**SCENARIO**:

George Wakefields is an estate agent in Kwa-Zulu-Natal and specialises in selling houses in the Durban area. Because his business is growing, he has formed a company, “Wakefields Properties”, and has taken on a few other estate agents to work for him.

Wakefield asked a programmer to write a couple of applications for the company: some for the estate agents, others for the personnel office. Unfortunately the programmer has left them and as a result the apps are unfinished.

**SECTION A: JAVA PROGRAMMING - 40**

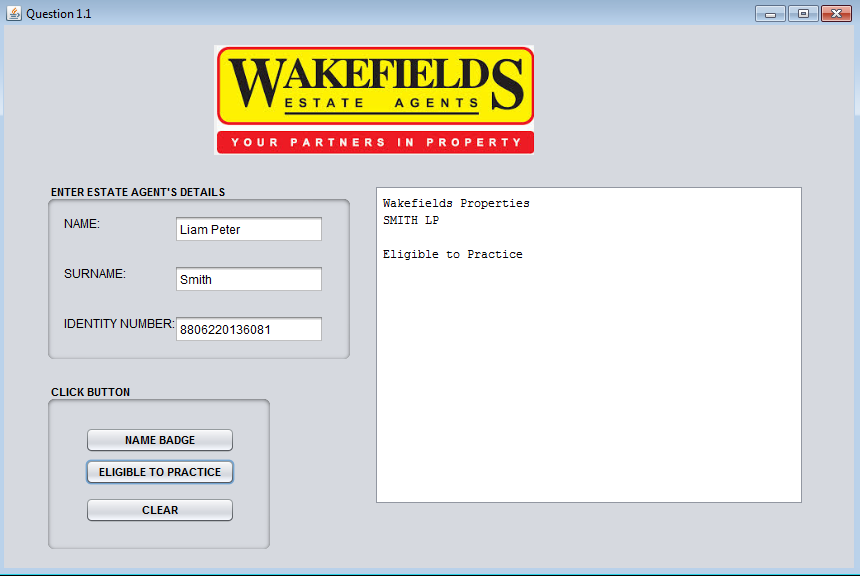
**Question 1.1: Wakefields Properties Estate Agents**

***You are provided with the GUI called Question1\_1***

****

|  |  |  |  |
| --- | --- | --- | --- |
| 1.1.1 | It is compulsory that every estate agent wears a name badge:  Enter the name and surname of an agent in the text fields provided on the user interface.  Create a badge for the agent as follows:   * Surname of the agent in capital letters. * Initials of agent name.  |  | | --- | | Wakefields Properties  <SURNAME><space><INITIALS> | | **(10)** |
| 1.1.2 | An estate agent has to be in a specific category to practice:  Enter the identity number of the agent from the text field.  An identity number must consist of 13 digits.  Validate the identity number that was entered to make sure that the length is 13 digits. If an invalid id is entered, a suitable error message must be displayed and the user must be allowed to re-enter the id number.  The program can only continue once a valid id number has been entered.  The following age categories apply:   |  |  | | --- | --- | | Female Agent | 21 - 45 | | Male Agent | 21 – 50 |   The first six digits of an identity number represent a person’s date of birth in the format <YYMMDD>, where YY refers to the year, MM refers to the month and DD refers to the day.  The next four digits in position 7 to 10 of the identity number indicate the person’s gender. The following applies:   * Male: >=5000 * Female: <5000   You are required to determine if an agent is eligible to practice. | **(15)** |

