Table of Contents

Question1: Portfolio Analysis Tool

- Objective: Develop a Python program that analyzes a stock portfolio.
  The program should include functions to perform the following tasks:

  (1)Input Portfolio: Allow the user to input a list of stock tickers and the number of shares they own.
  (2)Fetch Stock Prices: Retrieve current stock prices from an online source.
  (3)Calculate Portfolio Value: Compute the current value of the portfolio and the percentage allocation of each stock.

In [ ]:
```
stock_prices = {"AAPL": 150, "MSFT": 280, "GOOGL": 2600, "AMZN": 3300}

def input_portfolio():

    return portfolio


def calculate_portfolio_value(portfolio):

    return values, total_value


def main():
    portfolio = input_portfolio()
    values, total_value = calculate_portfolio_value(portfolio)
    print(f"Total Portfolio Value: ${total_value}")


if __name__ == "__main__":
    main()
```

```
Enter the number of different stocks in your portfolio: 2
Enter stock ticker: MSFT
Enter number of shares for MSFT: 10
Enter stock ticker: AAPL
Enter number of shares for AAPL: 1
Total Portfolio Value: $2950
```

In [ ]:
```
pip install yfinance
```

```
Defaulting to user installation because normal site-packages is not writeable
Collecting yfinance
  Downloading yfinance-0.2.32-py2.py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: pandas>=1.3.0 in c:\users\shubh.shubham_5260\appdata
\roaming\python\python310\site-packages (from yfinance) (2.1.0)
Requirement already satisfied: numpy>=1.16.5 in c:\users\shubh.shubham_5260\appdata
\roaming\python\python310\site-packages (from yfinance) (1.23.5)
Collecting requests>=2.31 (from yfinance)
  Downloading requests-2.31.0-py3-none-any.whl.metadata (4.6 kB)
Collecting multitasking>=0.0.7 (from yfinance)
  Downloading multitasking-0.0.11-py3-none-any.whl (8.5 kB)
Collecting lxml>=4.9.1 (from yfinance)
  Downloading lxml-4.9.3-cp310-cp310-win_amd64.whl.metadata (3.9 kB)
Collecting appdirs>=1.4.4 (from yfinance)
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Requirement already satisfied: pytz>=2022.5 in c:\users\shubh.shubham_5260\appdata\r
oaming\python\python310\site-packages (from yfinance) (2023.3.post1)
Collecting frozendict>=2.3.4 (from yfinance)
  Downloading frozendict-2.3.9-cp310-cp310-win_amd64.whl.metadata (21 kB)
Collecting peewee>=3.16.2 (from yfinance)
  Downloading peewee-3.17.0.tar.gz (2.9 MB)
     ---------------------------------------- 0.0/2.9 MB ? eta -:--:--
      --------------------------------------- 0.1/2.9 MB 2.0 MB/s eta 0:00:02
     --- ------------------------------------ 0.3/2.9 MB 3.0 MB/s eta 0:00:01
     ----- ---------------------------------- 0.4/2.9 MB 2.7 MB/s eta 0:00:01
     ------- -------------------------------- 0.6/2.9 MB 3.0 MB/s eta 0:00:01
     --------- ------------------------------ 0.7/2.9 MB 3.2 MB/s eta 0:00:01
     ------------ --------------------------- 0.9/2.9 MB 3.1 MB/s eta 0:00:01
     --------------- ------------------------ 1.1/2.9 MB 3.2 MB/s eta 0:00:01
     ---------------- ----------------------- 1.2/2.9 MB 3.1 MB/s eta 0:00:01
     ------------------ --------------------- 1.4/2.9 MB 3.4 MB/s eta 0:00:01
     --------------------- ------------------ 1.6/2.9 MB 3.5 MB/s eta 0:00:01
     ------------------------ --------------- 1.8/2.9 MB 3.6 MB/s eta 0:00:01
     ------------------------- -------------- 2.0/2.9 MB 3.7 MB/s eta 0:00:01
     ---------------------------- ---------- 2.2/2.9 MB 3.8 MB/s eta 0:00:01
     ------------------------------- ------- 2.4/2.9 MB 3.9 MB/s eta 0:00:01
     ---------------------------------- --- 2.7/2.9 MB 4.0 MB/s eta 0:00:01
     ----------------------------------- - 2.8/2.9 MB 3.9 MB/s eta 0:00:01
     ---------------------------------------- 2.9/2.9 MB 3.9 MB/s eta 0:00:00
  Installing build dependencies: started
  Installing build dependencies: finished with status 'done'
  Getting requirements to build wheel: started
  Getting requirements to build wheel: finished with status 'done'
  Preparing metadata (pyproject.toml): started
  Preparing metadata (pyproject.toml): finished with status 'done'
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\program files\python310
\lib\site-packages (from yfinance) (4.12.2)
