 **COSC 1320**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Seat # \_\_\_**

**Estimated Hours 1**

**Actual Hours**

**TA (Textual Analysis) for UML USE CASE Diagram MODEL**

**Programming Assignment 3:**

**(15 points)**

**(Due date found in the COSC 1320 BB Calendar!)**

**You must use Microsoft WORD program. Insert the TEMPLATEs for Actors and Use Cases in this Requirements Document.**

*Please have Step 1, 2, 3, and 4 in this order.*

PLEASE use the “TA for UML USE CASE Diagram MODEL Movie Company System.doc” AS TEMPLATE.

(Do not Show STEP 1,…; do not change Line Numbers; do not change Number of Pages)

Any DIAGRAM that is NOT the result of CUT and PASTE

WILL BE IGNORED. (YOU WILL GET ZERO POINTS)

**Requirements Analysis**: Construct the **UML Use Case Diagram** MODEL

**Requirements Analysis**: Construct the **UML USE CASE** Diagram MODEL

**ProgrammingAssignment3** **APPLICATION** to automate **adding**, **deleting**, and **displaying** the list of employees in a hospital.

EmployeeRoster

The set of **Class**es define these employees of a hospital: hospital employee, doctor, nurse, administrator, surgeon, receptionist, and janitor. **MUST use inheritance** in creating these **Class**es. You can use **arrays** to store objects of the same **Class**, you can assume MAX\_SIZE = 5 or **one array** to store all objects of all the **Class**es, you can assume MAX\_SIZE = 20.

(you are **not** to import STL Data Structures Classes - failure to conform will result in losing 20 points; you are **not** to use **STL** **ArrayList** or **LinkedList** or **Vector** Classes - failure to conform will result in losing 40 points; you are **to** use an C array - failure to conform will result in losing 30 points)

These employees are initially read and added from the **“Programming Assignment 3 Data.txt”** and are also **saved in same file** when your **ProgrammingAssignment3** **APPLICATION** terminates.

UC4 UCReadFile

Your **ProgrammingAssignment3** **APPLICATION** must **read** in the data file (FILE INPUT) into your **ProgrammingAssignment3** **APPLICATION** memory (**please DO NOT read from File and write to File skipping reading into the program memory**).

Your **ProgrammingAssignment3** **APPLICATION** must allow the user to **delete** a hospital employee, doctor, nurse, administrator, surgeon, receptionist, and janitor given the role and the name.

UC2 UCDeleteEmployee

Your **ProgrammingAssignment3** **APPLICATION** must allow the user to **add** a hospital employee, doctor, nurse, administrator, surgeon, receptionist, and janitor.

UC1 UCAddEmployee

Your **ProgrammingAssignment3** **APPLICATION** must allow the user to **display** **the hospital employees** in the format given below.

UC3 UCDisplayEmployee

Your **ProgrammingAssignment3** **APPLICATION** must allow the user to **save** the hospital employees before it exits from the **ProgrammingAssignment3** **APPLICATION** from the memory to the **SAME** data file (FILE OUTPUT) **“Programming Assignment 3 Data.txt”**.

UC5 UCWriteFile

Use **Constructors** to automatically initialize the **instance variables** that MUST be declared **private** - failure to make ALL **member variables** **private** will result in losing 20 points. Appropriate accessor and mutator **methods** for each **private** **instance variable** must be created.

Add (**overwrite**) the **methods** to display (must use **toString** method and super - failure to use **toString** will result in losing 10 points; - failure to use super will result in losing 10 points).

All **M**odel **Class**es must have the **equals** and **toString** methods, please **place them at the end** of the **Class**.

**MUST use inheritance** in creating these **Class**es - failure to use **inheritance** will result in losing 20 points).

A sample input file **“Programming Assignment 3 Data.txt”** follows:

h Vito 123

d Michael 234 Heart

n Sonny 456 6

a Luca 567 Business

r Tom 678 Talking Y

j Anthony 789 Maintenance N

d Nicos 891 Bone

The h stands for hospital employee role, Vito for **name** (only one string), and 123 is its hospital employee **number**.

The d stands for doctor, Michael for **name**, 234 for **number**, and Heart for **specialty**.

The s stands for surgeon, Vincent for **name**, 345 for **number**, Brain for **specialty**, and Y for **operating**.

The n stands for nurse, Sonny for **name**, 456 for **number**, and 6 for **numpatients**.

The a stands for administrator, Luca for **name**, 567 for **number**, and Business for **department**.

The r stands for receptionist, Tom for **name**, 678 for **number**, Talking for **department**, and Y for **answering**.

The j stands for janitor, Anthony for **name**, 789 for **number**, Maintenance for **department**, and Y for **sweeping**.

