Guided Exercise 05

Closed Book; Closed Notes; Time Given=90 minutes

"By proceeding I certify that I have neither received nor given unpermitted aid on this examination and that I have reported all such incidents observed by me in which unpermitted aid is given."

Zip the assignment folder and rename it to `'<rollnumer>_GE5.zip'`. Upload your zip file in the corresponding LMS submission tab. Example, `'26100036_GE5.zip'`

For this guided exercise, you need to access the file 'GE5.cpp'. It provides you with the required function definitions as well the test cases.

1 Introduction

The Pet Adoption Center Management System is designed to manage the collection of pets available for adoption in a pet shelter. The system allows the addition, removal, display, and searching of pets within the center.

2 System Requirements

You are to implement the system with the following two classes:

2.1 Pet Class

The `Pet` class is the core representation of a pet in the system. It should include the following attributes and methods:

2.1.1 Attributes

- **`name`**: A string representing the pet's name.
- **`species`**: A string representing the species of the pet.
- **`petID`**: An integer representing a unique ID for the pet.

2.1.2 Methods

- **`Pet()`**: A default constructor that initializes the pet's name and species to "N/A" and the pet ID to -1.
- **`Pet(string uname, string uspecies, int upetID)`**: A parameterized constructor to initialize a pet with given details.
- **`setPet(string uname, string uspecies, int upetID)`**: A method to set or update the pet's details.

- `displayPet()`: A method to display the pet's information, including its name, species, and ID.
- `getPetID()`: A method to retrieve the pet's unique ID.
- `getName()`: A method to retrieve the pet's name.
- `getSpecies()`: A method to retrieve the pet's species.
- `calculateAgeInDays()`: A method which returns the age of the pet in days provided the birth date of the pet and the current date as parameters. Keep in mind the date is passed as an int of form YYMMDD. Further you may simplify the calculation by assuming each year has 365 days and each month has exactly 30

2.2 PetAdoptionCenter Class

The `PetAdoptionCenter` class manages the collection of pets in the adoption center. It includes the following attributes and methods:

2.2.1 Attributes

- `pets`: An array of `Pet` objects, representing the catalog of pets available for adoption.

  Note: Just like how you can have an array of integers (e.g., int arr[10];), you can also have an array that stores objects of a class. For example, we have a Pet class, we can create an array of Pet objects inside another function or a class. The array of pet objects is considered an attribute of PetAdoptionCenter.

  To make an array of objects, you just declare the array using the class name as the type. Each element in this array will be an instance of the Pet class.

- `currentCount`: An integer tracking the current number of pets in the center.
- `CAPACITY`: A constant representing the maximum capacity of pets in the center (e.g., 100 pets).

  Note: Declare CAPACITY as a static constant integer. This is because, the size of a (static) array must be a compile-time constant. A constant declared inside a class but not marked as static is not guaranteed to be a compile-time constant. Using *static const* guarantees that the value will be known at compile time, making it suitable for array size declaration.

2.2.2 Methods

- `bool addPet(const Pet &newPet)` : Allows the user to add a new pet to the center. It prompts for the pet's name, species, and ID, and adds the pet to the catalog if there is available capacity. It should return true or false depending on whether it was successful.

  Note: The ampersand (&) denotes that the parameter (pet object) is passed by reference. Instead of passing a copy of the Pet object, a reference to the existing Pet object is passed. This avoids making a copy of the object, which can be more efficient, especially if the object is large.

- **`removePet(int petID)`**: Removes a pet from the catalog based on its ID. It resets the pet's information to indicate that it is no longer available for adoption. It should return true or false depending on whether it was successful.
- **`displayPets()`**: Displays the details of all pets currently available in the center. For each pet in the center, you should call displayPet() function of Pet class.
- **`Pet * searchPet(int searchID)`**: Allows the user to search for a pet based on its searchID. The function returns a pointer to the Pet object if it finds a pet with the given searchID. If no such pet is found, it returns *nullptr*.

  Note: The return type of the function is a pointer to a Pet object because it allows us to access and modify the pet's information directly if needed, not its copy.

  Hint: A pointer is a variable that stores the address of another variable (in this case, the Pet object). So when you return **&pets[i]**, you're returning a pointer to the pet.

3 Example Usage

The following is an example of how the program might be used:

- The user adds a new pet with the name "Buddy", species "Dog", and ID 101.
- The user removes the pet "Buddy" after adoption.
- The user displays the current list of available pets.