Collecting html5lib>=1.1 (from yfinance)
  Downloading html5lib-1.1-py2.py3-none-any.whl (112 kB)
     ---------------------------------------- 0.0/112.2 kB ? eta -:--:--
     ---------------------------------------- 112.2/112.2 kB 6.4 MB/s eta 0:00:00
Requirement already satisfied: soupsieve>1.2 in c:\program files\python310\lib\site-
packages (from beautifulsoup4>=4.11.1->yfinance) (2.4.1)
Requirement already satisfied: six>=1.9 in c:\program files\python310\lib\site-packa
ges (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in c:\program files\python310\lib\site-p
```

```
ackages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.2 in c:\program files\python310
\lib\site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: tzdata>=2022.1 in c:\users\shubh.shubham_5260\appdata
\roaming\python\python310\site-packages (from pandas>=1.3.0->yfinance) (2023.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\shubh.shubham_52
60\appdata\roaming\python\python310\site-packages (from requests>=2.31->yfinance)
(3.1.0)
Requirement already satisfied: idna<4,>=2.5 in c:\program files\python310\lib\site-p
ackages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\shubh.shubham_5260\app
data\roaming\python\python310\site-packages (from requests>=2.31->yfinance) (1.26.1
5)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\shubh.shubham_5260\app
data\roaming\python\python310\site-packages (from requests>=2.31->yfinance) (2022.1
2.7)
Downloading yfinance-0.2.32-py2.py3-none-any.whl (68 kB)
   ---------------------------------------- 0.0/69.0 kB ? eta -:--:--
   ---------------------------------------- 69.0/69.0 kB 1.9 MB/s eta 0:00:00
Downloading frozendict-2.3.9-cp310-cp310-win_amd64.whl (35 kB)
Downloading lxml-4.9.3-cp310-cp310-win_amd64.whl (3.8 MB)
   ---------------------------------------- 0.0/3.8 MB ? eta -:--:--
   -- ------------------------------------- 0.2/3.8 MB 12.2 MB/s eta 0:00:01
   --- ------------------------------------ 0.3/3.8 MB 4.7 MB/s eta 0:00:01
   ----- ---------------------------------- 0.5/3.8 MB 4.7 MB/s eta 0:00:01
   ------- -------------------------------- 0.7/3.8 MB 4.7 MB/s eta 0:00:01
   --------- ------------------------------ 0.9/3.8 MB 4.5 MB/s eta 0:00:01
   ------------ --------------------------- 1.2/3.8 MB 5.0 MB/s eta 0:00:01
   --------------- ------------------------ 1.5/3.8 MB 5.0 MB/s eta 0:00:01
   ------------------ --------------------- 1.7/3.8 MB 4.8 MB/s eta 0:00:01
   -------------------- ------------------- 1.9/3.8 MB 4.8 MB/s eta 0:00:01
   ----------------------- ---------------- 2.2/3.8 MB 5.0 MB/s eta 0:00:01
   ------------------------- -------------- 2.5/3.8 MB 5.1 MB/s eta 0:00:01
   ---------------------------- ---------- 2.8/3.8 MB 5.3 MB/s eta 0:00:01
   ---------------------------------- ------ 3.1/3.8 MB 5.4 MB/s eta 0:00:01
   ------------------------------------ --- 3.4/3.8 MB 5.5 MB/s eta 0:00:01
   ---------------------------------------  3.8/3.8 MB 5.6 MB/s eta 0:00:01
   ---------------------------------------- 3.8/3.8 MB 5.4 MB/s eta 0:00:00
Downloading requests-2.31.0-py3-none-any.whl (62 kB)
   ---------------------------------------- 0.0/62.6 kB ? eta -:--:--
   ---------------------------------------- 62.6/62.6 kB 3.5 MB/s eta 0:00:00
Building wheels for collected packages: peewee
  Building wheel for peewee (pyproject.toml): started
  Building wheel for peewee (pyproject.toml): finished with status 'done'
  Created wheel for peewee: filename=peewee-3.17.0-py3-none-any.whl size=135766 sha2
56=c585398fd600fd6544d78f48c490df765747b14882084484ee4011cba7d006a8
  Stored in directory: c:\users\shubh.shubham_5260\appdata\local\pip\cache\wheels\c7
\70\ad\212867e96e7004265a69c4aa5dcff00a95f547a67ba26e7e76
Successfully built peewee
Installing collected packages: peewee, multitasking, appdirs, requests, lxml, html5l
ib, frozendict, yfinance
  Attempting uninstall: requests
    Found existing installation: requests 2.29.0
    Uninstalling requests-2.29.0:
      Successfully uninstalled requests-2.29.0
Successfully installed appdirs-1.4.4 frozendict-2.3.9 html5lib-1.1 lxml-4.9.3 multit
```

```
asking-0.0.11 peewee-3.17.0 requests-2.31.0 yfinance-0.2.32
Note: you may need to restart the kernel to use updated packages.
```

