SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

ICT 162

Object Oriented Programming

Tutor-Marked Assignment

July 2021 Presentation

TUTOR-MARKED ASSIGNMENT (TMA)

This assignment is worth 24% of the final mark for ICT162, Object Oriented Programming.

The cut-off date for this assignment is **Wednesday, 27 October 2021, 2355 hrs.**

Note to Students:

Submit your solution document in the form of a single MS Word file. You are to include the following particulars in your submission: Course Code, Title of the TMA, SUSS PI No., Your Name, and Submission Date. Put this information in the first page of your solution document. Use the template word document provided – **Solution_Template.docx. Rename the file with your suss PI and full name according to the instructions under Assignment in your T group in Canvas.** Do NOT submit as a pdf document.

You should make only one submission for TMA.

You are to copy and paste Python source code into your solution document as text. If you submit source code as image, no marks will be awarded to your program. Submit screenshots for **only** output of your program, where applicable.

Assignment Requirements:

- Unless specified in the question, you CANNOT use packages not covered in this module, e.g. re, collections, numpy, pandas etc.
- All classes must be documented. Provide sufficient comments to your code and ensure that your program adheres to good programming practices such as not using global variables.
- Failing to do so can incur a penalty of as much as 50% of the mark allotted.

Question 1 (22 marks)

For companies resuming operations after the circuit breaker period, all Safe Management Measures should be in place before operations can be resumed at the workplace. One of these measures is to support as many employees in working from home as possible.

Refer to the class diagram in Figure 1. The Employee classes used for Leave Application System are enhanced to record if an employee is "work-from-home" (WFH).

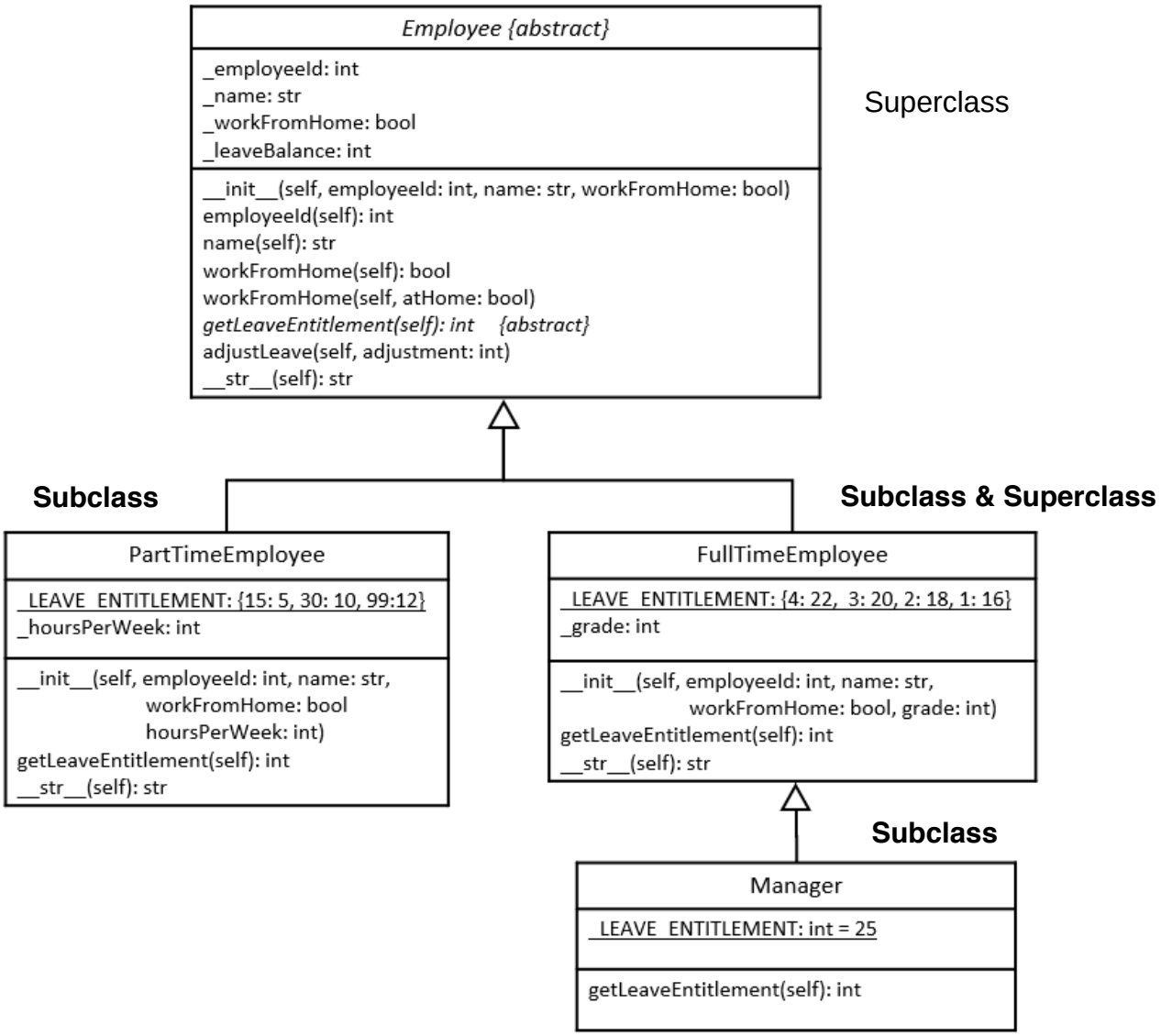


Figure 1

a) Implement the Employee class.

