package humanResources;

import java.util.\*;

public class HumanResources {

private static Scanner in = new Scanner(System.in);

private static int intCheck() { //validate integer inputs

int input = 0;

boolean check = false;

while(!check) {

if(in.hasNextInt()) {

input = in.nextInt();

check = true;

return input;

} else {

System.err.print("Please enter an integer: ");

check = false;

}

in.next();

}

return input;

}

private static int check(int n) { //validate choosing function

while(n < 0 || n > 7) {

System.err.print("Please enter an integer in range 0 - 7: ");

n = intCheck();

}

return n;

}

private static int checkStaffType(int n) { //validate choice to add an employee or a manager

while(n < 0 || n > 1) {

System.err.print("Please enter 0 or 1 ");

n = intCheck();

}

return n;

}

private static double doubleCheck() { //validate decimal inputs

double input = 0;

boolean check = false;

while(!check) {

if(in.hasNextDouble()) {

input = in.nextDouble();

check = true;

return input;

} else {

System.err.print("Please enter a decimal: ");

check = false;

}

in.next();

}

return input;

}

private static int checkDepartmentID(int n) { //validate choice to add an employee or a manager

while(n < 1 || n > 3) {

System.err.print("Please enter 1, 2 or 3 ");

n = intCheck();

}

return n;

}

private static String checkString() { //validate string not to be empty

String s = in.nextLine().toUpperCase();

while(s.trim().equals("")) {

System.err.print("Must not be empty, please enter information: ");

s = in.nextLine().toUpperCase();

}

return s;

}

private static boolean staffID(String s) { //validate unique staff's ID

for(Staff stf : staff) {

if(stf.getId().equalsIgnoreCase(s)) {

return false;

}

}

return true;

}

private static int compare(ICalculator a1, ICalculator a2) { //sorting by ascending order

return (int)(a1.calculateSalary() - a2.calculateSalary());

}

private static int compareTo(ICalculator a1, ICalculator a2) { //sorting by alphabet

return a1.getId().compareTo(a2.getId());

}

static List<Employee> employee = new ArrayList<>(); //List of Employee

static List<Manager> manager = new ArrayList<>(); //List of Manager

static List<Department> department = new ArrayList<>(); //List of departments

static List<Staff> staff = new ArrayList<>(); //List of all staffs

public static void main(String[] args) {

// TODO Auto-generated method stub

start();

boolean choose = true;

while(choose) {

//menu

System.out.println("Welcome to the human resource management application");

System.out.println("------------------------MENU------------------------");

System.out.println("1. Display the list of staffs");

System.out.println("2. Display departments");

System.out.println("3. Display staffs by departments");

System.out.println("4. Add new staff");

System.out.println("5. Search staffs' information by ID");

System.out.println("6. Display the wages statements of all staffs");

System.out.println("7. Display the wages statements in ascending order of received wages");

System.out.println("0. Exit");

System.out.print("Your choice: ");

/\*\*/

int choice = intCheck();

choice = check(choice);

switch(choice) {

case 1: //display the list of staffs

allStaffs();

break;

case 2: //Display departments

allDepartments();

break;

case 3: //Display staffs by departments

staffsByDepartments();

break;

case 4: //Add new staff

addStaff();

break;

case 5: //Search staff

searchStaff();

break;

case 6: //Display wage statements

wagesStatements();

break;

case 7: //Display wage statements by ascending order

sortWagesStatements();

break;

case 0: //Exit

exit();

choose = false;

break;

}

}

}

public static void start() {

employee.add(new Employee("E1", "NGUYỄN DUY ANH", 35, 2.2, "22/12/2009", "D1", 10, 10));

employee.add(new Employee("E2", "ĐẶNG THU HÀ", 20, 1.0, "1/8/2020", "D2", 2, 12));

employee.add(new Employee("E3", "ĐỖ HÙNG DŨNG", 28, 1.7, "10/1/2015", "D3", 12, 14));

//create and add managers to the list of managers

manager.add(new Manager("M1", "VŨ HÀ PHƯƠNG", 47, 5.5, "20/5/2005", "D1", 3, "BUSINESS LEADER"));

manager.add(new Manager("M2", "LÊ ĐỨC ANH", 40, 5.0, "1/4/2012", "D2", 7, "PROJECT LEADER"));

manager.add(new Manager("M3", "NGUYỄN VĂN QUYẾT", 50, 5.8, "4/5/2011", "D3", 10, "TECHNICAL LEADER"));

for (Employee anEm : employee) {

staff.add(anEm);

}

for (Manager aMan : manager) {

staff.add(aMan);