In [ ]:

```python
import yfinance as yf

def fetch_stock_prices(tickers):
    stock_prices = {}

    for ticker in tickers:
        try:
            stock_data = yf.Ticker(ticker)
            latest_price = stock_data.history(period='1d')['Close'].iloc[-1]
            stock_prices[ticker] = latest_price
        except:
            print(f"Error fetching data for {ticker}. Setting the price to 0.")
            stock_prices[ticker] = 0

    return stock_prices

def input_portfolio():
    portfolio = {}
    num_stocks = int(input("Enter the number of stocks in your portfolio: "))

    for _ in range(num_stocks):
        ticker = input("Enter stock ticker: ").upper()
        shares = float(input(f"Enter the number of shares for {ticker}: "))
        portfolio[ticker] = shares

    return portfolio

def calculate_portfolio_value(portfolio, stock_prices):
    values = {}
    total_value = 0

    for ticker, shares in portfolio.items():
        if ticker in stock_prices:
            value = shares * stock_prices[ticker]
            values[ticker] = value
            total_value += value

    return values, total_value

def main():
    portfolio = input_portfolio()
    stock_prices = fetch_stock_prices(portfolio.keys())
    values, total_value = calculate_portfolio_value(portfolio, stock_prices)

    print("\nStock Holdings:")
    for ticker, value in values.items():
        print(f"{ticker}: {value}")

    print(f"\nTotal Portfolio Value: ${total_value}")

if __name__ == "__main__":
    main()
```

```
Stock Holdings:
MSFT: 3774.2999267578125
AAPL: 189.97000122070312

Total Portfolio Value: $3964.2699279785156
```

Question 2: Risk and Return Analysis of Stocks

- Objective: Develop a Python program to calculate risk and return metrics for a set of stocks using basic Python functions and data structures. The tasks include:
  (1) Input Stock Return Data: Manually input annual return data for each stock.
  (2) Calculate Average Annual Return: Compute the mean return for each stock.
  (3) Calculate Standard Deviation: Determine the standard deviation of returns for each stock as a measure of risk.
  (4) Compute Sharpe Ratio: Calculate the Sharpe ratio for each stock using the risk-free rate.

