prices[in\_dates**.**index(start\_date)]))

**else**:

*# We do not want to inject when rate is greater than total volume minus volume*

print('Injection is not possible on date %s as there is insufficient space in the storage facility'**%start\_date**)

**elif** start\_date **in** out\_dates:

*#Withdraw on these dates and sum cash flows*

**if** volume **>=** rate:

volume **-=** rate

cash\_in **+=** rate **\*** out\_prices[out\_dates**.**index(start\_date)]

*# Withdrawal cost*

withdrawal\_cost **=** rate **\*** injection\_withdrawal\_cost\_rate

cash\_in **-=** withdrawal\_cost

print('Extracted gas on %s at a price of %s'**%**(start\_date, out\_prices[out\_dates**.**index(start\_date)]))

**else**:

*# we cannot withdraw more gas than is actually stored*

print('Extraction is not possible on date %s as there is insufficient volume of gas stored'**%start\_date**)

store\_cost **=** math**.**ceil((max(out\_dates) **-** min(in\_dates))**.**days **//** 30) **\*** storage\_cost\_rate

**return** cash\_in **-** store\_cost **-** buy\_cost

*# Example usage of price\_contract()*

in\_dates **=** [date(2022, 1, 1), date(2022, 2, 1), date(2022, 2, 21), date(2022, 4, 1)] *#injection dates*

in\_prices **=** [20, 21, 20.5, 22]*#prices on the injection days*

out\_dates **=** [date(2022, 1, 27), date(2022, 2, 15), date(2022, 3, 20), date(2022, 6, 1)] *# extraction dates*

out\_prices **=** [23, 19, 21, 25] *# prices on the extraction days*

rate **=** 100000 *# rate of gas in cubic feet per day*

storage\_cost\_rate **=** 10000 *# total volume in cubic feet*

injection\_withdrawal\_cost\_rate **=** 0.0005 *# $/cf*

max\_storage\_volume **=** 500000 *# maximum storage capacity of the storage facility*

result **=** price\_contract(in\_dates, in\_prices, out\_dates, out\_prices, rate, storage\_cost\_rate, max\_storage\_volume, injection\_withdrawal\_cost\_rate)

print()

print(f"The value of the contract is: ${result}")

*## Explaining the Methodology Adopted for this Task ##*

*# The given Python code implements a function `price\_contract` that calculates the profit or loss obtained by*

*# undertaking trades on given dates for a contract involving the buying, storing, and selling of natural gas the*

*# storage cost of the gas, the injection/withdrawal. The value of the contract is the profit or loss obtained by*

*# undertaking the trades on given dates. Play around with the parameters and you'll be able to see this.*

*# In the end the intent for this function returns the value of the contract.*

*#The function takes in eight inputs:*

*#- `in\_dates`: A list of dates on which the gas is being injected into the storage facility.*

*#- `in\_prices`: A list of prices of gas on each of the injection dates.*

*#- `out\_dates`: A list of dates on which the gas is being withdrawn from the storage facility.*

*#- `out\_prices`: A list of prices of gas on each of the withdrawal dates.*

*#- `rate`: The rate of gas in cubic feet per day.*

*#- `storage\_cost\_rate`: A fixed monthly fee to store the gas*

*#- `total\_vol`: The total volume of gas in cubic feet that can be stored.*

*#- `injection\_withdrawal\_cost\_rate`: The injection/withdrawal cost of gas in dollars per cubic foot.*

*# The function first ensures that all the dates are in sequence and sorted in ascending order. Then, it iterates*

*#over all the dates and calculates the cash flows on each date. If the current date is an injection date, it*

*#injects gas into the storage facility and calculates the cost to store the gas, the cost to purchase the gas,*

*#and the injection cost. If the current date is a withdrawal date, it withdraws gas from the storage facility and*

*#calculates the cash inflow from selling the gas, the cost to store the remaining gas, and the withdrawal cost.*

*# Finally, the function returns the net profit or loss by subtracting the storage cost and the cost to purchase*

*#the gas from the cash inflow from selling the gas.*

*# The example usage of the `price\_contract` function calculates the profit or loss for a contract that involves*

*#injecting gas on four different dates and withdrawing gas on four different dates, each with a different price.*

*#The other inputs such as the rate of gas, the storage cost rate, the total volume, and the injection/withdrawal*

*#cost rate are also provided. The output is printed to the console using an f-string.*