 **COSC 1320**

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

**Estimated Hours 2**

**Actual Hours**

**TA (Textual Analysis) for UML MVC CLASS 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 Class, Attribute, etc. in this Requirements Document.**

*Please have Step 1, 2, 3, 4, and 5 in this order.*

PLEASE use the “TA for UML MVC Class 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 MVC CLASS** 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.

EmployeeRoster: 4: readFile():void

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.

EmployeeRoster: 2: deleteEmployee():void

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

EmployeeRoster: 1: addEmployee():void

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

EmployeeRoster: 3: displayEmployee():void

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”**.

EmployeeRoster: 5: writeFile():void

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

HospitalEmployee: 1: role: char

HospitalEmployee

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

HospitalEmployee: 1: getRole():char

HospitalEmployee: 2: getName (): string

HospitalEmployee: 3: getID():int

HospitalEmployee: 4: setRole(char):void

HospitalEmployee: 5: setName(string):void

HospitalEmployee: 6: setID(int):void

HospitalEmployee: 3: idNumber: int

HospitalEmployee: 2:name: string

HospitalEmployee: 7: toString():string

HospitalEmployee: 8: equals(HospitalEmployee):bool

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

Doctor: 1: getSpec():string

Doctor: 2: setSpec(string):void

Doctor: 3: toString():string

Doctor: 4: equals(HospitalEmployee):bool

Doctor:1 : specialty:string

Doctor

Surgeon

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

Surgeon: 1: setOpStat(): char

Surgeon: 2: getOpStat(char): void

Surgeon: 3: toString(): string

Surgeon: 4: equals(HospitalEmployee):bool

Surgeon: 1: operating: char

Nurse

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

Nurse: 1: getPtCnt():int

Nurse: 2: setPtCnt (int): void

Nurse: 3: toString(): string

Nurse: 4: equals(HospitalEmployee):bool

Nurse: 1: numPatients: int

Administrator

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

Administrator: 1: setDept (string):void

Administrator: 2: getDept (): string

Administrator: 3: toString(): string

Administrator: 4: equals(HospitalEmployee):bool

Administrator: 1 department: string

Receptionist

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

Receptionist: 1: getAnswering (void):char

Receptionist: 2: setAnswering (char): void

Receptionist: 3: toString(): string

Receptionist: 4: equals(HospitalEmployee):bool

Receptionist: 1: answering: char

Janitor

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

Janitor: 1: getSweeping (void):char

Janitor: 2: setSweeping (char): void

Janitor: 3: toString(): string

Janitor: 4: equals(HospitalEmployee):bool

Janitor: 1: sweeping: char

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**.

|  |
| --- |
| Driver |
| -View //1  -EmployeeRoster //2 |
| +main():int |

|  |
| --- |
| View |
|  |
| + showMenu():void |

Has-a

|  |
| --- |
| EmployeeRoster |
| -empArr[]:HospitalEmployee\* //1  -empCtr:int //2 |
| + addEmployee():void //1  + deleteEmployee():void //2  + displayEmployee():void //3  + readFile():void //4  + writeFile():void //5 |

|  |
| --- |
| Janitor |
| - sweeping: char //1 |
| + getSweeping (void):char //1  + setSweeping (char): void //2  + toString(): string //3  + equals(HospitalEmployee):bool //4 |

|  |
| --- |
| HospitalEmployee |
| - role: char //1  - name: string //2  - idNumber: int //3 |
| + getRole():char //1  + getName (): string //2  + getID():int //3  + setRole(char):void //4  + setName(string):void//5  + setID(int):void //6 |

|  |
| --- |
| Doctor |
| - specialty:string //1 |
| + getSpec():string//1  + setSpec(string):void//2  + toString():string//3  + equals(HospitalEmployee):bool//4 |

|  |
| --- |
| Nurse |
| - numPatients: int //1 |
| + getPtCnt():int //1  + setPtCnt (int): void //2  + toString(): string //3  + equals(HospitalEmployee):bool //4 |

|  |
| --- |
| Surgeon |
| - operating: char //1 |
| + setOpStat(): char //1  + getOpStat(char): void //2  + toString(): string //3  + equals(HospitalEmployee):bool //4 |

|  |
| --- |
| Administrator |
| - department: string //1 |
| + setDept (string):void //1  + getDept (): string //2  + toString(): string //3  + equals(HospitalEmployee):bool //4 |

|  |
| --- |
| Receptionist |
| - answering: char //1 |
| + getAnswering (void):char //1  + setAnswering (char): void //2  + toString(): string //3  + equals(HospitalEmployee):bool //4 |

Is-a

uses

Has-a

Is-a

Is-a

Is-a

Is-a

Is-a

|  |
| --- |
| Driver |
| -View //1  -EmployeeRoster //2 |
| +main():int |

int main(){

creates the View and EmployeeRoster objects.

Loops for user input.

Calls appropriate EmployeeRoster method.

|  |
| --- |
| View |
|  |
| + showMenu():void |

}

void showMenu(){

displays the menu

|  |
| --- |
| EmployeeRoster |
| -empArr[]:HospitalEmployee\* //1  -empCtr:int //2 |
| + addEmployee():void //1  + deleteEmployee():void //2  + displayEmployee():void //3  + readFile():void //4  + writeFile():void //5 |

}

public void addEmployee(){

prompts user for input to add an employee to the array

}

public void deleteEmployee(){

prompts user for input to delete a specific employee from the array

}

public void displayEmployee(){

loops through the empArr[] and displays the object to the console.

}

public void readFile(){

reads “Programming Assignment 3 Data.txt” and adds contents to empArr[]

}

public void readFile(){

writes empArr[]contents to “Programming Assignment 3 Data.txt”

}