In [ ]:
```python
def input_stock_data():

    return stocks

def calculate_mean_return(returns):
    pass


def calculate_std_deviation(returns):
    pass


def calculate_sharpe_ratio(returns, risk_free_rate):
    pass


def main():
    stocks = input_stock_data()
    risk_free_rate = float(input("Enter the risk-free rate: "))

    for ticker, returns in stocks.items():
        sharpe_ratio = calculate_sharpe_ratio(returns, risk_free_rate)
        print(f"Sharpe Ratio for {ticker}: {sharpe_ratio:.2f}")

if __name__ == "__main__":
    main()
```

```
Enter the number of stocks: 2
Enter stock ticker: aapl
Enter annual returns for AAPL separated by space: 10 20 30 40
Enter stock ticker: amzn
Enter annual returns for AMZN separated by space: 2 3 4 5
Enter the risk-free rate: 0.3
Sharpe Ratio for AAPL: 2.21
Sharpe Ratio for AMZN: 2.86
```

In [ ]:

In [ ]:
```python
import numpy as np

def input_stock_data():
    stocks = {}
    num_stocks = int(input("Enter the number of stocks: "))

    for _ in range(num_stocks):
        ticker = input("Enter stock ticker: ").upper()
        returns = list(map(float, input(f"Enter annual returns for {ticker} separat
        stocks[ticker] = returns

    return stocks

def calculate_mean_return(returns):
    return np.mean(returns)

def calculate_std_deviation(returns):
    return np.std(returns)

def calculate_sharpe_ratio(returns, risk_free_rate):
    mean_return = calculate_mean_return(returns)
    std_deviation = calculate_std_deviation(returns)

    # Assuming annual risk-free rate is provided
    sharpe_ratio = (mean_return - risk_free_rate) / std_deviation
    return sharpe_ratio

def main():
    stocks = input_stock_data()
    risk_free_rate = float(input("Enter the risk-free rate: "))

    for ticker, returns in stocks.items():
        sharpe_ratio = calculate_sharpe_ratio(returns, risk_free_rate)
        print(f"Sharpe Ratio for {ticker}: {sharpe_ratio:.2f}")

if __name__ == "__main__":
    main()
```

```
Sharpe Ratio for AAPL: 2.21
Sharpe Ratio for AMZN: 2.86
```

Question 3: Loan Amortization Schedule Generator

- Objective: Develop a Python program that generates a loan amortization schedule. The program should include functions to perform the following tasks:

  (1) Input Loan Details: Allow the user to input the loan amount, annual interest rate, and loan term (in years).
  (2) Calculate Monthly Payments: Implement a function to calculate monthly payments using the standard formula.
  (3) Generate Amortization Schedule: Create a schedule showing the breakdown of each monthly payment into principal and interest, along with the remaining loan balance.
  (4) Display the Schedule: Neatly display the amortization schedule in a tabular format.

- The formula for the monthly payment calculation is:

$$M = P\frac{r(1+r)^n}{(1+r)^n - 1}$$

  where
  M is the monthly payment
  P is the principal loan amount
  r is the monthly interest rate (annual rate divided by 12)
  n is the number of payments (loan term in years multiplied by 12) \