The Employee class is an abstract superclass that models one employee, which has:

- Four instance variables: _employeeId (int), _name (string), _workFromHome (boolean) and _leaveBalance (int). For this TMA, we do not consider ½ day leave. All leaves applied are full day.
- Constructor that initialises all variables except _leaveBalance is set to 0.
- Getter methods for the 4 instance variables. Use the property decorator.
- Setter method for _workFromHome. Use the setter decorator.
- There is one abstract method – getLeaveEntitlement which returns the number of leave entitlement for this employee.
- Method adjustLeave takes in a parameter adjustment (int) and add to the _leaveBalance.
- The __str__ method returns a string representation of an Employee object. Here is an example in the suggested format:

```
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No
```

(5 marks)

b) Implement the PartTimeEmployee class.

The PartTimeEmployee class is a subclass of Employee, and it has:

- One additional instance variable: _hoursPerWeek representing the hours this part time employee worked in a week.
- Besides initialising the instance variable _hoursPerWeek, the constructor also determines the starting leave balance for part time employees.
- The method getLeaveEntitlement computes and returns the starting leave balance for part time employees using class variable _LEAVE_ENTITLEMENT, which is a dictionary that represents the following table:

Hours worked per week	Leave Entitlement (per year)
<= 15	5 days
<= 30	10 days
> 30	12 days

- The __str__ method returns a string representation of a PartTimeEmployee object. Here is an example in the suggested format:

```
ID: 103   Name: Joe      Leave Balance: 10   WFH: No    Hours/Week: 20
```

(6 marks)

c) Implement the FullTimeEmployee and Manager classes.

The FullTimeEmployee class is a subclass of Employee, and it has:

- One additional instance variable: _grade representing the grouping of the employees with similar positions or values in order to assign leave entitlement.
- Beside initialising the instance variable _grade, the constructor also determines the starting leave balance for full time employees.

- The method getLeaveEntitlement computes and returns the starting leave balance for full time employees using class variable _LEAVE_ENTITLEMENT, which is a dictionary representing the following table:

Grade	Leave Entitlement (per year)
4	22 days
3	20 days
2	18 days
1	16 days
Others	16 days

- The __str__ method returns a string representation of a FullTimeEmployee object. This is an example in the suggested format:

```
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No    Grade: 3
```

The Manager class is a subclass of FullTimeEmployee.

- There is no additional instance variable.
- The method getLeaveEntitlement references the class variable _LEAVE_ENTITLEMENT to return the leave balance for managers.

(8 marks)

d) Write a main() function to do the following:

i) Create the appropriate Employee objects with these details.

Employee ID	Name	Work-From-Home	Employee Type	Grade	Hours Per Week
101	Jeff	False	Full Time	3	
102	Jim	True	Full Time	4	
103	Joe	False	Part Time		20
104	Jack	True	Full Time	2	
105	Jane	False	Full Time	1	
106	Tom	False	Manager	4	
201	Neil	False	Manager	4	
205	Charles	False	Full Time	4	
204	Darren	True	Part Time		32
203	Elliot	False	Full Time	3	
202	Fred	True	Part Time		10

ii) Put the Employee objects created in (i) into a collection and invoke the __str__ method of the Employee objects. Your results should look like the following:

```
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No    Grade: 3
ID: 102   Name: Jim      Leave Balance: 22   WFH: Yes   Grade: 4
ID: 103   Name: Joe      Leave Balance: 10   WFH: No    Hours/Week: 20
ID: 104   Name: Jack     Leave Balance: 18   WFH: Yes   Grade: 2
ID: 105   Name: Jane     Leave Balance: 16   WFH: No    Grade: 1
ID: 106   Name: Tom      Leave Balance: 25   WFH: No    Grade: 4
ID: 201   Name: Neil     Leave Balance: 25   WFH: No    Grade: 4
ID: 205   Name: Charles  Leave Balance: 22   WFH: No    Grade: 4
ID: 204   Name: Darren   Leave Balance: 12   WFH: Yes   Hours/Week: 32
ID: 203   Name: Elliot   Leave Balance: 20   WFH: No    Grade: 3
ID: 202   Name: Fred     Leave Balance: 5    WFH: Yes   Hours/Week: 10
```

iii) The company needs to rotate the employees' workplace arrangement every 2 weeks. Write codes to toggle the Work-From-Home setting for each employee. That is from True to False, or False to True. Print out the results similar to (ii) to show that WFH is toggled successfully.

(3 marks)

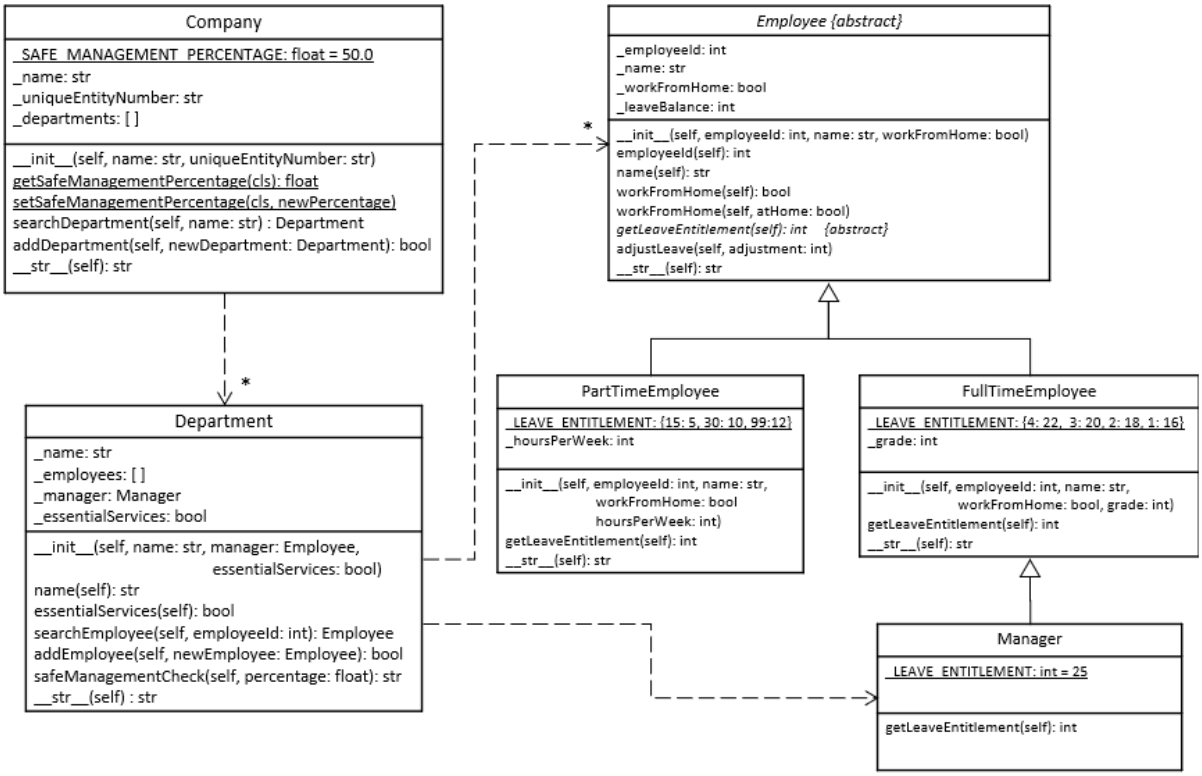
Question 2 (18 marks)

