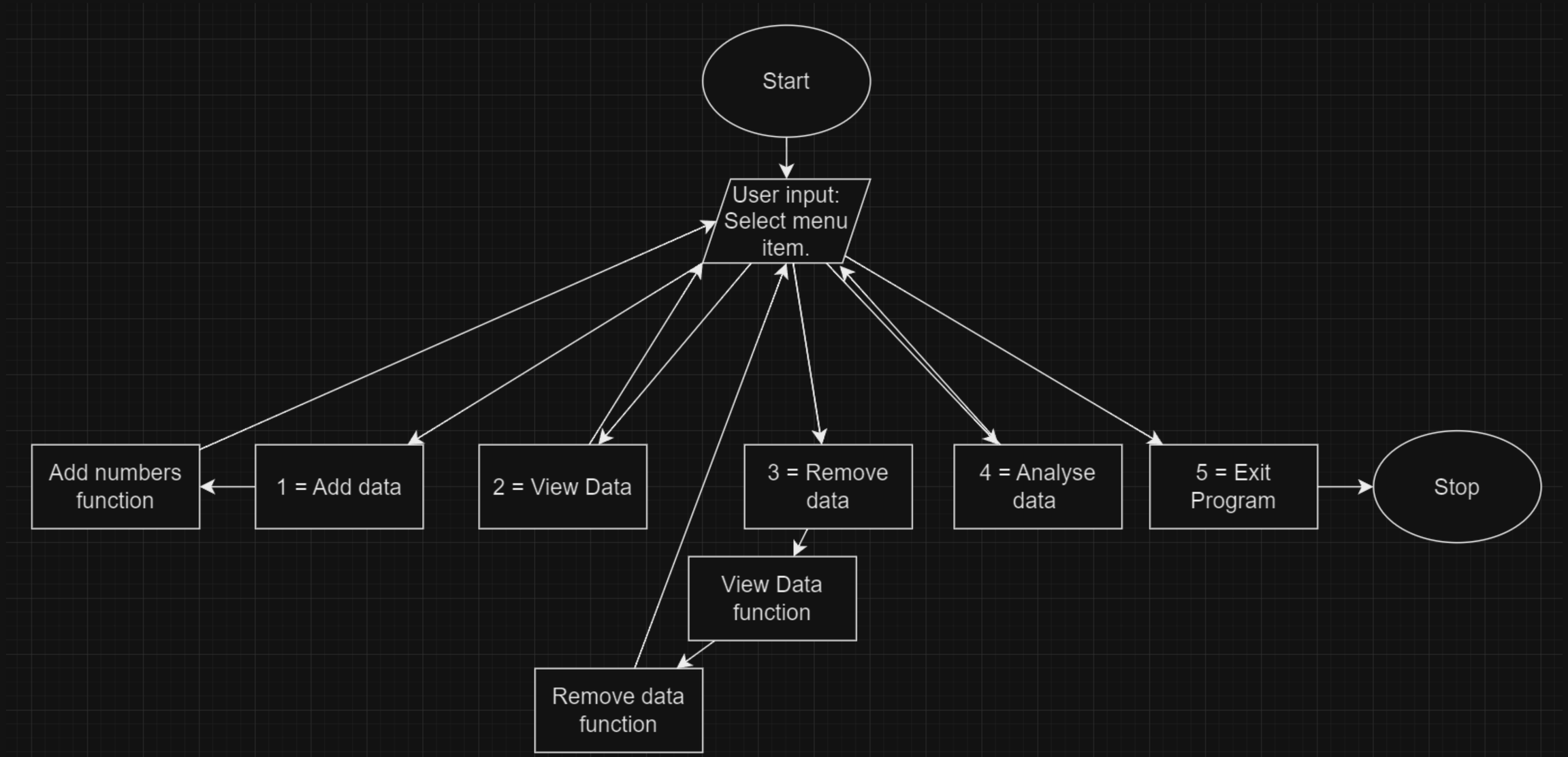