In [ ]:
```
def input_loan_details():
    pass

def calculate_monthly_payment(amount, annual_rate, years):
    pass

def generate_amortization_schedule(amount, annual_rate, years):
    pass

def display_schedule(schedule):
    pass

def main():
    amount, annual_rate, years = input_loan_details()
    schedule = generate_amortization_schedule(amount, annual_rate, years)
    display_schedule(schedule)

if __name__ == "__main__":
    main()
```

```
Enter loan amount: 100
Enter annual interest rate (as a percent): 0.2
Enter loan term in years: 10

Amortization Schedule:
Pmt No.   Payment        Principal      Interest       Balance
1         0.84           0.83           0.02           99.17
2         0.84           0.83           0.02           98.35
3         0.84           0.83           0.02           97.52
4         0.84           0.83           0.02           96.70
5         0.84           0.83           0.02           95.87
6         0.84           0.83           0.02           95.05
7         0.84           0.83           0.02           94.22
8         0.84           0.83           0.02           93.40
9         0.84           0.83           0.02           92.57
10        0.84           0.83           0.02           91.74
11        0.84           0.83           0.02           90.92
12        0.84           0.83           0.02           90.09
13        0.84           0.83           0.02           89.26
14        0.84           0.83           0.01           88.44
15        0.84           0.83           0.01           87.61
16        0.84           0.83           0.01           86.78
17        0.84           0.83           0.01           85.95
18        0.84           0.83           0.01           85.13
19        0.84           0.83           0.01           84.30
20        0.84           0.83           0.01           83.47
21        0.84           0.83           0.01           82.64
22        0.84           0.83           0.01           81.82
23        0.84           0.83           0.01           80.99
24        0.84           0.83           0.01           80.16
25        0.84           0.83           0.01           79.33
26        0.84           0.83           0.01           78.50
27        0.84           0.83           0.01           77.67
28        0.84           0.83           0.01           76.85
29        0.84           0.83           0.01           76.02
30        0.84           0.83           0.01           75.19
31        0.84           0.83           0.01           74.36
32        0.84           0.83           0.01           73.53
33        0.84           0.83           0.01           72.70
34        0.84           0.83           0.01           71.87
35        0.84           0.83           0.01           71.04
36        0.84           0.83           0.01           70.21
37        0.84           0.83           0.01           69.38
38        0.84           0.83           0.01           68.55
39        0.84           0.83           0.01           67.72
40        0.84           0.83           0.01           66.89
41        0.84           0.83           0.01           66.06
42        0.84           0.83           0.01           65.23
43        0.84           0.83           0.01           64.40
44        0.84           0.83           0.01           63.57
45        0.84           0.83           0.01           62.73
46        0.84           0.83           0.01           61.90
47        0.84           0.83           0.01           61.07
48        0.84           0.83           0.01           60.24
49        0.84           0.83           0.01           59.41
50        0.84           0.83           0.01           58.58
```

51	0.84	0.83	0.01	57.74
52	0.84	0.83	0.01	56.91
53	0.84	0.83	0.01	56.08
54	0.84	0.83	0.01	55.25
55	0.84	0.83	0.01	54.41
56	0.84	0.83	0.01	53.58
57	0.84	0.83	0.01	52.75
58	0.84	0.83	0.01	51.92
59	0.84	0.83	0.01	51.08
60	0.84	0.83	0.01	50.25
61	0.84	0.83	0.01	49.42
62	0.84	0.83	0.01	48.58
63	0.84	0.83	0.01	47.75
64	0.84	0.83	0.01	46.92
65	0.84	0.83	0.01	46.08
66	0.84	0.83	0.01	45.25
67	0.84	0.83	0.01	44.41
68	0.84	0.83	0.01	43.58
69	0.84	0.83	0.01	42.74
70	0.84	0.83	0.01	41.91
71	0.84	0.83	0.01	41.08
72	0.84	0.83	0.01	40.24
73	0.84	0.84	0.01	39.41
74	0.84	0.84	0.01	38.57
75	0.84	0.84	0.01	37.73
76	0.84	0.84	0.01	36.90
77	0.84	0.84	0.01	36.06
78	0.84	0.84	0.01	35.23
79	0.84	0.84	0.01	34.39
80	0.84	0.84	0.01	33.56
81	0.84	0.84	0.01	32.72
82	0.84	0.84	0.01	31.88
83	0.84	0.84	0.01	31.05
84	0.84	0.84	0.01	30.21
85	0.84	0.84	0.01	29.37
86	0.84	0.84	0.00	28.54
87	0.84	0.84	0.00	27.70
88	0.84	0.84	0.00	26.86
89	0.84	0.84	0.00	26.03
90	0.84	0.84	0.00	25.19
91	0.84	0.84	0.00	24.35
92	0.84	0.84	0.00	23.51
93	0.84	0.84	0.00	22.67
94	0.84	0.84	0.00	21.84
95	0.84	0.84	0.00	21.00
96	0.84	0.84	0.00	20.16
97	0.84	0.84	0.00	19.32
98	0.84	0.84	0.00	18.48
99	0.84	0.84	0.00	17.64
100	0.84	0.84	0.00	16.81
101	0.84	0.84	0.00	15.97
102	0.84	0.84	0.00	15.13
103	0.84	0.84	0.00	14.29
104	0.84	0.84	0.00	13.45
105	0.84	0.84	0.00	12.61
106	0.84	0.84	0.00	11.77

