

---

**Deadline – Wednesday, October 14, 2020****NOTE:**

1. **THIS ASSIGNMENT IS NOT LINKED WITH ASSIGNMENT NO. 4 SO DON'T USE OLD SOLUTION/CODE. THIS ASSIGNMENT IS IMPORTANT**
2. **Refer linq-cheatsheet.pdf from [Notes] folder for LINQ syntax.**

**Open Assignment No. 6 Console Application for HR department to analyze the dataset using LINQ Employee(EmpNo, Name, Designation, Salary, Commission, DeptNo) and Department(DeptNo, DeptName, Location) data.**

You need to provide following functionalities:

1. Create department and employee classes override ToString() to print object contents
2. In Main() method take department details from dept.csv as input and initialize DeptList (Dictionary)

Hint:

```
// Open file and Read line by line
while ((deptstring = reader.ReadLine()) != null) {
//line fetched from csv file
//string deptstring = "50,Training,Karad";
// Split details separated by a comma followed by space
string[] deptDetails = deptstring.Split(",");
Department dept = new Department();
dept.DeptNo=int.Parse(deptDetails[0]); dept.DeptName=deptDetails[1];
dept.Location=deptDetails[2];
//add dept object in DeptList
DeptList.Add(10, dept);
}
```

3. In Main() method take employee details from emp.csv as input and initialize EmpList (Dictionary) same like task 2

**USE LINQ for Sr. No. 4 - 11**

4. Find location name with single department in the dataset (use LINQ).
  - a. Implement your code into LocationWithSingleDept(DeptList) function.
  - b. You are expected to return list of location in which only one department is given in the data.
  - c. Function return type must be List<String>.
  - d. If any location having multiple departments, then return empty list.
5. Find department names in which no employees  
e.g. List<string> FindDeptsWithNoEmps(EmpList, DeptList)
6. Write function to calculate total salary (Sal+Comm) of all employees  
e.g. double Calculate\_Total\_Salary(EmpList)
7. Write function to display all employees of particular department  
e.g. List<Employee> GetAllEmployeesByDept(int DeptNo)
8. Write function to calculate department wise count of employees  
e.g. Dictionary<DeptNo,Double> DeptwiseStaffCount(EmpList)
9. Write function to calculate department wise average salary  
e.g. Dictionary<DeptNo,Double> DeptwiseAvgSal(EmpList)
10. Write function to calculate department wise minimum salary  
e.g. Dictionary<DeptNo,Double> DeptwiseMinSal(EmpList)
11. Display employee names along with department names  
e.g. Dictionary<EmpNo, DeptName, EmpName> GetEmpInfo(EmpList, DeptList)

**git commit -m "Assignment no. 6 modified"**

Important methods of (Immutable) String Class:

- Split
- Trim
- ToLower
- ToUpper
- Contains
- StartWith
- EndWith
- PadLeft
- PadRight
- Concat
- Join
- Remove
- Replace
- Reverse<>
- Insert
- IndexOf
- ToArray
- ToCharArray
- ToDictionary<>
- ToList<>

Following C# collections are important:

Dictionary<TKey, TValue>	ArrayList	List<T>
+ Count: int + Item[TKey]: object + Keys: IList<TKey> + Values: IList<TValue>	+ Capacity: int + Count: int + Item[int] + IsFixedSize: bool + IsReadOnly: bool + IsSynchronized: bool + SyncRoot	+ Capacity: int + Count: int + Item[int]: object
+ Add(TKey, TValue): void + Clear(): void + ContainsKey(TKey): bool + ContainsValue(TValue): bool + Remove(TKey): bool + ToString(): string + TryGetValue(TKey, out TValue): bool	+ Add(object): int + AddRange(ICollection): int + BinarySearch(object) + Clear(): void + Contains(object): bool + CopyTo(Array): void + IndexOf(object): int + Insert(int, object): void + LastIndexOf(object): int + Remove(object): void + RemoveAt(int): void + RemoveRange(int, int): void + Repeat(obj, int): ArrayList + Reverse(): void + SetRange(int, ICollection): void + Sort(): void + ToArray(): object[] + ToString(): string + TrimToSize(): void	+ Add(T): void + AddRange(IEnumerable<T>): void + BinarySearch(T): int + Clear(): void + Contains(T): bool + CopyTo(T[], int): void + Exists(Predicate<T>): bool + Find(Predicate<T>): T + ForEach(Action<T>): void + IndexOf(T): int + Insert(int, T): void + InsertRange(int, IEnumerable<T>): void + Remove(T): bool + RemoveRange(int, int): void + Reverse(): void + Sort(IComparer<T>): void + ToArray(object): object[] + ToString(): string + TrimToSize(): void