Universidade Federal de Campina Grande

Centro de Engenharia Elétrica e Informática – CEEI
Unidade Acadêmica de Sistemas e Computação – UASC
Disciplina: Laboratório de Programação 2

Laboratório 02

Controle Institucional da Situação Acadêmica (COISA)

Neste projeto, você deve desenvolver um sistema capaz de gerenciar o uso dos laboratórios de Ciência da Computação (LCC's) e sua vida acadêmica. O COISA se trata de um sistema complexo, logo a separação de responsabilidades através da criação de classes é de **extrema** importância, pois tem como objetivo estruturar, coerentemente, o seu programa.

A vida do aluno pode ser organizada em quatro atividades básicas: (1) organizar seu tempo de uso de internet para as disciplinas, o que é bem importante considerando o modelo remoto atual, (2) estudar para as disciplinas, (3) organizar suas finanças e (4) acompanhar sua saúde física e mental. Para permitir o controle dessas 4 atividades, você irá desenvolver um sistema que permite avaliar a quantidade de tempo de internet (online) que você tem usado nas disciplinas, a quantidade de horas que você tem estudado, como está sua situação financeira e se está conseguindo se sustentar no curso e, por fim, como está sua saúde física e mental (e por consequência, sua saúde geral).

Assim, para cada uma das atividades, é descrito um conjunto de valores referentes a cada atividade e de ações que podem ser feitas para aquela atividade.

Atividade	**Estado (dados)**	**Ações**
Registro de tempo online	Nome da disciplina Tempo gasto online (horas com a disciplina) Tempo esperado (horas para a disciplina)	Adicionar tempo online Verificar tempo online Imprimir estado
Disciplina	Nome da disciplina Horas de estudo Notas 1, 2, 3 e 4	Cadastrar horas de estudo Cadastrar uma nota Calcular média Verificar se foi aprovado Imprimir estado
Registro de finanças	Receita Despesa Fontes de renda	Adicionar despesa Adicionar receita Adicionar fonte Verificar despesas e receita Imprimir estado
Saúde	Saúde física Saúde mental	Definir estado de saúde mental Definir estado de saúde física Retorna estado geral de saúde

Para facilitar a sua vida, já existe uma classe pronta que irá ser executada para testar o funcionamento do seu programa. Por este programa é possível ver as classes que devem ser implementadas (dica: uma classe para cada atividade) bem como os métodos públicos. Você deve copiar essa classe no seu programa e fazê-la funcionar de acordo com as especificações que virão a seguir. Execute sempre esta classe para garantir que você está desenvolvendo corretamente cada atividade.

```
package lab2;

public class Coisa {

    public static void main(String[] args) {

      RegistroTempoOnline tempoLP2 = new RegistroTempoOnline("LP2", 30);
      tempoLP2.adicionaTempoOnline(10);
      System.out.println(tempoLP2.atingiuMetaTempoOnline());
      tempoLP2.adicionaTempoOnline(10);
      tempoLP2.adicionaTempoOnline(10);
      System.out.println(tempoLP2.atingiuMetaTempoOnline());
      tempoLP2.adicionaTempoOnline(2);
      System.out.println(tempoLP2.atingiuMetaTempoOnline());
      System.out.println(tempoLP2.toString());


       Disciplina prog2 = new Disciplina("PROGRAMACAO 2");
       prog2.cadastraHoras(4);
       prog2.cadastraNota(1, 5.0);
       prog2.cadastraNota(2, 6.0);
       prog2.cadastraNota(3, 7.0);
       System.out.println(prog2.aprovado());

       prog2.cadastraNota(4, 10.0);
       System.out.println(prog2.aprovado());
       System.out.println(prog2.toString());

       RegistroFinancas minhaFinanca = new RegistroFinancas(100000);
       minhaFinanca.aumentaReceita(12000, 1);
       minhaFinanca.aumentaReceita(72100, 2);
       minhaFinanca.pagaDespesa(20000);
       System.out.println(minhaFinanca.exibeFontes());
       System.out.println(minhaFinanca.toString());


       Saude saude = new Saude();
       System.out.println(saude.getStatusGeral());
       saude.defineSaudeMental("boa");
       saude.defineSaudeFisica("boa");
       System.out.println(saude.getStatusGeral());

       saude.defineSaudeMental("fraca");
       saude.defineSaudeFisica("fraca");
       System.out.println(saude.getStatusGeral());

       saude.defineSaudeMental("boa");
       saude.defineSaudeFisica("fraca");
       System.out.println(saude.getStatusGeral());
    }
}
```

