Programming Practice Lab

Assignment 3

CO3: Understand and implement OOP features through C++ Programming
CO4: Design and implement the solution following OOP paradigm

1. Write a function swap (a, b) to interchange the values of two variables. Do not use pointers.

2. Write a function max (a, b) that will return the reference of larger value. Store the returned information to x where x is a i) variable of type a or b, ii) variable referring to type of a or b. In both the cases modify x. Check also the values of a and b.

3. Write a function that will have income and tax rate as arguments and will return tax amount. In case tax rate is not provided it will be automatically taken as 10%. Call it with and without tax rate.

4. Write a function void f(int) that prints "inside f(int)". Call the function with actual argument of type: i) int, ii) char, iii) float and iv) double. Add one more function f(float) that prints "inside f(float)". Repeat the calls again and observe the outcomes.

5. Define functions f(int, int) and f (char, int). Call the functions with arguments of type (int, char), (char,char) and (float, float). Observe and analyze the outcome.

6. Define a structure student with roll and score as attributes and with two member functions to take input and to show the data. Use the member functions to take data for a structure variable and to show. Write global function i) to modify score and ii) to show the data again.

7. Design a class TIME which stores hour, minute and second. The class should have the methods to support the following:
   - User may give the time value in 24-hour format.
   - User may give the time value in AM/PM format
   - Display the time in 24-hour format.
   - Display the time in AM/PM format.
   - User may like to add minute with a time value.

8. Create a STACK class with operation for initialization, push and pop. Support for checking underflow and overflow condition are also to be provided.
9. Create an APPLICANT class with application id (auto generated as last id +1), name and score. Support must be there to receive applicant data, show applicant details and to find out number of applicants.

10. Design a STUDENT class to store roll, name, course, admission date and marks in 5 subjects. Provide methods corresponding to admission (marks are not available then), receiving marks and preparing mark sheet. Support must be there to show the number of students who have taken admission.

11. Create a class for linked list. Consider a separate class NODE for basic node activities and use it in class for linked list.

12. Design the class(es) for the following scenario:
    - An item list contains item code, name, rate, and quantity for several items.
    - Whenever a new item is added in the list uniqueness of item code is to be checked.
    - Time to time rate of the items may change.
    - Whenever an item is issued or received existence of the item is checked and quantity is updated.
    - In case of issue, availability of quantity is also to be checked.
    - User may also like to know price/quantity available for an item.

13. Design a BALANCE class with account number, balance and date of last update. Consider a TRANSACTION class with account number, date of transaction, amount and transaction type (W for withdrawal and D for deposit). If it is withdrawal then check whether the amount is available or not. Transaction object will make necessary update in the BALANCE class.