Exercícios Registros e Enum

1. Escreva um trecho de código para fazer a criação dos novos tipos de dados conforme solicitado abaixo:
• Horario: composto de hora, minutos e segundos.
• Data: composto de dia, mes e ano.
• Compromisso: composto de uma data, horario e descricao do compromisso.

2. Escreva uma função que receba duas datas (utilizando o tipo Data definido no exercício anterior), cada um representando uma data válida, e calcule a quantidade de dias que decorreram entre as duas datas. Implemente um programa para testar a sua função.

3. Escreva um programa que permita controlar as notas de alunos em uma turma. Para cada aluno são armazenadas as seguintes informações: matrícula, nome, faltas, 3 notas, 1 notas de reposição e o seu status (que indica a situação acadêmica do aluno). O status do aluno é representado por uma constante (REPF, REP, REPN, REC, APR, APRN, REMF ou RENF). Segue o significado de cada constante:

- **REPF** - Reprovado por Faltas - Reprovado por não atender os critérios de assiduidade (no mínimo 75% de assiduidade).
- **REP** - Reprovado por Média - Reprovado porque a média foi inferior a cinco (após a reposição) ou a três (neste caso sem direito a reposição).
- **REPN** - Reprovado por Média e Nota - Reprovado porque a média está entre cinco e sete e a nota da reposição foi inferior a três.
- **REC** - Em Recuperação - Aluno que fará a reposição.
- **APR** - Aprovado por Média - Aluno aprovado com média maior ou igual a sete.
- **APRN** - Aprovado por Notas - Aluno com média entre cinco e sete e que não tirou nenhuma nota inferior a três antes da reposição ou com nota entre cinco e sete após a reposição, e que tirou mais de três na reposição.
- **REMF** - Reprovado por Média e Faltas - Reprovado porque a média foi inferior a cinco (após a reposição) ou a três (neste caso sem direito a reposição) e também por não atender os critérios de assiduidade.
- **RENF** - Reprovado Por Notas e Faltas - Reprovado porque a média está entre cinco e sete e a nota da reposição foi inferior a três e também por não atender os critérios de assiduidade.

Sobre a turma são armazenadas as seguintes informações: nome da turma, total de aulas dadas, total de alunos e uma lista de alunos.

A fim de organizar o seu código, implemente, na ordem, as seguintes partes do programa:

1. Implemente o conjunto de registros e enumerações que permitam representar da melhor forma um aluno e uma turma, tendo em conta a descrição acima;
2. Implemente a função calculaMediaAluno(), seguindo a assinatura a seguir, que recebe um registro de aluno, calcula e retorna a média final do aluno, considerando as 3 notas das unidades e a nota de reposição (apenas quando for o caso);

```
float calculaMediaAluno (Aluno* registro)
```

3. Implemente a função atualizaSituacao(), seguindo a assinatura a seguir, que recebe um registro de aluno e o total de aulas da turma, aplica as regras para o status (definidas no início da questão), atualizando/definindo o estado do registro recebido por parâmetro. A função não retorna valor;

```
void atualizaSituacao (Aluno* registro, int totalAulas)
```

4. Implemente a função imprimeSituacao(), seguindo a assinatura a seguir, que recebe o registro de um aluno e retorna a descrição do status destes aluno (Ex: para o status REPF esta função deve retornar “Reprovado por Faltas”);

```
const char * imprimeSituacao (Aluno* registro)
```

5. Implemente a macro PERCENTO(a,b) que realiza o cálculo do percentual de a e b, ou seja, retorna o valor em percentual de a em função de b. Esta macro deverá ser usada para o cálculo do percentual de faltas do aluno;
6. Implemente outras funções que julgar necessário para permitir cadastrar uma turma e os dados dos seus alunos. (EXTRA: Aceite o desafio de carregar e armazenar os dados em arquivos!);
7. Implemente a função listaTurma(), seguindo a assinatura a seguir, que recebe um registro de uma turma e imprime os dados da turma, seguido da lista com os seguintes dados de cada aluno: matricula, nome, percentual de faltas, média final e descrição do status;

```
void listaTurma (Turma* registro)
```

8. Implemente um programa para testar os itens anteriores. Utilize redirecionamento de arquivos para os testes. Crie o seu próprio arquivo de dados para testes.