

**The objective of this lab is to:**

Refresh OOP concepts focusing on Abstract Data Types.

**Instructions:**

- 1) Follow the question instructions very carefully, no changes in function prototypes are allowed.
- 2) Make separate header files for ADTs. (.h for function prototypes) (.cpp for function implementations)

**Task 01(Complex ADT)**

**[20 Marks]**

You have studied complex numbers in elementary classes. Recall that a complex number is a number that can be expressed in the form:  $a + bi$ , where  $a$  and  $b$  are real numbers, and  $i$ , is the imaginary unit, which satisfies the equation  $i^2 = -1$ . You are required to create an ADT Complex.

**Part A:** Define a class named 'Complex' with two member variables: real(double) and imag(double). Each is a double. With default and parameterized constructors and getter setters for each, Destructor, **[2.5 Marks]**

**Note:** In each function, the first complex number will be the object calling the function, and the second complex number will be the number passed into the function as an argument.

**Part B:** Write a function named 'add' that adds two Complex numbers and returns the result. **[2.5 Marks]**  
`Complex add(const Complex& c);`

**Part C:** Write a function named 'subtract' that subtracts two Complex numbers and returns the result. **[2.5 Marks]**  
`Complex subtract(const Complex& c);`

**Part D:** Write function named 'multiply' that multiplies two Complex numbers and returns the result. **[2.5 Marks]**  
`Complex multiply(const Complex& c);`

**Part E:** Write function named 'divide' that divides two Complex numbers and returns the result. **[5 Marks]**  
`Complex divide(const Complex& c);`

**Part F:** Write function named 'conjugate' that returns Conjugate of the calling complex number **[2.5 Marks]**  
`Complex conjugate();`

**Part G:** Write function named 'display' that displays a Complex Number. **[2.5 Marks]**  
`void display();`

```
Sample output for display:
1 + 2i
2.3 - 2i
0 + 2.2i
2 - 0i
```

## Task 02 (Date ADT)

[15 Marks]

**Part A:** Define a class named 'Date' with three member variables: day(int), month(int), year(int). With default and parameterized constructors **Date(int day, int month, int year)** and getter setters, Destructor [2.5 Marks]

**Note:** For simplicity assume each month has 30 days, after which the next month starts.

**Part B:** Write a function named 'addDays' that adds certain number of days from date [5 Marks]  
`void addDays(int days);`

**Beware of Edge Cases:**

Try running your implementation on following scenario:

```
Date d(26, 1, 2024);  
d.addDays(10); // new date should be January 6th, 2024
```

**Part C:** Write a function named 'subtractDays' that subtracts certain number of days from date [5 Marks]  
`void subtractDays(int days);`

**Think of Edge Cases.**

**Part D:** Write a function named 'displayDate' that displays date in following format [2.5 Marks]  
`void displayDate();`

```
January 26, 2024  
November 13, 2023
```

## Task 03 (Animal Kingdom) (Polymorphism Refresher)

[15 Marks]

**In object-oriented programming, polymorphism refers to the ability of an object to take on many forms.**

**Part A:** Define an abstract base class named Animal with following pure virtual function [5 Marks]  
makeSound (No Parameters, void return type)

**Part B:** Define three concrete implementations (Dog, Cat and Mouse) for the interface Animal and override the virtual functions. [5 Marks]

**Part C:** Demonstrate polymorphic behavior in main function. [5 Marks]

**Good Luck!**

-----

**Note:** You must complete all your tasks individually. Absolutely NO collaboration is allowed. Any case of plagiarism/cheating would result in 0 marks in sessional activities.