Assignment 5 – Instructions

Python problems

Practical 5.1

Create a Python program that calculates the total score and final average score of a gymnast who took part in a competition. It should also display the highest and lowest scores as well. The scores of 5 judges (each score out of 10) are captured. To calculate the final average score, the lowest and highest scores are disregarded. The average of the remaining 3 scores is used to calculate the average score.

- Create a function **find_lowest**. This function compares the given score with the current lowest score and returns the minimum of the two.
- Create a function **find_highest**. This function compares the given score with the current highest score and returns the maximum of the two.
- Create a function **calc_score**. This function calculates the final average score by subtracting the lowest and highest scores from the total sum of scores and then dividing by 3.
- Create a function **input_scores**. This function prompts the user to input 5 scores between 0 and 10 and returns the sum of scores, lowest score, and highest score.
- Finally the main calls the **input_scores**() function to obtain the sum of scores, lowest score, and highest score. Calculates the final average score using the calc_score() function. Prints the total score, lowest score, highest score and final average score.

Submit your Python file(*.py) here on eFundi, under this assignment, and make sure you attach and submit your file successfully. Save file as ***StudentNumber_Prac5-1.py***

Output

```
Enter score 1: 8
Enter score 2: 23
Invalid input: Score must be between 0 and 10
Enter score 2: 7
Enter score 3: 2
Enter score 4: 0
Enter score 5: 6

Total Score: 23
Lowest Score: 0
Highest Score: 8
Final Average Score: 5.00
```

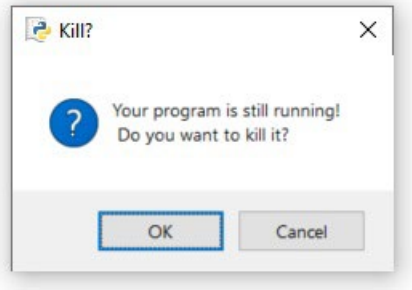
Practical 5.2

Create a Python program that simulates a simple bank management system for NWU Bank. The program allows users to perform various banking operations such as depositing money, withdrawing money, and checking their account balance.

- When you start the program, you will be prompted to enter a password that is divisible by 125. If the entered password does not meet this condition, you will be prompted to enter the password again.
- Upon successful login, you will be prompted to enter your name and initial balance, and then you will be ready to perform banking operations. See menu items in the image
- Create a function **deposit**. If you choose to deposit money, you will be prompted to enter the amount you wish to deposit.
- Create a function **withdraw**. If you choose to withdraw money, you will be prompted to enter the amount you wish to withdraw. If your account balance is insufficient, you will be notified.
- Create a function **check_balance**. If you choose to check your balance, your current balance will be displayed.

Submit your Python file(*.py) here on eFundi, under this assignment, and make sure you attach and submit your file successfully. Save file as ***StudentNumber_Prac5-2.py***

```
NWU Bank

Enter your password (Divisible by 125): 32
Enter your password (Divisible by 125): 1625
Login Successful!!

Enter your name: Peter
Enter initial balance: R5000
Account created successfully!

1. Deposit
2. Withdraw
3. Check Balance
4. Exit

Enter your choice: 1
Enter amount to deposit: R2500
Deposited R2500.0. Current balance: R7500.0

1. Deposit
2. Withdraw
3. Check Balance
4. Exit

Enter your choice: 2
Enter amount to withdraw: R4000
Withdrew R4000.0. Current balance: R3500.0

1. Deposit
2. Withdraw
3. Check Balance
4. Exit

Enter your choice: 3
Current balance for Peter: R3500.0

1. Deposit
2. Withdraw
3. Check Balance
4. Exit

Enter your choice: 4
Exiting...
```



	Rubric		
Prak 5-1	**Input Validation**	**2**	
	Inputs are validated to ensure they fall within the acceptable range (0 to 10 for scores).		
	Informative error messages to guide the user on valid inputs.		
	Calculation Accuracy	**3**	
	Functions for finding lowest and highest scores seem to work correctly.		
	The exclusion of lowest and highest scores in the final calculation.		
	Output Clarity	**2**	
	Output statements provide necessary information		
	Overall Logic and Correctness	**1**	
	There are logical errors in the program		
	Subtotal (8)		
Prak 5-1	**Input Validation**	**3**	
	Password validation is done but only for divisibility by 125		
	Inputs for deposit and withdrawal are validated for being numeric		
	Transaction Handling	**2**	
	Deposit and withdrawal functions work as intended.		
	The program correctly adjusts the balance based on deposits and withdrawals.		
	Output Clarity	**2**	
	Output messages are clear		
	More informative messages for errors like insufficient funds.		
	Overall Logic and Correctness	**2**	
	The logic for banking operations is generally correct.		
	Subtotal (9)		
	Code Structure and Readability	**3**	
	Both programs work as intended		
	Proper use of comments for explaining the purpose of each function and major code blocks.		
	Total	**20**	