Format to **display** the Hospital Employees use the following format (**must use** the **toString**() **method**):

**The Hospital has the following employees:**

**Hospital Employees: 1**

**Name: Vito Employee Number: 123**

**Doctors: 2**

**Name: Michael Employee Number: 234 Specialty: Heart**

**Name: Nicos Employee Number: 891 Specialty: Bone**

**Surgeons: 0**

**Nurses: 1**

**Name: Sonny Employee Number: 456 Number of Patients: 6**

**Administrators: 1**

**Name: Luca Employee Number: 567 Department: Business**

**Receptionists: 1**

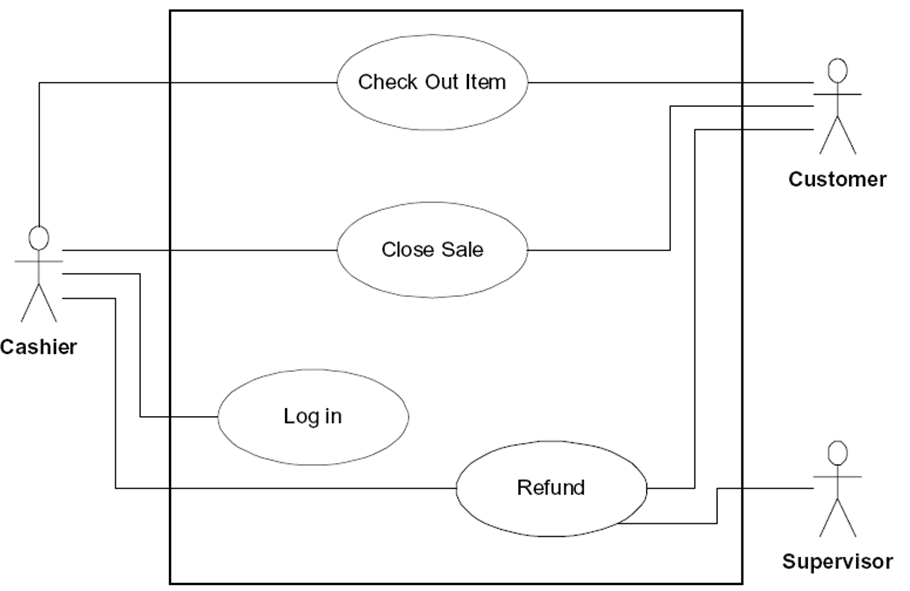
**Name: Tom Employee Number: 678 Department: Talking Answering: Y**

**Janitors: 1**

**Name: Anthony Employee Number: 789 Department: Maintenance Sweeping: N**

**Total number of Employees: 7**

Create a **ProgrammingAssignment3. cpp** that contains the **main method**.



**A1: User**

UCAddEmployee

UCDeleteEmployee

UCDisplayEmployee

UCReadFile

UCWriteFile

**Controller**

**UC 1**

**UC2**

**UC3**

**UC4**

**UC5**

|  |  |
| --- | --- |
| Name: | UCAddEmployee |
| Actor: | User |
| Description: | This use case describes the process used by User to Add another employee |
| Successful Completion: | User requests Adding employee   1. EmployeeRoster prompts the user for employee information 2. HospitalEmployee (or child) is added |
| Alternative: |  |
| Pre-Condition: | User requests Add employee |
| Post-Condition: | HospitalEmployee (or child) Added successfully or UNsuccessfully |
| Assumptions: | None |

|  |  |
| --- | --- |
| Name: | UCDeleteEmployee |
| Actor: | User |
| Description: | This use case describes the process used by User to delete another employee |
| Successful Completion: | User requests deleting employee   1. EmployeeRoster prompts the user for employee information 2. HospitalEmployee (or child) is added |
| Alternative: |  |
| Pre-Condition: | User requests delete employee |
| Post-Condition: | HospitalEmployee (or child) deleted successfully or UNsuccessfully |
| Assumptions: | None |

|  |  |
| --- | --- |
| Name: | UCDisplayEmployee |
| Actor: | User |
| Description: | This use case describes the process used by User to display list of employees |
| Successful Completion: | User requests to display all employees   1. EmployeeRoster calls displayEmployee 2. All employees in empArr[] are displayed |
| Alternative: |  |
| Pre-Condition: | User requests delete employee |
| Post-Condition: | All employees in empArr[] are displayed |
| Assumptions: | None |

|  |  |
| --- | --- |
| Name: | UCReadFile |
| Actor: | application |
| Description: | This use case describes the process used by the application to populate empArr[] with data from a file. |
| Successful Completion: | Application checks to see if file exists   1. empArr[] is populated with data from file |
| Alternative: | File does not exist, program terminates. |
| Pre-Condition: | Application attempts to read from file |
| Post-Condition: | empArr[] is populated with data from file. |
| Assumptions: | None |

|  |  |
| --- | --- |
| Name: | UCWriteFile |
| Actor: | User |
| Description: | This use case describes the process used by User to save empArr[] to a file and exit |
| Successful Completion: | User requests to write empArr[] to a file   1. Opens/creates the file, and populates the file with the current data in empArr[] |
| Alternative: |  |
| Pre-Condition: | User requests to save contents and exit |
| Post-Condition: | empArr[] is saved to a file and application terminates |
| Assumptions: | None |