107	0.84	0.84	0.00	10.93
108	0.84	0.84	0.00	10.09
109	0.84	0.84	0.00	9.25
110	0.84	0.84	0.00	8.41
111	0.84	0.84	0.00	7.57
112	0.84	0.84	0.00	6.73
113	0.84	0.84	0.00	5.89
114	0.84	0.84	0.00	5.05
115	0.84	0.84	0.00	4.21
116	0.84	0.84	0.00	3.37
117	0.84	0.84	0.00	2.52
118	0.84	0.84	0.00	1.68
119	0.84	0.84	0.00	0.84
120	0.84	0.84	0.00	-0.00

In [ ]:
```
def input_loan_details():
    amount = float(input("Enter loan amount: "))
    annual_rate = float(input("Enter annual interest rate (as a percent): ")) / 100
    years = int(input("Enter loan term in years: "))
    return amount, annual_rate, years

def calculate_monthly_payment(amount, annual_rate, years):
    n = years * 12
    r = annual_rate / 12.0 / 100.0
    monthly_payment = amount * (r * (1 + r)**n) / ((1 + r)**n - 1)
    return monthly_payment

def generate_amortization_schedule(amount, annual_rate, years):
    schedule = []
    monthly_payment = calculate_monthly_payment(amount, annual_rate, years)
    balance = amount

    for month in range(1, years * 12 + 1):
        interest = balance * annual_rate / 12.0 / 100.0
        principal = monthly_payment - interest
        balance -= principal

        schedule.append([month, round(monthly_payment, 2), round(principal, 2), rou

    return schedule

def display_schedule(schedule):
    # Display the amortization schedule in a tabular format
    print("\nAmortization Schedule:")
    print("{:<10} {:<15} {:<15} {:<15} {:<15}".format("Pmt No.", "Payment", "Princi

    for row in schedule:
        print("{:<10} {:<15} {:<15} {:<15} {:<15}".format(*row))

def main():
    amount, annual_rate, years = input_loan_details()

    print(f"Enter Loan Amount: {amount}")
    print(f"Enter Annual Interest Rate(as a percentage): {annual_rate*100}")
    print(f"Enter Loan Term in years: {years}")
```

```
    schedule = generate_amortization_schedule(amount, annual_rate, years)
    display_schedule(schedule)

if __name__ == "__main__":
    main()
```

```
Enter Loan Amount: 100.0
Enter Annual Interest Rate(as a percentage): 0.2
Enter Loan Term in years: 10

Amortization Schedule:
```

Pmt No.	Payment	Principal	Interest	Balance
1	0.83	0.83	0.0	99.17
2	0.83	0.83	0.0	98.33
3	0.83	0.83	0.0	97.5
4	0.83	0.83	0.0	96.67
5	0.83	0.83	0.0	95.83
6	0.83	0.83	0.0	95.0
7	0.83	0.83	0.0	94.17
8	0.83	0.83	0.0	93.33
9	0.83	0.83	0.0	92.5
10	0.83	0.83	0.0	91.67
11	0.83	0.83	0.0	90.83
12	0.83	0.83	0.0	90.0
13	0.83	0.83	0.0	89.17
14	0.83	0.83	0.0	88.33
15	0.83	0.83	0.0	87.5
16	0.83	0.83	0.0	86.67
17	0.83	0.83	0.0	85.83
18	0.83	0.83	0.0	85.0
19	0.83	0.83	0.0	84.17
20	0.83	0.83	0.0	83.33
21	0.83	0.83	0.0	82.5
22	0.83	0.83	0.0	81.67
23	0.83	0.83	0.0	80.83
24	0.83	0.83	0.0	80.0
25	0.83	0.83	0.0	79.17
26	0.83	0.83	0.0	78.34
27	0.83	0.83	0.0	77.5
28	0.83	0.83	0.0	76.67
29	0.83	0.83	0.0	75.84
30	0.83	0.83	0.0	75.0
31	0.83	0.83	0.0	74.17
32	0.83	0.83	0.0	73.34
33	0.83	0.83	0.0	72.5
34	0.83	0.83	0.0	71.67
35	0.83	0.83	0.0	70.84
36	0.83	0.83	0.0	70.0
37	0.83	0.83	0.0	69.17
38	0.83	0.83	0.0	68.34
39	0.83	0.83	0.0	67.5
40	0.83	0.83	0.0	66.67
41	0.83	0.83	0.0	65.84
42	0.83	0.83	0.0	65.0
43	0.83	0.83	0.0	64.17
44	0.83	0.83	0.0	63.34
45	0.83	0.83	0.0	62.5
46	0.83	0.83	0.0	61.67
47	0.83	0.83	0.0	60.84
48	0.83	0.83	0.0	60.0
49	0.83	0.83	0.0	59.17
50	0.83	0.83	0.0	58.34