Recording the work-from-home (WFH) status of each employee is only the first step to help companies manage and support the overall Safe Management Measures.

Write the Company and Department classes to help determine if the company/departments are ensuring COVID-safe workplace, meeting the minimal percent of employee working from home.

Refer to the class diagram in Figure 2.

Solid Arrow Means There is Inheritance



Dotted Arrow Means Dependency Arrow

Figure 2

a) Implement the Department class.

The Department class has:

- Four instance variables: _name (string), _employees (list), _manager (Employee) and _essentialServices (boolean).
- Constructor initialises these 3 variables: _name, _manager and _essentialServices. The instance variable _employees is set to an empty list.

- Getter methods for the 2 instance variables: _name and _essentialServices. Use the property decorator.
- Given an employeeId as parameter, the searchEmployee method searches the list of employees as well as the match for _manager and returns the Employee object with the matching employeeId. If not found, it returns None.
- The addEmployee method has an Employee object as parameter and adds it into the _employees list if this employee is not present in the department. The method returns True if the employee is added successfully into the list, and False otherwise. A manager is not added in the list.
- The safeManagementCheck method returns a string as follows:

```
No. of Employees working from home: 3 (50.0%) - passed requirement.
```

  The method counts all the employees who are working from home and computes the percentage of employee working from home (include the manager).

  Display "passed requirement" if the percentage working from home is greater or equal to the parameter "Percentage". Otherwise display "failed requirement".

  However, a department can be "exempted" from the above check if it is providing essential services. In such a case, the string output can be:

```
No. of Employees working from home: 2 (33.3%) – exempted.
```

- The __str__ method returns a string representation of a Department object. Here is an example in the suggested format:

```
Department: IT Helpdesk   Essential Services: Yes
Manager ID: 106   Name: Tom      Leave Balance: 25   WFH: No    Grade: 4
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No    Grade: 3
ID: 102   Name: Jim      Leave Balance: 22   WFH: Yes   Grade: 4
ID: 103   Name: Joe      Leave Balance: 10   WFH: No    Hours/Week: 20
ID: 104   Name: Jack     Leave Balance: 18   WFH: Yes   Grade: 2
ID: 105   Name: Jane     Leave Balance: 16   WFH: No    Grade: 1
```

(8 marks)

b) Implement the Company class.

- There is one class variable: _SAFE_MANAGEMENT_PERCENTAGE set to 50.0, indicating 50% of employees should be working from home.
- There are three instance variables: _name (string), _uniqueEntityNumber (string) and _departments (list).
- Constructor initialises these 2 variables: _name and_uniqueEntityNumber. The instance variable _ departments is set to an empty list.
- These 2 class methods are to retrieve and set the Safe Management Percentage respectively: getSafeManagementPercentage and setSafeManagementPercentage
- The searchDepartment method uses the department name parameter to search through the list of departments and returns the Department object with the matching name. If not found, it returns None.

- The addDepartment method has a Department object as parameter and adds it into the _departments list if this department is not present in the list. The method returns True if the department is added successfully, and False otherwise.
- The __str__ method returns a string as follows:

```
Company: SUSS      UEN: EDU1002334
Department: IT Helpdesk   Essential Services: Yes
Manager ID: 106   Name: Tom      Leave Balance: 25   WFH: No    Grade: 4
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No    Grade: 3
ID: 102   Name: Jim      Leave Balance: 22   WFH: Yes   Grade: 4
ID: 103   Name: Joe      Leave Balance: 10   WFH: No    Hours/Week: 20
ID: 104   Name: Jack     Leave Balance: 18   WFH: Yes   Grade: 2
ID: 105   Name: Jane     Leave Balance: 16   WFH: No    Grade: 1
No. of Employees working from home: 2 (33.3%) - exempted.
Department: Marketing     Essential Services: No
Manager ID: 201   Name: Neil     Leave Balance: 25   WFH: No    Grade: 4
ID: 205   Name: Charles  Leave Balance: 18   WFH: No    Grade: 4
ID: 204   Name: Darren   Leave Balance: 12   WFH: Yes   Hours/Week: 32
ID: 203   Name: Elliot   Leave Balance: 20   WFH: No    Grade: 3
ID: 202   Name: Fred     Leave Balance: 5    WFH: Yes   Hours/Week: 10
No. of Employees working from home: 2 (40.0%) - failed requirement.
```

(6 marks)

c) Write a main() function for the following:

i) Create Department objects and add Employee objects into the departments with these details.

