|  |  |  |
| --- | --- | --- |
| **LAB211 Assignment** | **Type:** | **Short Assignment** |
| **Code:** | **J1.S.P0056** |
| **LOC:** | **70** |
| **Slot(s):** | **1** |

**Title**

Program to manage worker information.

**Background**

N/A

**Program Specifications**

Create a program to manage worker:

1. Add a Worker.
2. Increase salary for worker.
3. Decrease for worker.
4. Show adjusted salary worker information.

***Function details:***

**Function 1:** Display a menu and ask users to select an option.

* Users run the program. The program prompts users to select an option.
* Users select an option, perform **Function** **2**.

**Function 2:** Perform function based on the selected option.

* Option 1: Add an worker
  + Prompt user to input task information (id, name,age, salary, work location)
  + Check data input is valid with following information:
    - Code(id) cannot be null or duplicated with existed Code in DB.
    - Age must be in range 18 to 50
    - Salary must be greater than 0
  + Add Worker to DB.
  + Return to main screen.
* Option 2: Increase salary
  + Prompt user to input Code(id) and amount of money to raise
  + Data must be valid with following conditions
    - Code(id) must be existed in DB.
    - Amount of money must be > 0
  + Add salary to worker and save salary history
  + Return to main screen
* Option 3: Decrease salary
  + Prompt user to input Code(id) and amount of money to cut.
  + Data must be valid with following conditions
    - Code(id) must be existed in DB.
    - Amount of money must be > 0
  + Substract salary to worker and save salary history
  + Return to main screen
* Option 4: Show all worker have been adjusted salary by worker code.
* Option 5: Quit program.

***Expectation of User interface:***

======== Worker Management =========

1. Add Worker
2. Up salary
3. Down salary
4. Display Information salary
5. Exit

--------- Add Worker ----------

Enter Code:

Enter Name:  
Enter Age:

Enter Salary:   
Enter work location:

1

------- Up/Down Salary --------

Enter Code:

Enter Salary:

--------------------Display Information Salary-----------------------

Code Name Age Salary Status Date

W 1 Nghia 20 1100 UP 23/06/2015

W 1 Nghia 20 1500 UP 23/07/2015

W 3 Lien 20 1300 DOWN 23/07/2015

2

4

3

5

**Guidelines**

**Student must implement methods**

* addWorker
* changeSalary
* getInfomationSalary

**in startup code.**

**Example:**

Class Management contains functions add, show, increase, decrease salary of workers.

**Option** **1:** Add worker

* Named function: public boolean addWorker(Worker worker) throws Exception
  + Input:
* worker: worker information.
  + Return values:
* Worker added status.
* Exceptions list.

**Option** **2 & Option** **3:** Adjust salary.

* Named function: public boolean changeSalary(SalaryStatus status, String code, double amount)
  + Input:
* status: is increase or decrease.
* code: code Worker
* amount: amount of money
  + Return values:
* Status of adjusted.
* Exception list.

**Option** **4:** Display the list of adjusted salary workers.

* Named functions: public List<SalaryHistory> getInfomationSalary()
  + Input:
  + Return value: List of worker sort by id.