ASSESMENT QN

ANSU MARIUM SHIBU

Requirements

1. Define Data Types

Create a structure AquaticSpecies with the following fields:

speciesID (integer): Unique identifier for each species.

speciesName (string): Name of the aquatic species.

habitatType (string): Habitat type (e.g., "Freshwater," "Brackish," "Marine").

population (integer): Number of individuals in the farm.

averageWeight (float): Average weight per individual (in kilograms).

status (string): Farming status (e.g., "Active," "Endangered," "Under Observation").

Create a union HabitatDetails to store habitat-specific details:

temperatureRange (float[2]): Minimum and maximum water temperature for the species (for general conditions).

salinity (float): Water salinity in parts per thousand (for brackish or marine species).

oxygenLevel (float): Minimum dissolved oxygen level required (for freshwater species).

2. Features

Dynamic Memory Allocation:

Dynamically allocate memory for an array of AquaticSpecies structures based on user input (N species).

Input and Output:

Input details for each aquatic species, including habitat-specific attributes.

Allow the user to choose which specific field of the union to populate based on the habitat type.

Display:

Display all aquatic species records in a tabular format, including general details and habitat-specific attributes.

Search:

Allow the user to search for a species by speciesID and display its details.

Update:

Update the population, average weight, or status of a species.

Sorting:

Sort species by population in descending order.

3. Typedef:

Use typedef to define aliases for the AquaticSpecies structure and the HabitatDetails union.

Program Requirements

1. Menu Options

Add new species details.

Display all species records.

Search for a species by ID.

Update species details (population, average weight, or status).

Sort species by population in descending order.

Exit the program.

PROGRAM:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

typedef struct {
    int speciesID;
    char speciesName[50];
    char habitatType[20];
    int population;
    float averageWeight;
    char status[20];
} AquaticSpecies;

void addSpecies(AquaticSpecies *species, int n);
void displaySpecies(AquaticSpecies *species, int n);
void searchSpecies(AquaticSpecies *species, int n, int id);
```

```
void updateSpecies(AquaticSpecies *species, int n, int id);
void sortSpecies(AquaticSpecies *species, int n);

int main() {
    int choice, n, id;
    AquaticSpecies *species;

    printf("Enter the number of species: ");
    scanf("%d", &n);
    species = (AquaticSpecies *)malloc(n * sizeof(AquaticSpecies));

    do {
        printf("\nMenu:\n");
        printf("1. Add Species\n");
        printf("2. Display Species\n");
        printf("3. Search Species by ID\n");
        printf("4. Update Species\n");
        printf("5. Sort by Population\n");
        printf("6. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                addSpecies(species, n);
                break;
            case 2:
                displaySpecies(species, n);
                break;
            case 3:
                printf("Enter Species ID to search: ");
```

```
            scanf("%d", &id);
            searchSpecies(species, n, id);
            break;
        case 4:
            printf("Enter Species ID to update: ");
            scanf("%d", &id);
            updateSpecies(species, n, id);
            break;
        case 5:
            sortSpecies(species, n);
            displaySpecies(species, n);
            break;
        case 6:
            printf("Exiting program.\n");
            break;
        default:
            printf("Invalid choice. Please try again.\n");
    }
  } while (choice != 6);

  free(species);
  return 0;
}

void addSpecies(AquaticSpecies *species, int n) {
  for (int i = 0; i < n; i++) {
    printf("\nEnter details for species %d:\n", i + 1);
    printf("Species ID: ");
    scanf("%d", &species[i].speciesID);
    printf("Species Name: ");
    scanf("%s", species[i].speciesName);
```

```
        printf("Habitat Type (Freshwater/Brackish/Marine): ");
        scanf("%s", species[i].habitatType);
        printf("Population: ");
        scanf("%d", &species[i].population);
        printf("Average Weight (kg): ");
        scanf("%f", &species[i].averageWeight);
        printf("Status (Active/Endangered/Under Observation): ");
        scanf("%s", species[i].status);
    }
}

void displaySpecies(AquaticSpecies *species, int n) {
    printf("\nID\tName\t\t\tHabitat\t\tPopulation\tWeight\t\tStatus\n");
    for (int i = 0; i < n; i++) {
        printf("%d\t%-20s\t%-15s\t%d\t\t%.2f\t\t%s\n",
               species[i].speciesID, species[i].speciesName,
               species[i].habitatType, species[i].population,
               species[i].averageWeight, species[i].status);
    }
}

void searchSpecies(AquaticSpecies *species, int n, int id) {
    for (int i = 0; i < n; i++) {
        if (species[i].speciesID == id) {
            printf("\nSpecies Found:\n");
            printf("ID: %d, Name: %s, Habitat: %s, Population: %d, Weight: %.2f, Status: %s\n",
                   species[i].speciesID, species[i].speciesName, species[i].habitatType,
                   species[i].population, species[i].averageWeight, species[i].status);
            return;
        }
    }
```

```
    printf("\nSpecies with ID %d not found.\n", id);
}

void updateSpecies(AquaticSpecies *species, int n, int id) {
    for (int i = 0; i < n; i++) {
        if (species[i].speciesID == id) {
            printf("\nUpdating details for species ID %d:\n", id);
            printf("New Population: ");
            scanf("%d", &species[i].population);
            printf("New Average Weight (kg): ");
            scanf("%f", &species[i].averageWeight);
            printf("New Status: ");
            scanf("%s", species[i].status);
            printf("Details updated successfully.\n");
            return;
        }
    }
    printf("\nSpecies with ID %d not found.\n", id);
}

void sortSpecies(AquaticSpecies *species, int n) {
    AquaticSpecies temp;
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (species[j].population < species[j + 1].population) {
                temp = species[j];
                species[j] = species[j + 1];
                species[j + 1] = temp;
            }
        }
    }
```

```
    printf("\nSpecies sorted by population in descending order.\n");
}
```

OUTPUT:

```
Enter the number of species: 2

Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 1

Enter details for species 1:
Species ID: 12
Species Name: salmon
Habitat Type (Freshwater/Brackish/Marine): freshwater
Population: 200
Average Weight (kg): 34
Status (Active/Endangered/Under Observation): active

Enter details for species 2:
Species ID: 45
Species Name: salmon1
Habitat Type (Freshwater/Brackish/Marine): marine
Population: 700
Average Weight (kg): 67
Status (Active/Endangered/Under Observation): active

Menu:
```

```
Species ID: 45
Species Name: salmon1
Habitat Type (Freshwater/Brackish/Marine): marine
Population: 700
Average Weight (kg): 67
Status (Active/Endangered/Under Observation): active

Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 2

ID      Name                    Habitat         Population      Weight          Status
12      salmon                  freshwater      200             34.00           active
45      salmon1                 marine          700             67.00           active

Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
```

```
Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 3
Enter Species ID to search: 12

Species Found:
ID: 12, Name: salmon, Habitat: freshwater, Population: 200, Weight: 34.00, Status: active

Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice:
```

```
Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 4
Enter Species ID to update: 45

Updating details for species ID 45:
New Population: 600
New Average Weight (kg): 89
New Status: active
Details updated successfully.
```

```
Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 5

Species sorted by population in descending order.

ID      Name                    Habitat         Population      Weight          Status
45      salmon1                 marine          600             89.00           active
12      salmon                  freshwater      200             34.00           active

Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
```

```
Menu:
1. Add Species
2. Display Species
3. Search Species by ID
4. Update Species
5. Sort by Population
6. Exit
Enter your choice: 6
Exiting program.


...Program finished with exit code 0
Press ENTER to exit console.
```