}

}

public static void allStaffs() { //print out list of staffs

System.out.println("-----------------------------------------------------------------Employee's list-----------------------------------------------------------------");

System.out.println();

System.out.println(String.format("%-10s%-20s%-20s%-20s%-20s%-20s%-20s%-20s", "ID", "NAME", "AGE", "SALARY COEFFICIENT", "START DATE", "DEPARTMENT", "DAYS OFF", "OVERTIME HOURS"));

for(Employee anEm : employee) {

System.out.println(anEm.toString());

}

System.out.println();

System.out.println("-----------------------------------------------------------------Managers's list-----------------------------------------------------------------");

System.out.println();

System.out.println(String.format("%-10s%-20s%-20s%-20s%-20s%-20s%-20s%-20s", "ID", "NAME", "AGE", "SALARY COEFFICIENT", "START DATE", "DEPARTMENT", "DAYS OFF", "POSITION"));

for(Manager aMan : manager) {

System.out.println(aMan.toString());

}

System.out.println();

}

public static void allDepartments() { //display information of departments

//create and add departments to list of departments

department.add(new Department("D1", "BUSINESS", 0));

department.add(new Department("D2", "PROJECT", 0));

department.add(new Department("D3", "TECHNICAL", 0));

//count number of staffs in each department

int countD1 = 0;

int countD2 = 0;

int countD3 = 0;

for(Staff stf : staff) { //count number of employee in each department

if(stf.getDepartmentID().equalsIgnoreCase("D1")) {

countD1++;

} else if(stf.getDepartmentID().equalsIgnoreCase("D2")) {

countD2++;

} else {

countD3++;

}

}

for(Department dep : department) { //store number of staffs in each department

if(dep.getDepartmentID().equalsIgnoreCase("D1")) {

dep.setNumStaff(countD1);

} else if(dep.getDepartmentID().equalsIgnoreCase("D2")) {

dep.setNumStaff(countD2);

} else {

dep.setNumStaff(countD3);

}

}

for(Department dep : department) { //Display list of departments

System.out.println(dep.toString());

}

System.out.println();

}

public static void staffsByDepartments() {

//Display all staffs of Business

System.out.println("--------------------Business--------------------");

System.out.println(String.format("%-10s%-20s%-10s", "ID", "NAME", "AGE"));

for(Staff stf : staff) {

if(stf.getDepartmentID().equalsIgnoreCase("D1")) {

System.out.println(String.format("%-10s%-20s%-10d", stf.getId(), stf.getName(), stf.getAge()));

}

}

System.out.println();

//Display all staffs of Project

System.out.println("--------------------Project--------------------");

System.out.println(String.format("%-10s%-20s%-10s", "ID", "NAME", "AGE"));

for(Staff stf : staff) {

if(stf.getDepartmentID().equalsIgnoreCase("D2")) {

System.out.println(String.format("%-10s%-20s%-10d", stf.getId(), stf.getName(), stf.getAge()));

}

}

System.out.println();

//Display all staffs of Technical

System.out.println("--------------------Technical--------------------");

System.out.println(String.format("%-10s%-20s%-10s", "ID", "NAME", "AGE"));

for(Staff stf : staff) {

if(stf.getDepartmentID().equalsIgnoreCase("D3")) {

System.out.println(String.format("%-10s%-20s%-10d", stf.getId(), stf.getName(), stf.getAge()));

}

}

System.out.println();

}

public static void addStaff() { //Add new staff

System.out.println("Enter information of a new staff");

in.nextLine();

//ID

System.out.print("ID: ");

String id = "";

while(in.hasNext()) {

id = in.next().toUpperCase();

if(staffID(id)) {

break;

} else {

System.err.print("This ID has already existed, please create another ID ");

}

}

in.nextLine();

//Name

System.out.print("Name: ");

String name = checkString().toUpperCase();

//Age

System.out.print("Age: ");

int age = intCheck();

//Salary coefficient

System.out.print("Salary coefficient: ");

double salaryCoefficient = doubleCheck();

//Start date

System.out.print("Start date: ");

in.nextLine();

String startDate = checkString();

//Department

System.out.print("Department ID (1 = D1,2 = D2, 3 = D3): ");

int chooseDepartment = intCheck();

chooseDepartment = checkDepartmentID(chooseDepartment);

String departmentID = "";

switch(chooseDepartment) {

case 1:

departmentID = "D1";

break;

case 2:

departmentID = "D2";

break;

case 3:

departmentID = "D3";

break;

}

//Days off

System.out.print("Number of days off: ");

int daysOff = intCheck();