Department	Essential Services	Manager's Employee ID
IT Helpdesk	True	106
Marketing	False	201

Employee ID	Name	Work-From-Home	Employee Type	Grade	Hours Per Week	Department
101	Jeff	False	Full Time	3		IT Helpdesk
102	Jim	True	Full Time	4		IT Helpdesk
103	Joe	False	Part Time		20	IT Helpdesk
104	Jack	True	Full Time	2		IT Helpdesk
105	Jane	False	Full Time	1		IT Helpdesk
106	Tom	False	Manager	4		IT Helpdesk
201	Neil	False	Manager	4		Marketing
205	Charles	False	Full Time	4		Marketing
204	Darren	True	Part Time		32	Marketing
203	Elliot	False	Full Time	3		Marketing
202	Fred	True	Part Time		10	Marketing

ii) Create a Company object, and then add the Department objects created in (i) into the company. Invoke the __str__ method of the company. Your results should look like the following:

```
Company: SUSS      UEN: EDU1002334
Department: IT Helpdesk   Essential Services: Yes
Manager ID: 106   Name: Tom      Leave Balance: 25   WFH: No    Grade: 4
ID: 101   Name: Jeff     Leave Balance: 20   WFH: No    Grade: 3
ID: 102   Name: Jim      Leave Balance: 22   WFH: Yes   Grade: 4
ID: 103   Name: Joe      Leave Balance: 10   WFH: No    Hours/Week: 20
ID: 104   Name: Jack     Leave Balance: 18   WFH: Yes   Grade: 2
ID: 105   Name: Jane     Leave Balance: 16   WFH: No    Grade: 1
No. of Employees working from home: 2 (33.3%) - exempted.
Department: Marketing     Essential Services: No
Manager ID: 201   Name: Neil     Leave Balance: 25   WFH: No    Grade: 4
ID: 205   Name: Charles  Leave Balance: 18   WFH: No    Grade: 4
ID: 204   Name: Darren   Leave Balance: 12   WFH: Yes   Hours/Week: 32
ID: 203   Name: Elliot   Leave Balance: 20   WFH: No    Grade: 3
ID: 202   Name: Fred     Leave Balance: 5    WFH: Yes   Hours/Week: 10
No. of Employees working from home: 2 (40.0%) - failed requirement.
```

iii) Update the Company class variable _SAFE_MANAGEMENT_PERCENTAGE to 40. Invoke the __str__ method of the company. Showcase any differences in your output.

(4 marks)

Question 3 (40 marks)

Question 3 is a continuation of Question 2. Refer to the class diagram in Figure 3.

