Sistema Empresa XYZ

Considere-se as seguintes classes implementadas em Java, a partir de uma análise bastante simplificada do corpo funcional de uma grande empresa.

```
public class estagiario{
        private int ID;
        private String nome;
        private int departamento;
        private double salario;
        private double valecoxinha;

        public boy (int ID, String nome, int departamento, double salario, double valecoxinha){
                this.ID = ID;
                this.nome=nome;
                this.departamento= departamento;
                this.salario= salario;
                this. valecoxinha = valecoxinha;
        }
        void setValeCoxinha( double valecoxinha ){
                this. valecoxinha = valecoxinha;
        }
        double pagamento(){
                // Calcula o salário líquido
                return(this.salario+this.valecoxinha);
        }
}

public class funcionario{
        private int ID;
        private String nome;
        private int departamento;
        private double salario, previdencia, adicional;

        public funcionario(int ID, String nome, int departamento, double salario, double adicional){
                this.ID = ID;
                this.nome=nome;
                this.departamento= departamento;
                this.salario= salario;
                this.previdencia=0.05; // Desconto fixo de um sistema de previdência da empresa
                this. adicional = adicional;  // adicional caso o funcionário seja por exemplo chefe.
        }
        void setAdicional(double novo_percentual){
                this.adicional = adicional*(1+novo_percentual);
        }
        double pagamento(){
                // Calcula o salário líquido
                return(salario*(1- this.previdencia+ this.adicional));
        }
}
```

```
public class presidente{
        private int ID;
        private String nome;
        private int departamento;
        private double salario, previdencia, adicional_whisky, adicional_helicoptero,
                adicional_adicional;

        public presidente( int ID, String nome, int departamento, double salario){
                this.nome=nome;
                this.data_nascimento= data_nascimento;
                this.data_admissao= data_admissao;
                this.departamento= departamento;
                this.salario= salario;
                this.previdencia=0.05;  // Desconto fixo de um sistema de previdência da empresa
                this.adicional_whisky=0.9;
                this.adicional_helicoptero=0.7;
                this.adicional_adicional=3.8;

        }
        void setAdicional(double novo_percentual){
                adicional_whisky=0.9*(1+novo_percentual);
                adicional_helicoptero=0.7*(1+novo_percentual);
                adicional_adicional=3.8*(1+novo_percentual);
        }

        double pagamento(){
                // Calcula o salário líquido
                return(salario*(1-this.previdência+this.adicional_whisky+this.adicional_helicoptero
                                + this.adicional_adicional));
        }
}
```

1ª Parte do trabalho

Observando-se cada classe isoladamente, nota-se que cada uma representa bem um tipo de coloborador. Porém, quando observadas em conjunto, várias modificações podem ser introduzidas para se aproveitar as funcionalidades da orientação a objetos, tais como herança e polimorfismo. Assim reescreva as classes acima de forma aproveitando as funcionalidades da orientação a objetos vistas na disciplina, ou seja, defina uma classe base para armazenar os atributos e os métodos genéricas e classes especializadas para representar cada tipo de funcionário.

2ª Parte do trabalho

Após você reescrever as classes escreva a **classe Empresa** que armazena os funcionários da empresa. A **classe Empresa** utiliza um ***container*** (**coleção objetos**), ou seja, um **vetor objetos** (não pode usar ArrayList) com **no máximo 100 posições** para armazenar os objetos que representam os funcionários da empresa. Além disso a **classe Empresa** disponibiliza **métodos** que implementam as seguintes operações:

1) Adicionar um funcionário;
2) Calcular pagamento do funcionário.
3) Aumentar o adicional de todos os funcionários.
4) Relatório dos funcionários da empresa.

Para um usuário conseguir administrar os funcionários armazenados na **classe Empresa**, implemente uma **classe principal** com a função **main()**, que apresenta um menu para o usuário escolher qual das operações acima irá executar, em seguida é executado o método correspondente a operação, e disponibilizado novamente o menu para que usuário escolha uma próxima operação, e é claro, você deve disponibilizar no menu uma opção para finalizar o programa.

Na implementação dos **métodos** da **classe Empresa** as informações necessárias para a sua execução devem ser passadas por parâmetro, ou seja, os métodos realizam a entrada de ados (**Scanner**) e nem saída de dados (**System.out.println**) .

Por exemplo, se for solicitada a 1ª operação (Adicionar um funcionário) o usuário deverá informar os dados do funcionário, em seguida é criado um novo funcionário (objeto) e a **classe principal** faz a chamada do **método da classe Empresa** informando o objeto a ser inserido, que realiza inserção e atualização no *container*.

Detalhamento das operações

1) Para **adicionar** um, no **método da classe Empresa** responsável por essa operação, você pode adicionar o funcionário sempre no final do vetor.
2) Para **calcular o pagamento** do funcionário o método da classe Empresa responsável por essa operação recebe por parâmetro o ID do funcionário, em seguida busca no *container* o funcionário calcula e devolve o valor do salário do funcionário para a classe principal.
3) Para **aumentar o adicional de todos funcionário** o método da classe Empresa responsável por essa operação recebe por parâmetro o ID do funcionário e a percentual de aumento, em seguida, para todos os funcionários da empresa que tem direito ao adicional é aplicado o percentual de aumento de adicional para os funcionários.
5) Para fazer o **relatório dos funcionários da empresa** o método da classe Empresa responsável por essa operação deve devolver uma String com a lista com todos os funcionários da empresa contendo o ID, o nome do funcionário e o salário de cada um dos funcionários armazenados no *container*. Na tela deve ser impresso um funcionário por linha. Sugestão reescreva o método toString().

Restrições do projeto

1) Para armazenar os funcionários do seu sistema você deverá usar **vetor de objetos** como **container**, não é permitido usar classes prontas do Java (**Java Collections**) para isso, como **ArrayList**.

2) O programa deve estar bem documentado e implementado na **linguagem Java**, aplicando os conhecimentos sobre programação orientado a objetos e tipo abstrato de dados vistos nesse semestre.

3) Este trabalho pode ser desenvolvido em grupos de até **2 alunos**. Como este trabalho pode ser feito em grupo, evidentemente você pode "discutir"' o problema dado com outros grupos, inclusive as "dicas"' para chegar às soluções, mas você deve ser responsável pela solução final e pelo desenvolvimento do seu programa. Ou seja, qualquer tentativa de fraude será punida com a **nota zero**. Para maiores esclarecimentos leiam o documento "**Orientações para Desenvolvimento de Trabalhos Práticos**".

4) A entrega consistirá em uma apresentação do projeto ao professor responsável pela disciplina e será a avaliado de acordo com os seguintes critérios:
   a) Funcionamento do programa, ou seja, programas com erros de compilação e não executando receberão nota 0 (zero);
   b) O quão fiel é o programa quanto à descrição do enunciado;
   c) Clareza e organização, programas com código confuso (linhas longas, variáveis com nomes não-significativos, etc.) e desorganizado (sem indentação, sem comentários, etc.) também serão penalizados
   d) E principalmente a aplicação dos conceitos de orientação a objetos vistos durante as aulas da disciplina;