Essa classe ao ser executada deve produzir a seguinte saída:

```
false
true
true
LP2 32/30
false
true
PROGRAMACAO 2 4 7.0 [5.0, 6.0, 7.0, 10.0]
1 - 112000
2 - 72100
3 - 0
4 - 0
Receita total: 184100, Receita atual: 164100, Despesas totais: 20000
boa
boa
fraca
ok
```

A seguir apresentaremos em detalhe cada funcionalidade, buscando facilitar o seu entendimento do que deve ser feito.

1. Registro de tempo online

O registro de tempo online deve ser responsável por manter a informação sobre quantidade de horas de internet que o aluno tem dedicado a uma disciplina remota. Para cada disciplina, seria criado um objeto para controle desse estado (tempo online usado).

Para uma disciplina de X horas, espera-se que o aluno dedique o dobro de tempo online para realizar tal disciplina. Por padrão, o tempo online esperado seria de 120 horas. É possível passar a quantidade de horas online esperada para a disciplina como parâmetro na construção do objeto que representa o registro de tempo online. O *toString* deste objeto deve gerar uma String que o representa contendo: o nome da disciplina, o tempo online já usado e o tempo online esperado.

Ao atingir o tempo esperado, o método de verificação de tempo online deve indicar que o usuário atingiu o tempo esperado para aquela disciplina (através de um booleano). Mesmo atingindo o tempo online esperado, o usuário ainda pode adicionar dados, ou seja, ele pode usar mais tempo online para aquela disciplina.

Para implementar esta funcionalidade, você deve criar uma classe **RegistroTempoOnline** com os construtores: **RegistroTempoOnline (String nomeDisciplina)** e **RegistroTempoOnline (String nomeDisciplina, int tempoOnlineEsperado)**. O tempo será representado em horas.

É importante também implementar os métodos abaixo:

- **void adicionaTempoOnline(int tempo)**
- **boolean atingiuMetaTempoOnline()**
- **String toString()**

2. Disciplina

Por padrão toda disciplina tem 4 notas. Calcula-se a média da disciplina, fazendo a média aritmética dessas 4 notas (sem arredondamento). Todo aluno é considerado aprovado quando atinge a média igual ou acima de 7.0. Caso alguma das notas não seja cadastrada, ela é considerada como zero. Se a alguma nota (nota 1, nota 2, nota 3 ou nota 4) for cadastrada mais de uma vez, consideramos a última nota cadastrada (ou seja, é possível repor notas nesse sistema). É possível cadastrar horas de estudo para determinada disciplina. As horas de estudo são cumulativas, ou seja, a nova quantidade cadastrada soma-se às previamente cadastradas.

O *toString()* deve gerar uma representação da disciplina que contenha o nome da disciplina, o número de horas de estudo, a média do aluno e as notas de cada prova.

Cada objeto que representa uma disciplina começa sem horas de estudo cadastradas. Para desenvolver esta funcionalidade, você deve implementar a classe **Disciplina** que tem o construtor **Disciplina(String nomeDisciplina)**. Implemente, também, os métodos:

- **void cadastraHoras(int horas)**
- **void cadastraNota(int nota, double valorNota)** // notas possíveis: 1, 2, 3 e 4
- **boolean aprovado( )**
- **String toString( )**

3. Registro de finanças

O aluno pode manter um registro para suas finanças o que lhe ajuda a controlar seus gastos durante o curso. Para tal controle o aluno deve saber qual a sua receita e seus gastos a cada momento (ambos em centavos). A receita é gerada a partir das fontes de renda do aluno que podem ser de 4 tipos: família, projetos institucionais com bolsa, auxílio institucional e doações externas. O valor da receita aumenta à medida que o aluno recebe dinheiro de suas fontes de renda, e diminui quando ocorrem os gastos. É possível informar o histórico de quanto foi recebido de cada tipo de fonte, independente dos gastos.