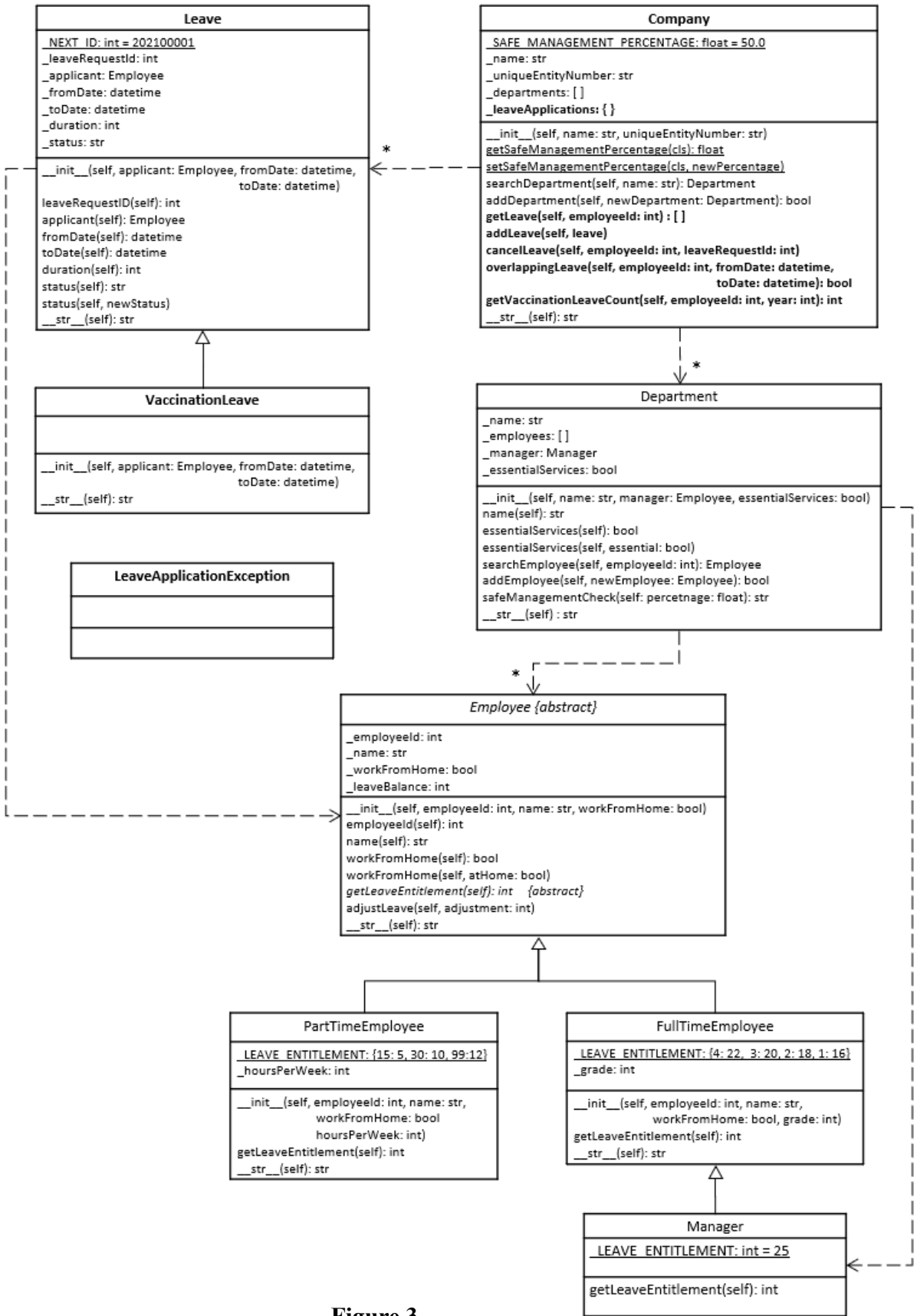


Figure 3

After recording the work-from-home (WFH) status of each employee, the company also decided to support the vaccination programme by Ministry of Health. When employees go for vaccinations, they can apply up to 2 vaccination leave days in a calendar year. The vaccination leave is a paid time off to allow employees to rest for the day when they go for Covid-19 vaccination. Furthermore, the system needs to enhance to truly reflect the WFH status of the employee, especially when the employee is on leave.

You are to implement the new classes (Leave, VaccinationLeave, LeaveApplicationException) and highlighted changes in Company class shown in Figure 3. The main() function is a menu-driven application to create a Company, populate it department and employees and test out the classes implemented.

a) Implement a subclass, LeaveApplicationException of the Exception class. This class has no additional attribute or method. When the application encounters a business rule violation, an exception from this class is raised.

(1 mark)

b) Implement the Leave and VaccinationLeave classes, that represent leave requests from an employee.

The following are requirements for the Leave class:

- It has a class variable: _NEXT_ID used for generating running numbers, typically in this format YYYY99999. In Figure 3, this class variable is set to 202100001.
- There are 6 instance variables:
  - _leaveRequestId: a unique number for this leave request, using the class variable _NEXT_ID to generate running numbers.
  - _applicant: the employee requesting this leave.
  - _fromDate: start date of the leave period. Use datetime.
  - _toDate: end date of the leave period. Use datetime.
  - _duration: number of days for this leave period. (For this TMA, no ½ day leave)
  - _status: Approved or Cancelled.

- The constructor initialises these 3 instance variables: _ applicant, _fromDate and toDate using the given parameters. The constructor validates the leave request and raise LeaveApplicationException if the following conditions are not met:
  - _fromdate is before, or same as _toDate.
  - _fromDate does not fall on Saturday or Sunday. (For this TMA, ignore public holidays)
  - _duration is computed and there is enough leave balance for this request. Saturdays and Sundays are excluded in the computation.

  If no exception is raised, the Leave object is created with _status set to "Approved".

- Getter methods for all instance variables. Use the property decorator.
- Setter method for _status. Use the setter decorator.

- The __str__ method returns a string representation of a Leave object. Here is an example in the required format:

```
Leave Request ID: 202100001
ID: 101    Name: Jeff
From: 30 Jun 2021  to  05 Jul 2021
Duration: 4 days
Status: Approved
```

(7 marks)

The VaccinationLeave class is a subclass of Leave. As it is a paid time off to employees, vaccination leave will not deduct from employee's leave balance.

- There is no additional instance variable.
- In the constructor of VaccinationLeave, there are new set of rules to adhere to. Raise LeaveApplicationException if the following conditions are not met:
  - _fromdate must be same as _toDate.
  - _fromDate or _toDate are from 30 Dec 2020 onwards.
  - _fromDate does not fall on Saturday or Sunday. (For this TMA, ignore public holidays)

  If no exception is raised, the VaccinationLeave object is created with _duration set to 0 and _status set to "Approved".

- The __str__ method returns a string representation of a VaccinationLeave object. An example is shown below:

```
Leave Request ID: 202100001
ID: 101   Name: Jeff
From: 23 Jun 2021  to  23 Jul 2021
Duration: 0 day (vaccination)
Status: Approved
```

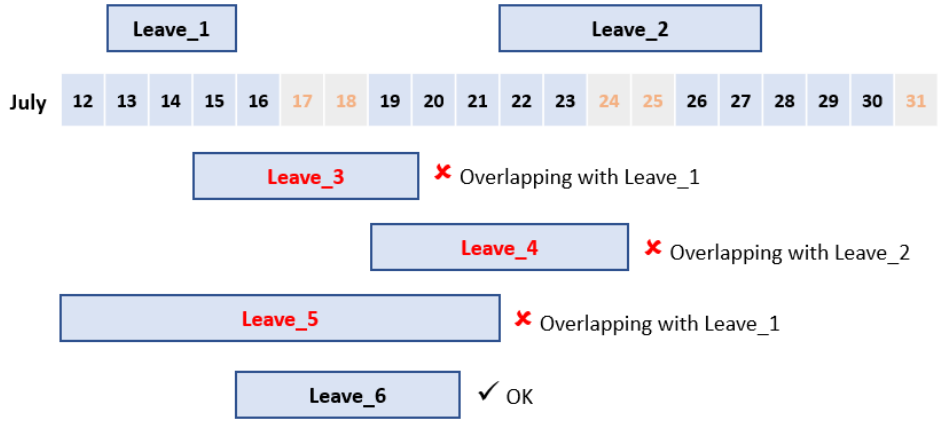
(3 marks)

c) Enhance the Company class with the following:

- There is one additional instance variable: _leaveApplications, which is a dictionary. In this dictionary, use the employee Id as the key and the value will be a list of Leave objects for this employee.
- Constructor will set _leaveApplications to an empty dictionary. The leaveApplications dictionary will have employeeId as key and a list of leave objects as value. Example: { 101: [leaveObject1, leaveObject2, …] }

There are 5 new methods to implement for Company class:

- The getLeave method returns a list of leave objects for the given employeeId. An empty list will be return if this employee has no leave request.
- addLeave method has a Leave object as parameter. For this leave request, there is a need to check if there is any overlapping of dates with existing approved leaves.
  - If there is overlapping, the method raises LeaveApplicationException with message stating there is overlapping with approved leaves. (See illustration below)



  - If no overlapping, the following actions are to be performed before the method returns:
    - Leave object is added into leaveApplications dictionary.
    - Deduct the duration from applicant's leave balance.
    - If the fromDate and toDate of this leave request include today, set the workFromHome to True for this applicant.