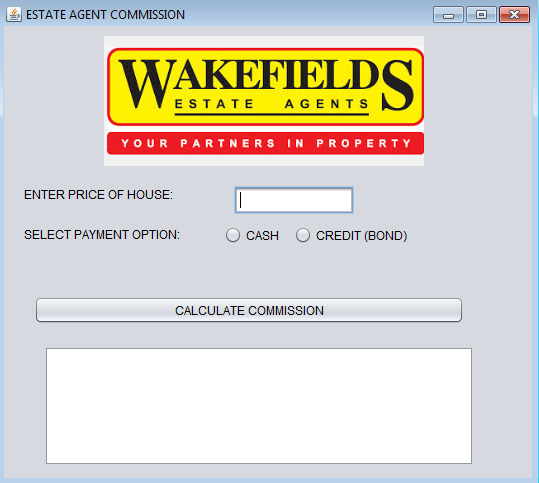
**Example of Output**

****



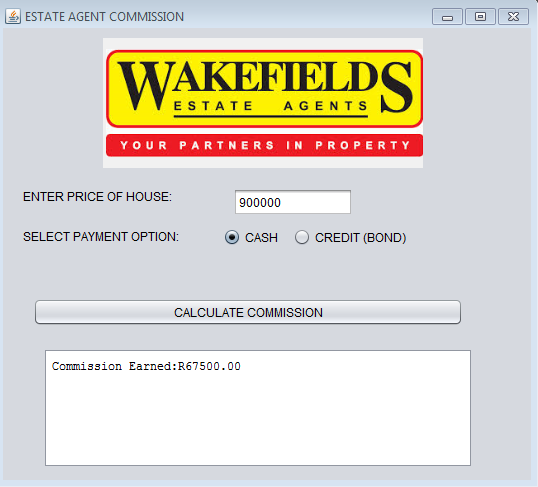
**Question 1.1.2: Wakefields Properties Commission**

***You are provided with the GUI called Question1\_2***

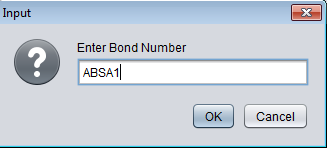
****

|  |  |  |
| --- | --- | --- |
| 1.1.3 | A commision will be earned based on the price of the house sold |  |
|  | Extract the price of the house and the buyer‘s payment option (Cash or Bond) from the respective GUI components  Ensure that a number is entered for price.  Perform the following steps if the user selects the **Cash option**:   * Calculate commission earned at a rate of 7.5% of price of house. * Display the commission earned formatted to 2 decimal places.   Perform the following steps if the user selects the **Credit(Bond**) option:   * Prompt the user to enter a 5 character bond number via a JOptionPane. * The first character of the bond number must be a character. * The last character of the bond number must be a digit. * The bond number must be validated. If the bond number is not a valid number, prompt the user to re-enter a number until a valid number is entered. * Calculate commission earned at a rate of 5% of price of house * Display the bond number and the commission earned formatted to 2 decimal places. | (15) |

**Example of Output (Cash)**

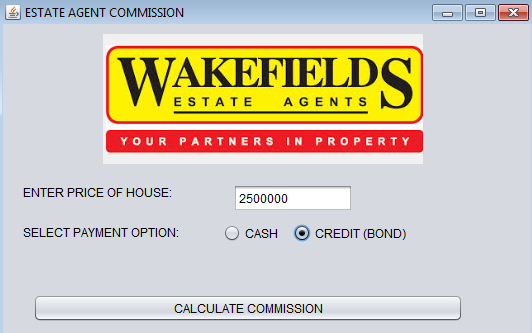
****

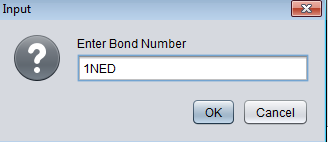
**Example of Output (Credit)**

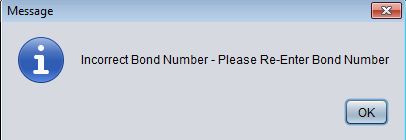
****

****

**Example of Output (Credit and Invalid Bond Number)**

****

****

****

**TOTAL : 40**

**SECTION B: : Object Oriented Programming**

**Question 2.1: OOP without Text Files - Wakefield Properties Staff Salaries**

|  |  |  |
| --- | --- | --- |
|  | Employees salaries are determined using a payroll system at Wakefields Properties. The results should be reported to the financial clerk in a text file format.  The office is divided into three main areas with different specialisation:   * Residential property – houses * Residential property – flats * Industrial property |  |
| 2.1 | You have been provided with a class named **Employee.java**.  This class has the following fields:  private double baseSalary = 6000.0  private String empName  private String empDuty  private int empExperience  private char empAreaCode  The accessor and mutator methods (GETs and SETs) have also been provided. Please add the following methods to the Employee class: |  |
| 2.1.1 | Write a ***parameterised constructor*** for the last 4 attributes, i.e. all attributes except baseSalary (that already has a value). The parameters should be used to initialise the fields of the class. | (1) |
| 2.1.2 | The area for Wakefields Properties is divided into 3 areas. A commission is given based on the number of areas (captured in the area code: A, B or C) that each employee works in.  Code A (all 3 areas) 12% commission  Code B (2 areas) 7% commission  Code C (just 1 area) 4% commission  Any other code 0% commission  Write a method called ***getCommissionPercentage()*** that will determine and return the commission percentage based on the code that has been captured – A, B or C. | (8) |
| 2.1.3 | On top of the commission the employees do receive a bonus based on experience. The bonus is calculated by taking 3% of their basic salary multiplied with the number of years of experience. Write the method ***getBonus()***that calculates and returns the bonus. | (3) |
| 2.1.4 | Write a method ***getSalary()*** that calculates and returns the salary for an employee which consists out of their base salary, their commission and their bonus. **NB**: Make use of the getCommissionPercentage() and getBonus() methods! | (3) |
| 2.1.5 | Each employee will get a code which is made up as follows:   * It consists out of the first 3 letters of his name * The first letter of his field of duty is added – an H (for houses), an F (for Flats) or an I (for Industrial Property). * Everything must be in upper case.   Write a method ***getEmployeeCode()***that returns the code according to these rules. | (5) |
| 2.1.6 | Complete the ***toString()*** method that returns the basic information on each employee arranged as follows (here with data from a test record):  Employee code: JOSH  Employee name: Josephine Jacobs  Area of duty: Selling Houses  Years of experience: 10  Salary: R 8520.00  <empty line>  **NB**: The salary must have an ‘R’ in front and must be rounded off to two decimal places. | (7) |