51	0.83	0.83	0.0	57.5
52	0.83	0.83	0.0	56.67
53	0.83	0.83	0.0	55.84
54	0.83	0.83	0.0	55.0
55	0.83	0.83	0.0	54.17
56	0.83	0.83	0.0	53.34
57	0.83	0.83	0.0	52.5
58	0.83	0.83	0.0	51.67
59	0.83	0.83	0.0	50.84
60	0.83	0.83	0.0	50.0
61	0.83	0.83	0.0	49.17
62	0.83	0.83	0.0	48.34
63	0.83	0.83	0.0	47.5
64	0.83	0.83	0.0	46.67
65	0.83	0.83	0.0	45.84
66	0.83	0.83	0.0	45.0
67	0.83	0.83	0.0	44.17
68	0.83	0.83	0.0	43.34
69	0.83	0.83	0.0	42.5
70	0.83	0.83	0.0	41.67
71	0.83	0.83	0.0	40.84
72	0.83	0.83	0.0	40.0
73	0.83	0.83	0.0	39.17
74	0.83	0.83	0.0	38.34
75	0.83	0.83	0.0	37.5
76	0.83	0.83	0.0	36.67
77	0.83	0.83	0.0	35.84
78	0.83	0.83	0.0	35.0
79	0.83	0.83	0.0	34.17
80	0.83	0.83	0.0	33.34
81	0.83	0.83	0.0	32.5
82	0.83	0.83	0.0	31.67
83	0.83	0.83	0.0	30.84
84	0.83	0.83	0.0	30.0
85	0.83	0.83	0.0	29.17
86	0.83	0.83	0.0	28.34
87	0.83	0.83	0.0	27.5
88	0.83	0.83	0.0	26.67
89	0.83	0.83	0.0	25.84
90	0.83	0.83	0.0	25.0
91	0.83	0.83	0.0	24.17
92	0.83	0.83	0.0	23.34
93	0.83	0.83	0.0	22.5
94	0.83	0.83	0.0	21.67
95	0.83	0.83	0.0	20.83
96	0.83	0.83	0.0	20.0
97	0.83	0.83	0.0	19.17
98	0.83	0.83	0.0	18.33
99	0.83	0.83	0.0	17.5
100	0.83	0.83	0.0	16.67
101	0.83	0.83	0.0	15.83
102	0.83	0.83	0.0	15.0
103	0.83	0.83	0.0	14.17
104	0.83	0.83	0.0	13.33
105	0.83	0.83	0.0	12.5
106	0.83	0.83	0.0	11.67

107	0.83	0.83	0.0	10.83
108	0.83	0.83	0.0	10.0
109	0.83	0.83	0.0	9.17
110	0.83	0.83	0.0	8.33
111	0.83	0.83	0.0	7.5
112	0.83	0.83	0.0	6.67
113	0.83	0.83	0.0	5.83
114	0.83	0.83	0.0	5.0
115	0.83	0.83	0.0	4.17
116	0.83	0.83	0.0	3.33
117	0.83	0.83	0.0	2.5
118	0.83	0.83	0.0	1.67
119	0.83	0.83	0.0	0.83
120	0.83	0.83	0.0	0.0