- cancelLeave method has 2 parameters: employeeId (int) and leaveRequestId (int). This method searches the dictionary to retrieve the list of leave objects for the given employeeId, and perform the following:
  - If this employee has no leave requests, the method raises a LeaveApplicationException with an appropriate message.
  - If there is no matching leave object with the requestId, the method raises a LeaveApplicationException with an appropriate message.
  - If there is an approved leave object matching the requestId, this leave status is set to "Cancelled", the duration added back to the applicant's leave balance before the method returns.

- Implement the overlappingLeave method. Given the employeeId, fromDate and toDate as parameters, the method searches the _leaveApplications for approved leave requests for the given employeeId. The method returns True if the fromDate and toDate have any overlapping with existing leave requests. The method returns False otherwise.

- Implement the getVaccinationLeaveCount method. Given the employeeId and the year as parameters, the method returns the number of approved vaccination leaves matching the employeeId for that year.

(12 marks)

d) Write an application that creates a Company object, populates it with Department, Employee and Leave objects using the data provided below, before presenting a menu to the users.

i) The application provides the following menu options.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option:
```

Before presenting the above menu, your application needs to create a Company object and populate it with Department and Employee objects as shown in Q2c part (i). In addition, create Leave objects to add into the Company object using the table below:

Employee ID	From Date	To Date	Vaccination?
101	30/6/2021	5/7/2021	N
101	15/7/2021	19/7/2021	N
103	29/6/2021	6/7/2021	N
104	30/6/2021	30/6/2021	Y
105	30/6/2021	5/7/2021	N
105	7/7/2021	22/7/2021	N
106	30/6/2021	30/6/2021	Y
106	30/7/2021	30/7/2021	Y
201	30/6/2021	5/7/2021	N
201	6/7/2021	6/7/2021	Y
205	30/6/2021	5/7/2021	N
205	30/7/2021	30/7/2021	Y
204	30/6/2021	5/7/2021	N
204	7/7/2021	15/7/2021	N
203	30/6/2021	5/7/2021	N
203	9/7/2021	13/7/2021	N
202	5/7/2021	8/7/2021	N
202	13/7/2021	13/7/2021	N

(3 marks)

Refer to the sample runs below to understand what each option should be doing. Your application must handle all exceptions including input error. Type/Value error should be handled by allowing the user to re-enter. Simply handle any raised LeaveApplicationException by printing the error message.

ii) Apply Leave

- This option validates the employee Id and department name entered by the user.
- The from-date and to-date must also be in correct date format.

- Depending on Vaccination (Y/N), the appropriate Leave object is created.
- If Vaccination is yes, ensure that this employee has not applied more than 2 vaccination leaves in the same calendar year.
- If no exception, ensure the created Leave object is added into the Company's dictionary of leave applications.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 102
Enter employee's department: IT Workshop
No matching department, please re-try

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 108
Enter employee's department: IT Helpdesk
No such employee, please re-try

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 101
Enter employee's department: IT Helpdesk
Enter from-date in dd/mm/yyyy: 6/30/2021
6/30/2021 is not in the format dd/mm/yyyy
Enter from-date in dd/mm/yyyy: 30/6/2021
Enter to-date in dd/mm/yyyy: 30/6/2021
Vaccination leave? (Y/N): y
Leave request should not overlap with approved leaves

Menu
======
1. Apply Leave
```

```
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 104
Enter employee's department: IT Helpdesk
Enter from-date in dd/mm/yyyy: 13/8/2021
Enter to-date in dd/mm/yyyy: 13/8/2021
Vaccination leave? (Y/N): Y
Leave Request added!!
Leave Request ID: 202100011
ID: 104    Name: Jack
From: 13 Aug 2021  to  13 Aug 2021
Duration: 0 day (vaccination)
Status: Approved

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 101
Enter employee's department: IT Helpdesk
Enter from-date in dd/mm/yyyy: 14/8/2021
Enter to-date in dd/mm/yyyy: 14/8/2021
Vaccination leave? (Y/N): Y
Leave request should not have from-date on weekend

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 1

Enter employee ID: 106
Enter employee's department: IT Helpdesk
Enter from-date in dd/mm/yyyy: 27/8/2021
Enter to-date in dd/mm/yyyy: 27/8/2021
Vaccination leave? (Y/N): y
Not allow to apply more than 2 vaccination leaves within same year
```

(4 marks)

iii) Cancel Leave

- To cancel leave request, user need to provide the employee Id and leave request Id.
- Invoke the appropriate method of the Company class to cancel the leave request.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 2

Enter employee ID: 101
Enter leave request ID to cancel: 202100033
Leave request 202100033 not found for this employee

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 2

Enter employee ID: 102
Enter leave request ID to cancel: 202100001
No leave requests for this employee

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 2

Enter employee ID: 101
Enter leave request ID to cancel: 202100001
Leave request 202100001 cancelled successfully

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
```

```
Enter option: 2

Enter employee ID: 101
Enter leave request ID to cancel: 202100001
Leave request 202100001 not found for this employee
```

(3 marks)

iv) Display Employee Leave Profile

- Validate the employee Id and department name entered by the user, before displaying the employee details and all his/her leave requests.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 3

Enter employee ID: 166
Enter employee's department: IT Helpdesk
No such employee, please re-try

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 3
Enter employee ID: 106
Enter employee's department: IT Helpdesk

ID: 106   Name: Tom      Leave Balance: 25   WFH: No    Grade: 4
Leave Request ID: 202100006
ID: 106    Name: Tom
From: 30 Jun 2021  to  05 Jul 2021
Duration: 4 days
Status: Cancelled

Leave Request ID: 202100005
ID: 106    Name: Tom
From: 16 Jul 2021  to  16 Jul 2021
Duration: 0 day (vaccination)
Status: Approved
```