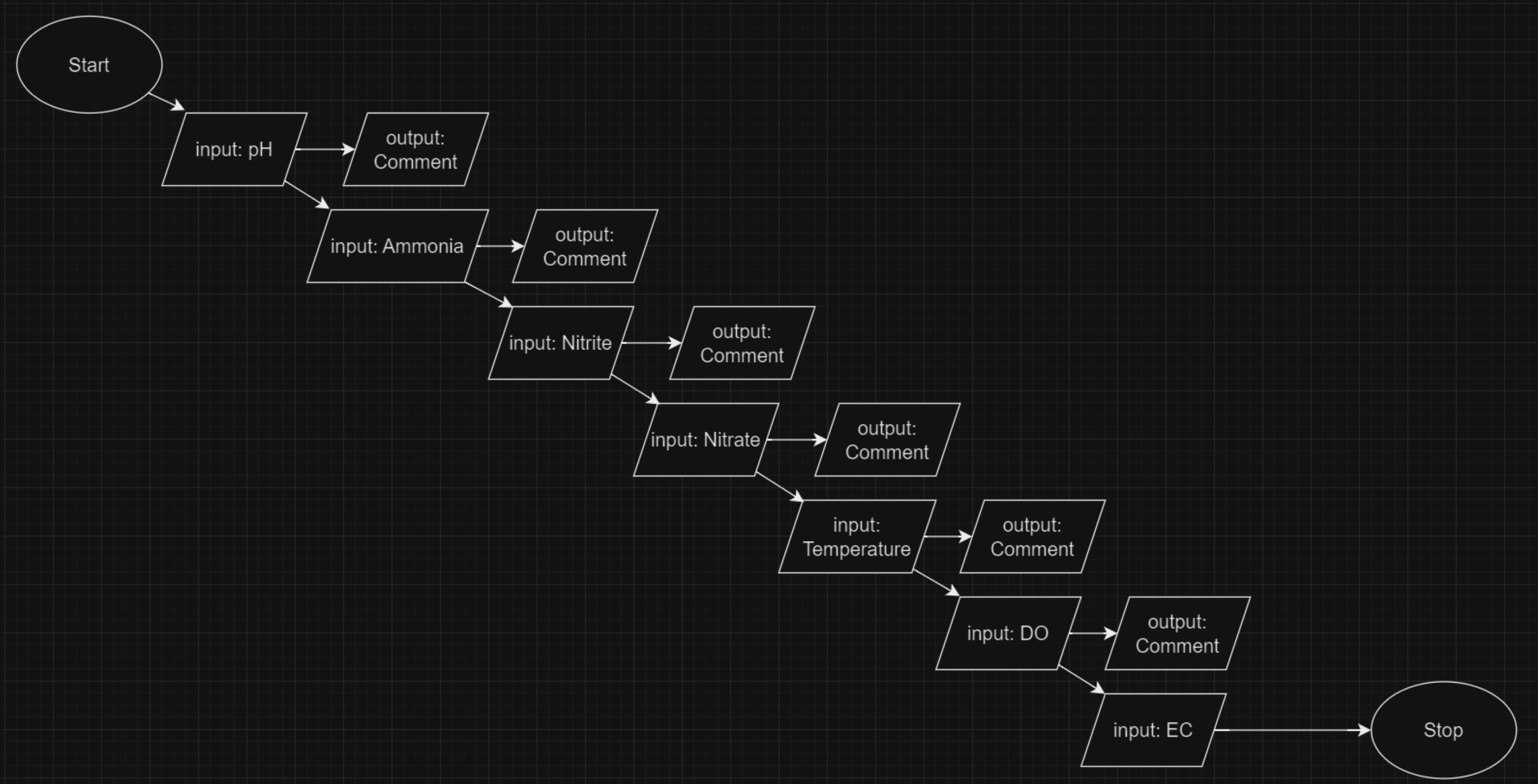
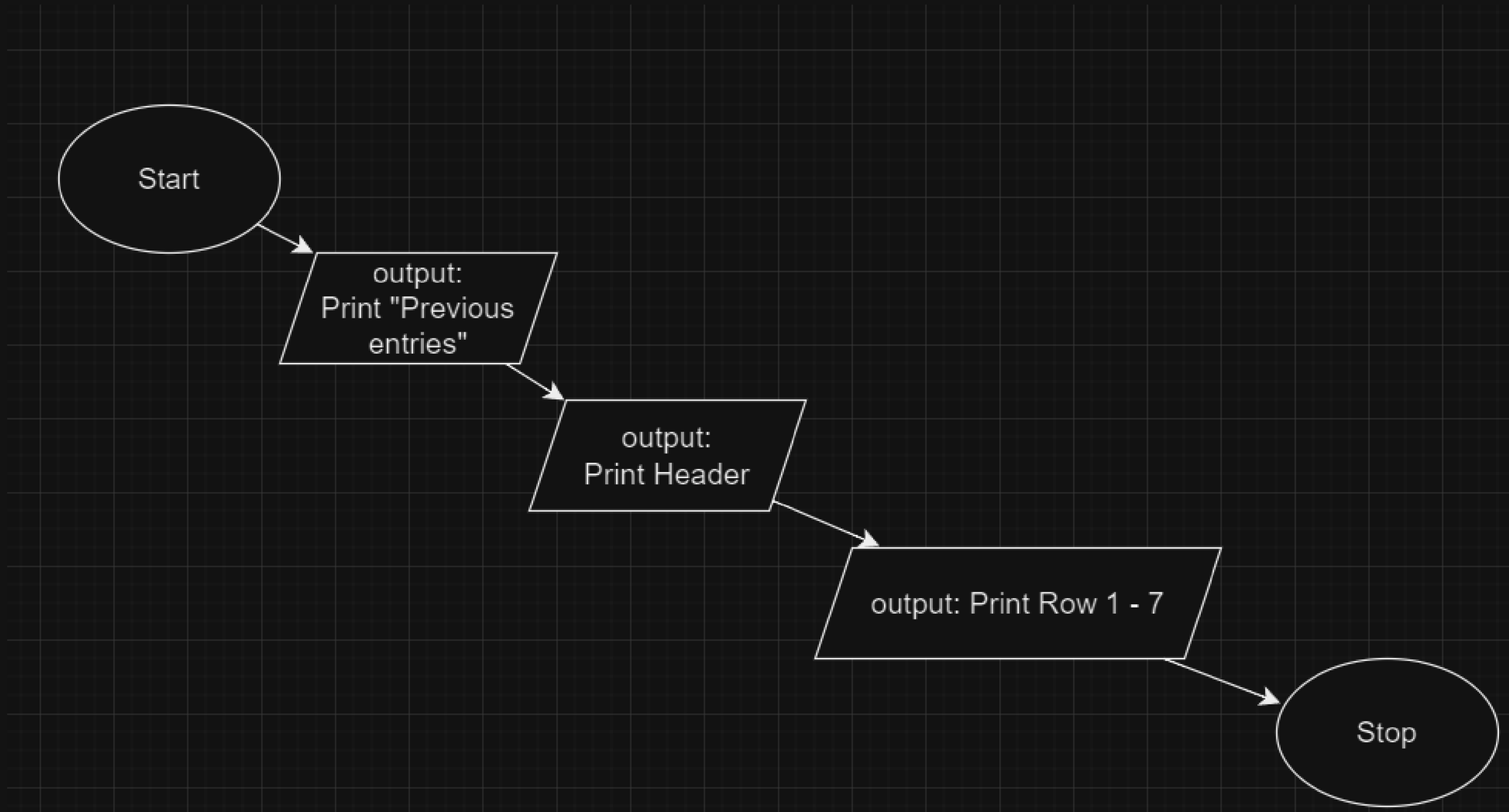
# Colin Hill
# T1A3


Start
User input: Select menu item.