Por fim, o usuário pode gerar uma string representando o histórico de sua conta. O método *toString(),* comunica: o valor da receita atual, o valor das despesas acumulado e o valor da receita acumulado. Observe que este método apresenta o valor da receita acumulado independentemente das despesas ocorridas.

Para controlar as suas finanças, você deve criar a classe **RegistroFinancas** com o construtor **RegistroFinancas(int receitaInicialDoTipo1)**. Esta classe deve ter os métodos:

- **void aumentaReceita(int valorCentavos, int tipoFonte) //tipo fonte pode ser 1, 2, 3 e 4**
- **void pagaDespesa(int valorCentavos)**
- **String exibeFontes( )**
- **String toString( )**

4. Sua saúde física e mental

Por fim, é importante acompanhar o estado de saúde geral do aluno. Todo aluno tem uma saúde física e mental. Cada uma delas pode estar boa ou fraca. Caso ambas estejam fracas, a saúde geral do aluno é “fraca”. Se ambas estiverem boas, a saúde geral do aluno é “boa”. Se uma delas estiver fraca, mas a outra estiver boa, a saúde geral do aluno é considerada “ok”.
Faça uma classe **Saude** que tenha um construtor inicial sem parâmetros e que irá definir a saúde mental e a saúde física do aluno como “boa”. Essa classe precisa ter 3 métodos:

- **void defineSaudeMental(String valor)**
- **void defineSaudeFisica(String valor)**
- **String getStatusGeral()**

5. Bônus!

Ao terminar as atividades propostas anteriormente, você pode tentar completar as tarefas descritas a seguir.

5.1 - Mais Notas na Disciplina

Na classe Disciplina, crie um construtor adicional que recebe também o número de notas da disciplina. Além desse construtor, crie outro construtor que recebe o número de notas e um array de inteiros com os pesos de cada uma das notas, para o cálculo de uma média ponderada.

Por exemplo, uma disciplina de 2 notas e pesos [6, 4] tem sua média calculada como (6 * Nota_1 + 4 * Nota_2) / 10. Caso o array de pesos não seja passado, considere cada nota com mesmo peso (ou seja, a média é uma média aritmética: (Nota_1 + Nota_2)/2).

5.2 - Registro Detalhado de Despesas

No registro de finanças, além de registrar o valor das despesas, você pode registrar um pequeno texto descrevendo esta despesa. Por exemplo, “Aluguel de março 2021”, ou, “Cento de salgados para happy hour com as garotas”. Este registro é opcional. Além disso você deve criar um método que retorne os últimos 5 detalhes cadastrados.

Ou seja, você deve implementar, além dos métodos básicos de RegistroFinancas, os métodos:

- **void pagaDespesa(int valorCentavos, String detalhes)**
- **String listarDetalhes( )** // deve retornar uma string com os últimos 5 detalhes, um em cada linha

5.3 - Nível de Saúde

Além da saúde mental e física, o aluno pode cadastrar um emoji que descreva sua última sensação em geral. Por exemplo, há certos dias que ele pode estar “**:(**”, “*_*”, “:o)”,“<(^_^<)”, “¯_(ツ)_/¯” ...

O emoji, não tem relação com o cálculo da saúde geral do aluno, que é baseado na saúde física e mental. Ele expressa o sentimento do aluno no momento.

Caso seja registrado um emoji, ele deve ser retornado como adicional ao estado de saúde geral do aluno. No entanto, caso o estado geral de saúde mude isto é, haja alguma alteração na saúde física ou mental, a mensagem de estado geral de saúde deve voltar a ter a saída padrão (sem o emoji).

Assim, além da alteração no método **getStatusGeral()**, é preciso adicionar um novo método:

- **void definirEmoji(String valor)**