(2 marks)

v) Daily Movement Update

- Validate the employee Id and department name entered by the user, before displaying the current work from home status (True/False).
- Ask if the user want to change the status. If yes, toggle the workFromHome value for this employee.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 4

Enter employee ID: 103
Enter employee's department: IT Helpdesk
Current work from home status is False
Change the status? (Y/N): Y

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 4

Enter employee ID: 103
Enter employee's department: IT Helpdesk
Current work from home status is True
Change the status? (Y/N): N
```

(2 marks)

vi) Update Safe Management Measure Percentage

- This option displays the current Safe Management Measure % to the user.
- Validate the user input for the new % (0 to 100).
- Print out the adjusted % as confirmation.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 5

Current Safe Management Measure % is 50.0
Enter new Safe Management Measure %: 45
```

```
Safe Management Measure % updated to 45.0

Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 5

Current Safe Management Measure % is 45.0
Enter new Safe Management Measure %: 999
Sorry, please re-enter within range (0, 100)
```

(2 marks)

vii) Display Departments' SMM status

- This option prints out the current Safe Management Measure status for all departments of the company.

```
Menu
======
1. Apply Leave
2. Cancel Leave
3. Display Employee Leave Profile
4. Daily Movement Update
5. Update Safe Management Measure Percentage
6. Display Departments' SMM status
0. Exit
Enter option: 6

Company: SUSS      UEN: EDU1002334
Department: IT Helpdesk   Essential Services: Yes
Manager ID: 106   Name: Tom      Leave Balance: 21   WFH: No    Grade: 4
ID: 101   Name: Jeff     Leave Balance: 13   WFH: No    Grade: 3
ID: 102   Name: Jim      Leave Balance: 22   WFH: Yes   Grade: 4
ID: 103   Name: Joe      Leave Balance: 4    WFH: No    Hours/Week: 20
ID: 104   Name: Jack     Leave Balance: 18   WFH: Yes   Grade: 2
ID: 105   Name: Jane     Leave Balance: 0    WFH: No    Grade: 1
No. of Employees working from home: 2 (33.3%) - exempted.
Department: Marketing     Essential Services: No
Manager ID: 201   Name: Neil     Leave Balance: 21   WFH: No    Grade: 4
ID: 205   Name: Charles  Leave Balance: 14   WFH: No    Grade: 4
ID: 204   Name: Darren   Leave Balance: 1    WFH: Yes   Hours/Week: 32
ID: 203   Name: Elliot   Leave Balance: 13   WFH: No    Grade: 3
ID: 202   Name: Fred     Leave Balance: 0    WFH: Yes   Hours/Week: 10
No. of Employees working from home: 2 (40.0%) - failed requirement.
```

(1 mark)

Question 4 (20 marks)

Write a Python Quiz assessment program using graphical user interface. All questions have only True or False answers. As this is only a prototype, the questions and answers are hardcoded initially using a list structure as follows:

```
questionBank = [ ['Variable names cannot start with digit', True], \
                 ["x='1'+1 is a valid statement", False], \
                 ['= and == can be used interchangeably', False], \
                 ['logical operator and has higher precedence than or', True], \
                 ['String type is immutable', True], \
                 ['x,y = y, x swaps the values of x and y', True], \
                 ['2=x is a valid statement', False], \
                 ['Variable names can be 50 letters long', True]]
```

Each value in the list is a list containing the question and the answer. The quiz consists of 4 questions randomly selected from the question bank.

a) Implement the GUI as shown in Figure 4. The title of the GUI MUST include your name.

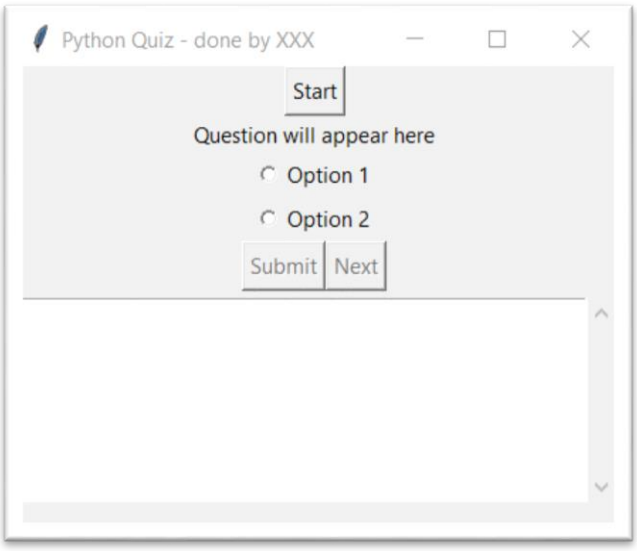
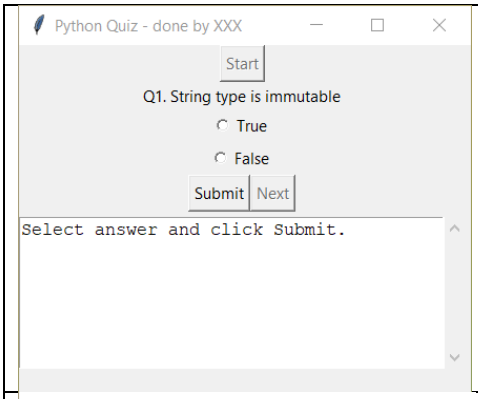
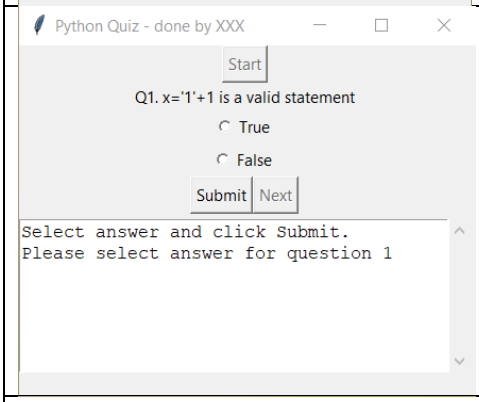
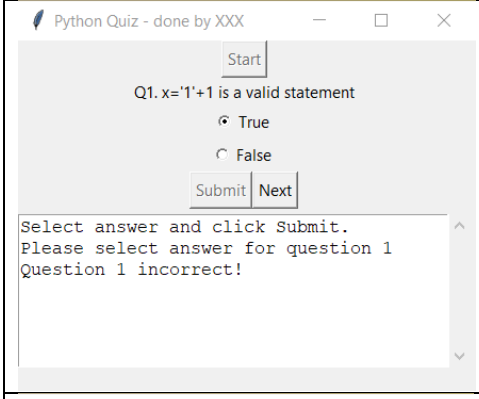
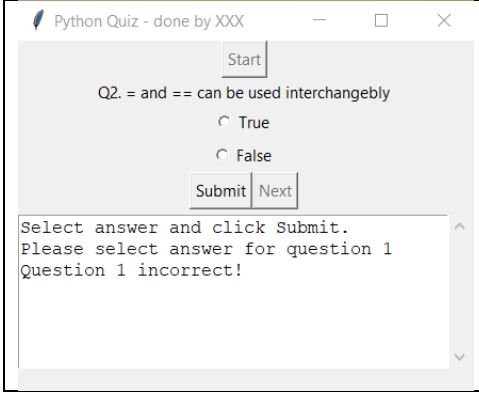