Add numbers function
1 = Add data
2 = View Data
3 = Remove data
4 = Analyse data
5 = Exit Program
Stop
View Data function
Remove data function

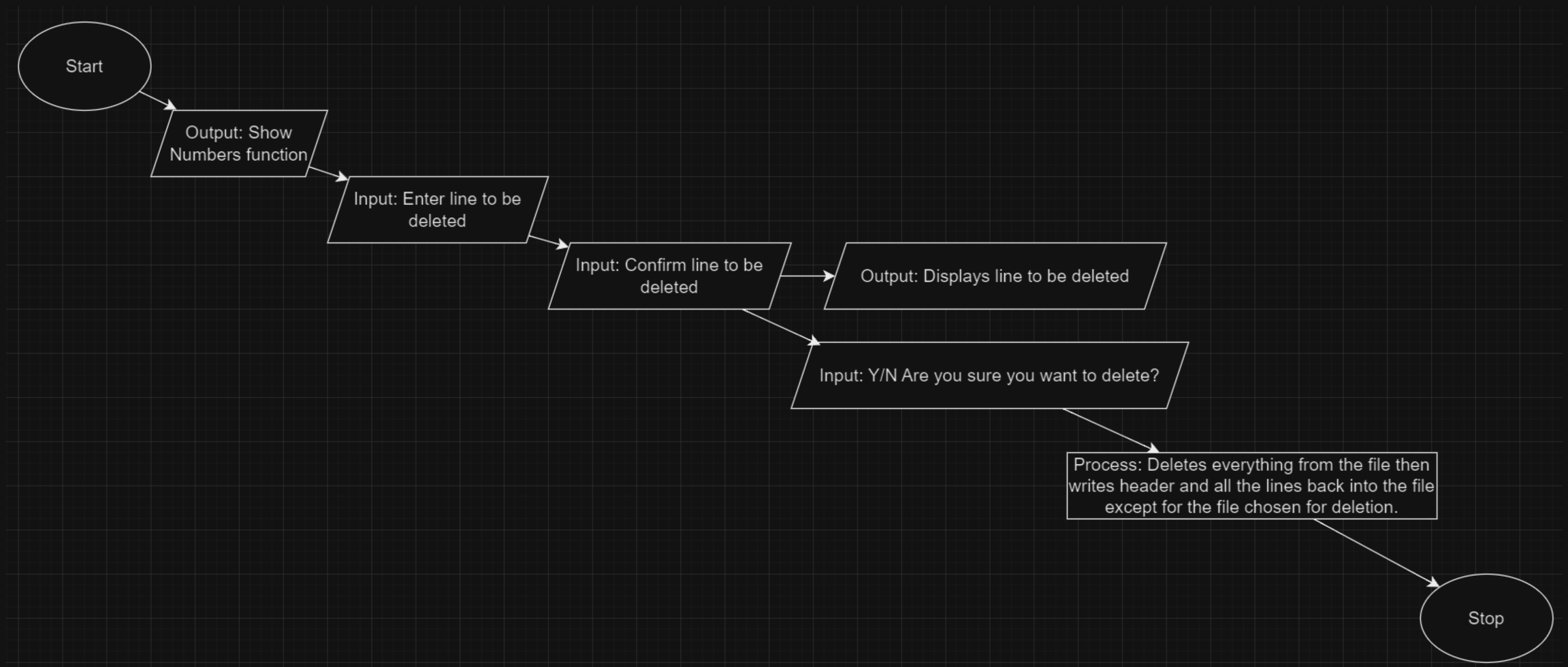
```python
print("\nWelcome to the Aquaponics data tracker")

def menu():
    print("\n1. Enter 1 to add today's data")
    print("2. Enter 2 to view previous data")
    print("3. Enter 3 to remove previous entries")
    print("4. Enter 4 to see analysis of stored data")
    print("5. Enter 5 to exit")
    choice = input("\nEnter your selection: ")
    return choice

user_choice = ""

while user_choice != "5":
    user_choice = menu()
    if (user_choice == "1"):
        add_numbers(file_name)
    elif (user_choice == "2"):
        show_numbers(file_name)
    elif (user_choice == "3"):
        show_numbers(file_name)
        line_number = int(input("Enter line number to remove (1-based): "))
        remove_numbers(file_name, line_number)
    elif (user_choice == "4"):
        if len(show_numbers(file_name)) > 1:
            analysis(file_name)
        else:
            print("No data available for analysis. Please add some entries first.")