**Subtotal 2.1 [27]**

|  |  |  |
| --- | --- | --- |
| **2.1** | You have been provided with an application class named **Question2\_1GUI.java**. When run the GUI will look as follows: |  |
| 2.1.7 | Add code to the ***btnProcessActionPerformed()*** method to activate the ‘Process’ button.   * The method must first read the name and validate it to be at least 3 characters long. If the name field is empty or the name is only 1 or 2 characters long then a popup error message must appear and the user be asked to enter a valid name. Only when the name is valid must the program proceed. * The method must read the values from the 4 input components and use them to instantiate an Employee object. Using the toString() method from the Employee class the information must be displayed in the ‘Output’ text area. | (10) |

**Subtotal 2.2 [10]**

**Total [36]**

**Question 2.2: OOP with Text Files – (78) Wakefields Properties Staff Bonuses**

Wakefields Properties wants to give increases to their sole mandate estate agents who do perform really well as well as keep track of the salary allocation every month. They called in a personnel assessor who rated the staff on the following criteria:

* Communication
* Group Work
* Organisational Skills
* Work Ethics

Each rating is out of 5. The four ratings together make a score out of 20 and can also be expressed as a percentage.

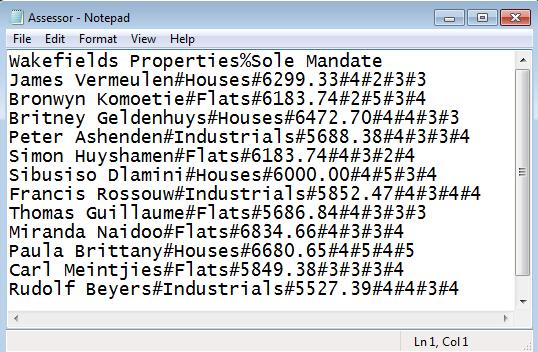
The assessor has sent his results through in a text file called ***Assessor.txt***. It is your task to process the data in this text file and produce a report for Wakefields Properties.

This is what the text file looks like:

**Line 1:** Name of estate agency%Category of estate agents

**Remaining Lines:** Employee assessment information separated by hash (#)

name of employee#occupation type#salary#communication rating#group work rating#organisational skills rating#work ethics rating



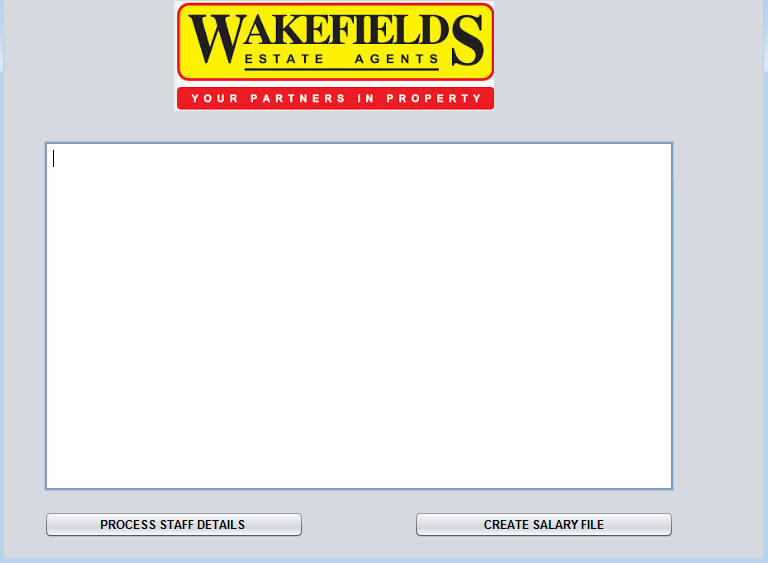
The class diagram below lists the attributes and the methods that are required.

