Week 5: String Class, Array of Objects, Reference Variable, Use of Project feature in CodeBlocks IDE.
Term Project Proposal Discussion

Learning Materials: Chapter 7

Demonstration:

1. Create a Project in codeblocks IDE to Implement.
2. Use of reference variables in return type.
3. Array of objects
4. Class String

Task 1:

Create a **Products** class. An object of this class has Product Name (type: String), Product Category Name (type: String), Product Description (type: String multiline input), Amount in Store (type: int), Regular Price (type: float), and Discount Rate (type: float).

Provide a public getter setter function for each member variable.

Provide PurchaseProduct (int amount) and RestockProduct (int amount) functions (public) with necessary checking.

Provide a public member function **double calculateDiscount (int amountOfProducts)** that calculates the discount based on the amount of product purchased. A discount (provided in the class by Discount Rate) will be applied if the customer buys 5 or more products of the same category. An additional 3% discount will be given if the number of products purchased is 10 or more. This function should return the discounted amount (i.e. 25 BDT). Ensure that this function does not change any member data.

Provide a public member function **netTotal** (**int amountOfProducts**) which will calculate total cost, the discount, and give you the net total to pay for the customer.

Write a driver program to test the class Products.

Create 100 **Products** objects.

Write a function (non-member) void **EditInformationByKeyboard**() that takes a Products object as parameter. This function will take input from the keyboard and set the member variable. Now call this function for the first 2 objects.

Write a function (non-member) void **generateInformaiotnRandom**() takes a Products object as parameter. This function will assign the value randomly from a range. Call this function for the remaining objects.

Product Name (type: String): Comprising three words. Each word has a length of 3-7.
Product Category Name (type: String): Comprising one word of length of 3-7.
Product Description (type: String): Comprise of 10 words. Each word has a length of 3-7.
Amount in Store (type: int): indicates number of products in storage.
Regular Price (type: float): indicates regular price of the product.
Discount Rate (discount rate): 3~15 percent.

Write a function (non-member) void **ShowAllDiscount(Products ar[])** that displays the discount amount of all products.

Write a function (non-member) void **grossTotal(Products ar[])** that will calculate the net total for every product. Use a random number to determine the number of products to purchase. Sum up all the net toals for all the products, show the price, and ask for the confirmation to confirm purchase.

Write a function (non-member) void **ShowAllAlphabetically**(Products ar[]) that displays all the products name and its current storage according to the alphabetical order of the product name.

Task 2:

Define a class in C++ with following description:
Private Members:

- A data member Name: The length has to be more than two. Otherwise, assign the default name, John Doe
- A data member Date of Birth: Every employee has an age higher than 18. If an invalid value is given, assign 1 January 2002.
- A data member BaseSalary: The salary has to be in between BDT 10000 and BDT 100000. If an invalid value is given, assign BDT 10000.

- A data member JoinigYear
- A member function calculateTotalSalary() to calculate the total salary using the formula:
  - Base Salary + 10% transport allowance + 30% housing allowance + 10% miscellaneous allowance.
  - Every year, the base salary increases by 3%.
- A member function calculateBonus() to calculate the employee's bonus based on their status, years of service, and the updated base salary.
  - 5% for "Low" status.
  - 10% for "Moderate" status.
  - 15% for "High" status.
- A member function getStatus() to **return** the status of an employee on the basis of the following criteria

Age	**Total Salary**	**Status**
<=25	<=20000	Low
	>20000	Moderate
>25	<=21000	Low
	>21000 and <=60000	Moderate
	>60000	High

Define the setInfo() function, which will call all the setter functions to set the necessary information of an employee object.
Define a function named getInfo(), which will display all the stored information belonging to an employee object using the return value of the getter function.

Include one const member function named Employee& compareTotalSalary(Employee& e) which will return the Higher salaried employee object based on the date of birth.

In the main function, create 100 employees.

Write a function (non-member) void **generateInformaiotnRandom**() takes an employee object as a parameter. This function will assign the values from a valid range. Call this function for the all created objects. Hints: See last page.

Write a function (non-member) void **ShowAllBasedOnTotalSalary**(Employee ar[]) that displays all the employees names and total salary in ascending order.

```
/// Some example code
#include <string>
#include <cstdlib> // For rand() and srand()
#include <ctime>   // For time()

// Function to generate random string
std::string generateRandomString(int length) {
    std::string randomString;
    const char alphabet[] = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
       int alphabetSize = sizeof(alphabet) - 1; // Exclude null
character

      srand(static_cast<unsigned int>(time(0))); // Seed the random
number generator

    for (int i = 0; i < length; ++i) {
        randomString += alphabet[rand() % alphabetSize];
    }

    return randomString;
}

int randomInRange(int min, int max) {
    // Ensure min is less than or equal to max
    if (min > max) {
        std::swap(min, max); // Swap if min is greater than max
    }

    return rand() % (max - min + 1) + min;
}

// Function to generate a random double within a given range [min,
max]
double randomInRange(double min, double max) {
    // Ensure min is less than or equal to max
    if (min > max) {
        std::swap(min, max); // Swap if min is greater than max
    }

    // Generate a random double between 0 and 1
    double randomFraction = static_cast<double>(rand()) / RAND_MAX;

    // Scale and shift the random value to the desired range
    return min + randomFraction * (max - min);
}
```