    elif (user_choice == "5"):
        continue
    else:
        print("Invalid Input")

print("\nHave a nice day!")
```


Start
input: pH
output: Comment
input: Ammonia
output: Comment
input: Nitrite
output: Comment
input: Nitrate
output: Comment
input: Temperature
output: Comment
input: DO
output: Comment
input: EC
Stop

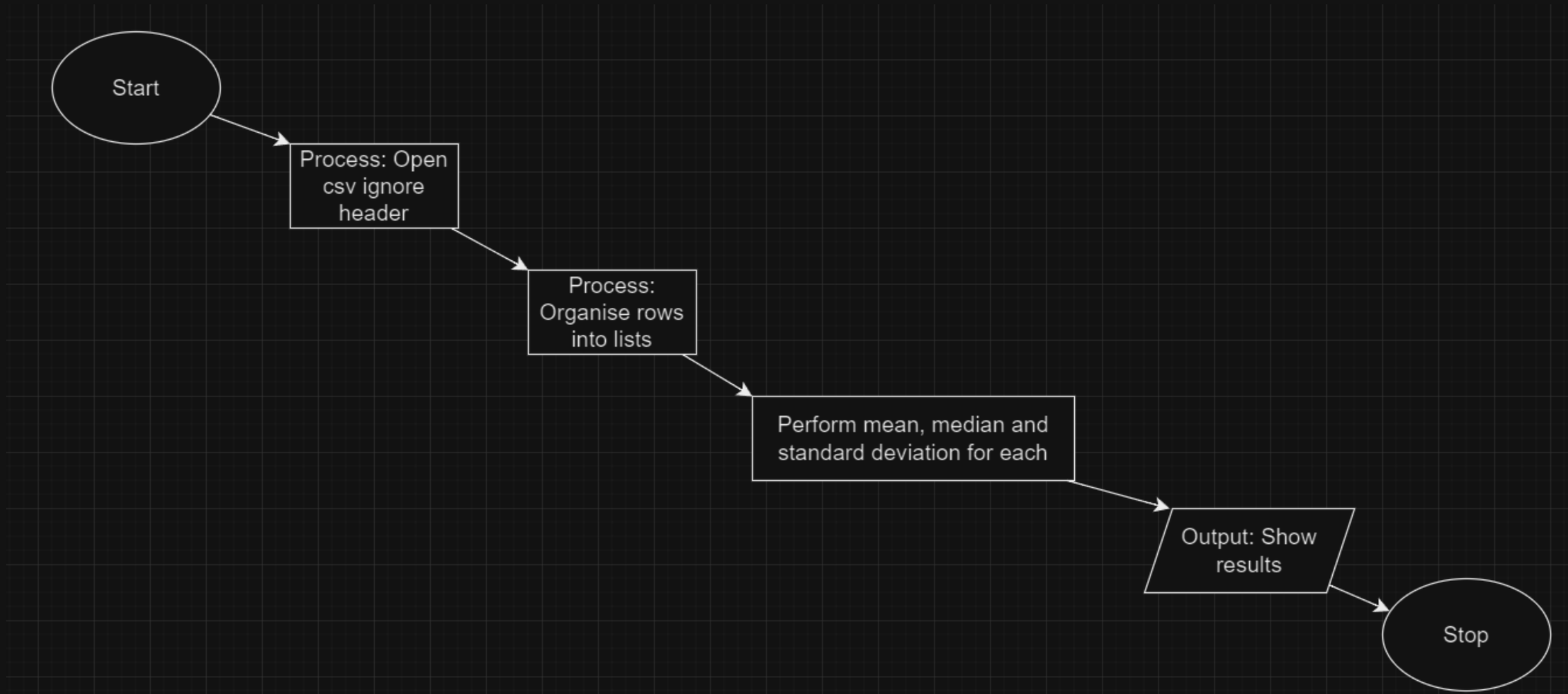
```python
def add_numbers(file_name):
    print("Add today's numbers")

    ph = get_user_input("Enter pH level: ", float, 0, 14)
    if ph < 6.6:
        print("Warning: Take Action: add lime to boost pH level")
    elif ph > 7:
        print("Warning: Nutrient lockout occurring")
        if ph > 8:
            print("Danger! Add acid to lower pH level")

    ammonia = get_user_input("Enter ammonia (ppm): ", float, 0, 5)
    if ammonia > 0.75:
        print("High ammonia reading! Check for dead fish.")
        if ammonia > 1:
            print("Stop feeding fish until ammonia is below 0.75ppm.")
        if ammonia > 2:
            print("Warning! Toxic ammonia level. Change 1/3 of water immediately.")

    nitrite = get_user_input("Enter nitrite (ppm): ", float, 0, 5)
    if nitrite > 0.75:
        print("High nitrite reading! You may have damaged bacteria.")
        if nitrite > 1:
            print("Warning! Toxic nitrite level. Change 1/3 of water immediately.")

    nitrate = get_user_input("Enter nitrate (ppm): ", float, 0, 160)
    if nitrate > 70:
        print("Consistently introduce more plants")
    if nitrate > 100:
        print("Consider adding another grow bed to reduce nitrate levels.")
    if nitrate > 150:
        print("Warning! Nitrate level toxic for fish. Take immediate action.")

    temperature = get_user_input("Enter water temperature (°C): ", float, 0, 100)
    if temperature < 0 or temperature > 49:
        print("Warning! Nitrifying bacteria and edible plants will die at temperatures below 0°C or above 49°C.")
