**Object Oriented Programming**

**Fall 2022**

|  |  |
| --- | --- |
| **Lab 03** | |
| **Topic** | **Classes in C++** , **Overloaded Constructors, shallow vs deep copy, destructor, setter, getters, char \* as attribute** |
| **Objective** | Making students familiarize with classes and their implementations in  C++,methods, constructors,settersGetters**,** char \* as attributeshallow vs deep copy, destructor and access Specifiers |

**Instructions:**

* Comment your code.
* Use meaningful variable names.
* Plan your code carefully on a piece of paper before you implement it.
* Name of the program should be same as the task name. i.e. the first program should be Task\_1.cpp
* **void main() is not allowed. Use int main()**
* **You are not allowed to use system**("**pause**")
* **You are not allowed to use any built-in functions**
* **You are required to follow the naming conventions as follow:**

**Variables:** firstName; (no underscores allowed)

**Function:** getName(); (no underscores allowed)

**ClassName:** BankAccount (no underscores allowed)

**Students are required to complete the following tasks in lab timings.**

**Task 1:**

Define a class **Cat** having attributes

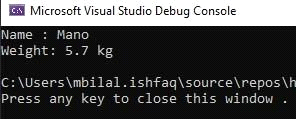
* name (char \*)
* weight (double )

Now do the following operations on the above-mentioned class:

* Write **parameterized constructor with default arguments and copy the constructor** (with an output statement to check the lifetime scope of the object)**.**
* Write a **destructor** (with an output statement to check the lifetime scope of the object)**.**
* Write separate setter (mutator) functions for each attribute to set value (**with no memory leakage**).
* Write separate getter (accessor) functions for each attribute to get value (**which should never return the original memory handler**).
* Write a **display**() function to display attributes of class Cat on screen.

Now write the main function to declare different objects of class Cat. Then call the display function against every object to see the information stored.

**For Example:**



**Task 2:**

Define a class **Student** having attributes

* name (char \*)
* registerationNumber (char \*)
* CGPA (double)
* dateOfBirth (Date)

**Hint: Use a separate class of Date to store it.**

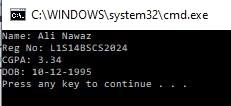
Now do the following operations on above-mentioned class:

* Write **parameterized constructor and copy the constructor** (with an output statement to check the lifetime scope of the object)**.**
* Write a **destructor** (with an output statement to check the life time scope of object)**.**
* Write separate setter (mutator) functions for each attribute to set value (**with no memory leakage**).
* Write separate getter (accessor) functions for each attribute to get value (**which should never return the original memory handler**).
* Write a **display**() function to display attributes of class Employee on screen.

**Hint: For displaying the date call the display function of class Date.**

Now write the main function to declare different objects of class Student. Then call the display function against every object to see the information stored.

**For Example:**



**Task 3:**

Define a class **Employee** having attributes

* name (char \*)
* employeeCode (char \*)
* dateOfJoining (Date)

**Hint: Use a separate class of Date to store it.**

Now do the following operations on above mentioned class:

* Write **parameterized constructor with default arguments and copy constructor** (with an output statements to check the life time scope of object)**.**
* Write a **destructor** (with an output statement to check the life time scope of object)**.**
* Write separate setter (mutator) functions for each attribute to set value (**with no memory leakage**).
* Write separate getter (accessor) functions for each attribute to get value (**which should never return the original memory handler**).
* Write a **display**() function to display attributes of class Employee on screen.

**Hint: For displaying date call display function of class Date.**

Now write a main function to create three different objects of Employee with different data.

Then ask the user to enter current date. Then call display function against every object to see information stored and also print the total number of days of the employee in the organization.

**For Example:**