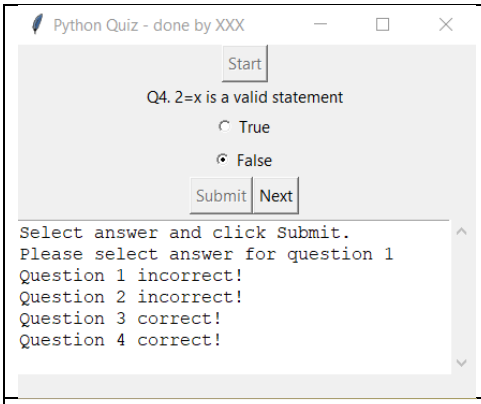
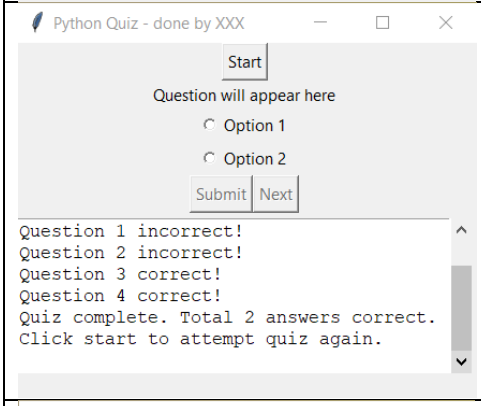
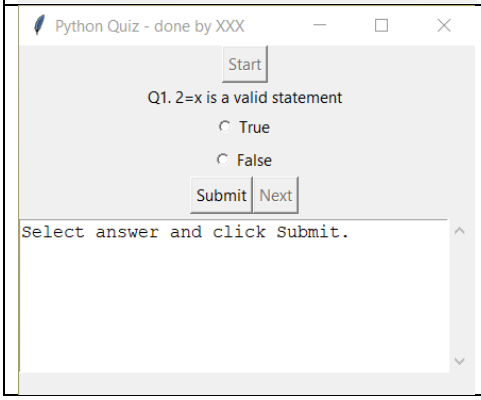
Figure 4

- Layout with name included in the title. (replace XXX with your name)
- All widgets are centered. These are the widgets:
  - 'Start' Button enabled.
  - Label which initially shows 'Question will appear here'.
  - 2 Radio Buttons, both unchecked with values 'Option 1' and 'Option 2'. The values will change to 'True' and 'False' when the quiz starts.
  - 2 Buttons 'Submit' and 'Next', both disabled.
  - Scrolled Text disabled.

(7 marks)

b) The following describes the execution of the Python Quiz assessment program. Implement event handling for the 3 buttons.

	Once the Start button is clicked, it becomes disabled. Display 'Select answer and click Submit' in the scrolled text. The label text is populated with a random question from the question bank, and the 2 options changed to True and False. The 2 radio buttons should remain unchecked. The Submit button is enabled.
	When Submit button is clicked, in the Submit button event handler, - check that at least one radio button is checked. - Otherwise, display 'Please select answer for question X'.
	- check the user attempt against the answer and display either 'Question X is correct' or 'Question X is incorrect!'. Then disable the Submit button and enable the Next button.
	When the Next button is clicked, populate the label text with the next question randomly picked from the question bank. The question must not be a repeat of the previous question. The 2 radio buttons become unchecked. The Submit button is enabled, and the Next button disabled.

	There are only 4 questions. This is the last question. The next screen shows what happens when Next button is clicked.
	When the Next button is clicked, a summary output 'Quiz complete. Total X answers correct' is displayed. The Next button is disabled. The Start button is enabled so that the user can retake the quiz again. The next screen shows what happens when Start button is clicked.
	The GUI returns back to the initial stage, with the scrolled text cleared and a new quiz started

Apply Object oriented principles when writing the GUI program. The program must be written as a class. Do not deviate from the user requirements.

Submit at least 3 set of screenshots: one for initial GUI, one set showing 1st quiz attempt with all questions answered, and one set showing 2nd quiz attempt.

(13 marks)

---- END OF ASSIGNMENT ----