    else:
        if temperature < 4:
            print("Warning! No bacterial activity will occur below 4°C.")
        else:
            if temperature < 10:
                print("The water is cold. Bacterial growth rate is decreased 75%.")
            elif temperature < 18:
                print("The water is cool. Bacterial growth rate is decreased 50%.")
            elif temperature < 30:
                print("The water is within the optimal temperature range for nitrifying bacteria.")
            else:
                print("The water is too hot and may harm bacteria.")

        if 18 <= temperature <= 25:
            print("The water temperature is optimal for plant growth.")
        else:
            print("The water temperature is not ideal for plant growth.")

    do = get_user_input("Enter dissolved oxygen (mg/L): ", float, 0, 20)
    if do < 5:
        print("Oxygen level low! Fish may die. Lower water temp or check aeration systems.")

    ec = get_user_input("Enter electrical conductivity (µS/cm): ", float, 0, 500)
    if ec == 0:
        print("Nutrient levels low!")

    create_csv(file_name, [ph, ammonia, nitrite, nitrate, temperature, do, ec])

    print("Data saved successfully!")
```


Start
output:
Print "Previous
entries"
output:
Print Header
output: Print Row 1 - 7
Stop

```python
def show_numbers(file_name):
    print("Previous entries")

    print("Date\t\tpH\tAmmonia\tNitrite\tNitrate\tTemperature\tDO\tEC")

    with open(file_name, "r") as csvfile:
        reader = csv.reader(csvfile)
        data = list(reader)[1:]

        for row in data:
            print(f"{row[0]}\t{row[1]}\t{row[2]}\t{row[3]}\t{row[4]}\t{row[5]}\t\t{row[6]}\t{row[7]}")

    return data
```


Start
Output: Show Numbers function
Input: Enter line to be deleted
Input: Confirm line to be deleted
Output: Displays line to be deleted
Input: Y/N Are you sure you want to delete?