|  |
| --- |
| StaffAssessment |
| Attributes   * agencyName * category * occupationFlat * occupationHouse * occupationIndustrial * totalSalaries |
| Methods  + Constructor(name of estate agency, month of assessment)  + getOccupationFlat()  + getOccupationHouse()  + getOccupationIndustrial()  + getTotalSalaries(); |

You have been provided with a class named StaffAssessment.java.

|  |  |  |
| --- | --- | --- |
| 2.2.1 | Complete the code for the parameterised constructor method. Assign the private fields to the parameters provided. | (1) |
| 2.2.2 | Complete the method setOccupationFlat() that will increase the number of agents selling flats by 1 | (1) |
| 2.2.3 | Complete the method setOccupationHouse() that will increase the number of agents selling houses by 1 | (1) |
| 2.2.4 | Complete the method setOccupationIndustrial() that will increase the number of agents selling industries by 1 | (1) |
| 2.2.5 | Complete the method setTotal(double salary) that receives a parameter and increases the attribute by the parameter. | (1) |
| 2.2.6 | Complete the toString() method to return the attributes as follows:  Estate Agency: <name of agency>  Estate Agents Selling Flats: <number of agents selling flats>  Estate Agents Selling Houses: <number of agents selling houses>  Estate Agents Selling Industrial Parks: <number of agents selling industrial parks>  Total Salaries: <R><total salaries including bonuses> | (7) |

**You are provided with a GUI called Question2\_2 GUI**

****

|  |  |  |
| --- | --- | --- |
| 2.2.7 | Create an object of the Assessor class.  Add code to the ***btnReadAssessment()***that reads the information from the ***Assessor.txt*** file processes one line at a time.  Test if the text file ***Assessor.txt*** exists. If the file does not exist, a suitable message must be displayed and the program must terminate.  If the file exists do the following:   * Read the first line from the text file and extract the following fields: * Estate Agency Name * Category Name   Instantiate the Assessor object.   * Loop through the file to read in a line of text from the text file. * Separate the text into the name, field of occupation, salary and the 4 ratings by the assessor. Thereafter, do the following:  1. Call one of the following methods **setOccupationFlat(), setOccupationHouse() or setOccupationIndustrial()** to increase the respective field of occupation. 2. Call the setTotal(double salary) method to increase the total salaries paid for the month. 3. Use the ratings to do the following:  * Add 4 ratings to produce a score. Express this score as a percentage. * If the score is above 70% the employee is granted a performance bonus of 5% of his salary. Total salaries must be adjusted. * Display the details all estate agents according to the sample output below:      1. Create a text file using the estate agency name as the name of the text file.   **Do not hardcode the name as the text file name may change**  Use the relevent method to store the data about the sole mandate  Estate agents in the text file in the format as shown below | (30) |

**(5)**

**2.2 Total: 42**

**TOTAL : 78**

**SECTION C: TWO DIMENSIONAL ARRAYS**

**Question 3: Wakefields Properties Sales Statistics**

****

The CEO of Wakefields Properties evaluates sales in the different accredited areas each half yearly.

Wakefield Properties is accredited with the following areas:

* Ballito
* La Lucia
* Umhlanga
* Izinga Ridge
* Morning Side

|  |  |  |
| --- | --- | --- |
| 3.1 | Create an array called arrSales that has 5 rows and 7 columns (5 x 7) | (2) |
| 3.2 | **Generate Sales**  When the user selects this menu option, the program must randomly generate sales amounts in the range 900 000 and 25 00 000 for the 5 areas and 6 months and the values must be stored in array **arrSales**. | (4) |
| 3.3 | **View Sales Statistics**  When the user selects this menu option, the program must display the sales statistics with headings as indicated in the sample output below.  *Note: The display will differ for each run of the*  *program as the data is randomly generated.* | (7) |
| 3.4 | **Swap Month 2 and Month 3**  The values for February and March were incorrectly captured. When the user selects this menu option the program must:   * Swap the sales amount for February and March * Activate the “View Staff Statistics” button to view the refreshed statistics. | (4) |
| 3.5 | **Total Sales**  When the user selects this menu option, the program must calculate the total sales for each area and store the result in the last column **of arrSales**. | (5) |
| 3.6 | **Average Sales**  When the user selects this menu option, the program must calculate the average sales for each month. The result must be stored in a one dimensional array. | (5) |
| 3.7 | **Unpopular Areas**  When the user selects this menu option the program must determine and output the unpopular area/areas.  *An unpopular area is the area with the lowest sales.*  *Note: If more than one area has the lowest total sales, the each area must be displayed.* | (5) |

**TOTAL : 32**

**FINAL TOTAL: 150**