//choose to add employee or manager

System.out.print("Add an employee or a manager?(0 = employee, 1 = manager) ");

int staffType = intCheck();

staffType = checkStaffType(staffType);

if(staffType == 0) { //employee

System.out.print("Number of overtime hours: ");

int overtime = intCheck();

//add new employee to lists

Employee newEm = new Employee(id, name, age, salaryCoefficient, startDate, departmentID, daysOff, overtime);

employee.add(newEm);

staff.add(newEm);

} else { //manager

System.out.print("Position: ");

in.nextLine();

String position = checkString();

//add new managers to lists

Manager newMan = new Manager(id, name, age, salaryCoefficient, startDate, departmentID, daysOff, position);

manager.add(newMan);

staff.add(newMan);

}

System.out.println();

}

public static void searchStaff() {

in.nextLine();

System.out.print("Want to find? ");

String searchKey = checkString().toUpperCase();

//search staff

List<Staff> searchStaff = new ArrayList<>();

for(Staff stf : staff) {

if(stf.getId().contains(searchKey) || stf.getName().contains(searchKey)) {

searchStaff.add(stf);

}

}

/\*\*/

//Display staffs found

if(!searchStaff.isEmpty()) {

for(Staff search : searchStaff) {

if(search instanceof Employee) {

System.out.println("----------------------------------------------------------------------Employee----------------------------------------------------------------------");

System.out.println(String.format("%-10s%-20s%-20s%-20s%-20s%-20s%-20s%-20s", "ID", "NAME", "AGE", "SALARY COEFFICIENT", "START DATE", "DEPARTMENT", "DAYS OFF", "OVERTIME HOURS"));

System.out.println(((Employee)search).toString());

System.out.println();

} else {

System.out.println("----------------------------------------------------------------------Manager----------------------------------------------------------------------");

System.out.println(String.format("%-10s%-20s%-20s%-20s%-20s%-20s%-20s%-20s", "ID", "NAME", "AGE", "SALARY COEFFICIENT", "START DATE", "DEPARTMENT", "DAYS OFF", "POSITION"));

System.out.println(((Manager)search).toString());

System.out.println();

}

}

} else {

System.err.print("Do not exist!");

System.out.println();

}

}

public static void wagesStatements() { //Display wage statements

System.out.println("--------------------Employee--------------------");

System.out.println(String.format("%-10s%-20s%-15s", "ID", "NAME", "SALARY"));

employee.sort(HumanResources::compareTo);

for(Employee em : employee) {

System.out.println(String.format("%-10s%-20s%-15s", em.getId(), em.getName(), (int)em.calculateSalary()));

}

System.out.println("--------------------Managers--------------------");

System.out.println(String.format("%-10s%-20s%-15s", "ID", "NAME", "SALARY"));

manager.sort(HumanResources::compareTo);

for(Manager man : manager) {

System.out.println(String.format("%-10s%-20s%-15s", man.getId(), man.getName(), (int)man.calculateSalary()));

}

System.out.println();

}

public static void sortWagesStatements() { //Display wage statements in ascending order

System.out.println("--------------------Employee--------------------");

System.out.println(String.format("%-10s%-20s%-15s", "ID", "NAME", "SALARY"));

employee.sort(HumanResources::compare);

for(Employee em : employee) {

System.out.println(String.format("%-10s%-20s%-15s", em.getId(), em.getName(), (int)em.calculateSalary()));

}

System.out.println("--------------------Managers--------------------");

System.out.println(String.format("%-10s%-20s%-15s", "ID", "NAME", "SALARY"));

manager.sort(HumanResources::compare);

for(Manager man : manager) {

System.out.println(String.format("%-10s%-20s%-15s", man.getId(), man.getName(), (int)man.calculateSalary()));

}

System.out.println();

}

public static void exit() { //Exit program

System.err.print("========" + " " + "|| ||" + " " + "========" + " " + "========" + " " + "||\\ | " + "======" + "\n");

System.err.print(" ||" + " " + "|| ||" + " " + "||" + " " + "||" + " " + "||\\\\ | " + "|| ||" + "\n");

System.err.print(" ||" + " " + "|======|" + " " + "||===== " + " " + "||===== " + " " + "|| \\\\ | " + "|| ||" + "\n");

System.err.print(" ||" + " " + "|| ||" + " " + "||" + " " + "||" + " " + "|| \\\\ | " + "|| ||" + "\n");

System.err.print(" ||" + " " + "|| ||" + " " + "========" + " " + "========" + " " + "|| \\\\| " + "======");

}

}