Process: Deletes everything from the file then writes header and all the lines back into the file except for the file chosen for deletion.
Stop

```python
def remove_numbers(file_name, line_number):
    while True:
        try:
            line_number = int(input("\nConfirm line number to remove (1-based): "))

            with open(file_name, "r", newline="") as csvfile:
                reader = csv.reader(csvfile)
                rows = list(reader)[1:]

            if 1 <= line_number <= len(rows):
                entry = rows[line_number - 1]
                print("Entry retrieved:", entry)

                confirm = input("\nAre you sure you want to remove this entry? (y/N) ")
                if confirm.lower() == "y":
                    rows.pop(line_number - 1)

                    with open(file_name, "w", newline="") as csvfile:
                        writer = csv.writer(csvfile)
                        writer.writerow(["Date", "pH", "ammonia", "nitrite", "nitrate", "Temperature", "DO", "EC"])
                        writer.writerows(rows[1:])

                    print(f"\nEntry on line {line_number} successfully removed!")
                    break
                else:
                    print("\nEntry remains unchanged.")
                    break
            else:
                print(f"Invalid line number. Please enter a valid value between 1 and {len(rows)}")
        except ValueError:
            print("Invalid input. Please enter a valid integer.")

        exit_choice = input("\nDo you want to try removing another entry? (Y/N): ")
        if exit_choice.lower() != "y":
            print("\nExiting remove previous entries.")
            break
```


Start
Process: Open csv ignore header
Process: Organise rows into lists
Perform mean, median and standard deviation for each
Output: Show results
Stop

```python
def analysis(file_name):
    with open(file_name, "r") as csvfile:
        reader = csv.reader(csvfile)
        next(reader)
        data = list(reader)

        data = [[float(value) for value in row[1:]] for row in data]

        ph_vals, ammonia_vals, nitrite_vals, nitrate_vals, temp_vals, do_vals, ec_vals = zip(*data)

        print("Data Analysis:")

        print("pH:")
        print(f"\tMean: {mean(ph_vals):.2f}")
        print(f"\tMedian: {median(ph_vals):.2f}")
        print(f"\tStandard deviation: {stdev(ph_vals):.2f}")

        print("Ammonia:")
        print(f"\tMean: {mean(ammonia_vals):.2f}")
        print(f"\tMedian: {median(ammonia_vals):.2f}")
        print(f"\tStandard deviation: {stdev(ammonia_vals):.2f}")

        print("Nitrite:")
        print(f"\tMean: {mean(nitrite_vals):.2f}")
        print(f"\tMedian: {median(nitrite_vals):.2f}")
        print(f"\tStandard deviation: {stdev(nitrite_vals):.2f}")

        print("Nitrate:")
        print(f"\tMean: {mean(nitrate_vals):.2f}")
        print(f"\tMedian: {median(nitrate_vals):.2f}")
        print(f"\tStandard deviation: {stdev(nitrate_vals):.2f}")
```

```python
print("Temperature:")
print(f"\tMean: {mean(temp_vals):.2f}")
print(f"\tMedian: {median(temp_vals):.2f}")
print(f"\tStandard deviation: {stdev(temp_vals):.2f}")

print("DO:")
print(f"\tMean: {mean(do_vals):.2f}")
print(f"\tMedian: {median(do_vals):.2f}")
print(f"\tStandard deviation: {stdev(do_vals):.2f}")

print("EC:")
print(f"\tMean: {mean(ec_vals):.2f}")
print(f"\tMedian: {median(ec_vals):.2f}")
print(f"\tStandard deviation: {stdev(ec_vals):.2f}")
```