EDVONALDO_HORÁCIO_DOS_SANTOS
PROVA TEÓRICA

Classe Abstrata Usuarios.
Métodos:

1) nomeUsuario(): retorna uma string que declara o nome do usuário.
2) tipoUsuario(): retorna, através de uma string, qual tipo o usuário é (graduação, mestrado, doutorado, professores, pesquisadores, profissionais e técnicos).
3) codUsuario(): retorna um inteiro, único, que identifica o usuário.
4) codProjeto(): método abstrato que será implementado nas classes concretas (ou derivadas). Retorna uma string (letra P seguido de um número), única, que identifica um projeto.

Classe derivada (concreta) Graduacao.
Métodos:

1) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Mestrado.
Métodos:

2) codProjeto retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Doutorado.
Métodos:

3) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Professor.
Métodos:

4) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Pesquisador.
Métodos:

5) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto.Digita -1 para encerrar o projeto.

Classe derivada (concreta) Tecnico.
Métodos:

6) codProjeto retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o método.

Classe abstrata Profissionais.
Métodos:

1) tipoProfissional(): retorna, através de uma string, qual tipo o profissional é: desenvolvedor, testador ou analista.
7) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Desenvolvedor.
Métodos:

1) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Testador.
Métodos:

1) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe derivada (concreta) Analista.
Métodos:

1) codProjeto(): retorna uma string (letra P seguido de um número), única, que identifica um projeto. Digita -1 para encerrar o projeto.

Classe Projetos.
Métodos:

1) identificacaoP(): retorna uma string (letra P seguido de um número), única, que identifica um projeto.
2) descricaoP(): retorna uma string que descreve o projeto.
3) dataInicioP(): retorna uma string no formato xx/xx/xxxx indicando a data de início do projeto.
4) horaInicioP(): retorna uma string no formato hh:mm:ss indiciando a hora de início do projeto.
5) dataTerminoP(): retorna uma string no formato xx/xx/xxxx indicando a data de término do projeto.
6) horaTerminoP(): retorna uma string no formato hh:mm:ss indiciando a hora em que o projeto foi concluído do projeto.
7) coordenadorP(): recebe uma string com nome do coordenador. Verifica se o coordenador é a string professor ou pesquisador.

8) profissionaisEnvolvidosP(): retorna os profissionais envolvidos no projeto. Para isso utiliza o código de identificação do usuário, uma vez que ele é único.
9) numeroAtivides(): retorna um inteiro indicando o número de atividades do projeto. O número varia conforme são adicionadas ou removidas atividades.
10) atividadesP(): retorna uma string no formato Ax, onde x é um número que identifica a atividade. Assim, A1 e A2 são atividades diferentes em um mesmo projeto. E assim sucessivamente.
11) statusP(): retorna um enum indicando o status do projeto. Por padrão quando se cria um projeto atribui-se a ele "Em processo de criação", preenchendo os métodos identificacaoP(), descricaoP(), dataInicioP(), horaInicioP(), dataTerminoP(), horaTerminoP(), coordenadorP(), profissionaisEnvolvidosP(), atividadesP(). Além disso:
    a. Para mudar de "Em processo e criação" para "Iniciado" a função verifica se tem as informações básicas.
    b. Para mudar de "Inciado" para "Em andamento" é confirmada a alocação.
    c. Para mudar "Em andamento" para concluído" deve verificar se há descrição do projeto em descricaoP().

Classe Atividades.

Métodos:

1) identificacaoA(): retorna uma string (número seguida de A), única, que identifica uma atividade. Aqui chama o número de atividades do projeto.
2) descricaoA(): retorna uma string que descreve atividade.
3) dataInicioA(): retorna uma string no formato xx/xx/xxxx indicando a data de início da atividade.
4) horaInicioA(): retorna uma string no formato hh:mm:ss indiciando a hora de início da atividade.
5) dataTerminoA(): retorna uma string no formato xx/xx/xxxx indicando a data de término da atividade.
6) horaTerminoA(): retorna uma string no formato hh:mm:ss indiciando a hora em que a atividade foi concluída.
7) responsavelA(): retorna uma string indicando os responsáveis pela atividade. Digita –1 para encerrar.

8) profissionaisEnvolvidosA(): retorna os profissionais envolvidos na atividade. Para isso utiliza o código de identificação do usuário, uma vez que ele é único.
9) numeroTarefas(): retorna um inteiro indicando o numero de tarefas.
10) tarefasA(): retorna uma string no formato Tx, onde x é um número que identifica a tarefa. Assim, T1 e T2 são tarefas diferentes. E assim sucessivamente.

Classe Consulta.

Métodos:

1) usuarioId(): recebe uma string indicando o nome do usuário. Verifica se o mesmo existe. Retorna os seguintes dados: tipo do usuário, projeto dos quais participa (código do projeto).
2) projetoId(): recebe uma string indicando o nome do projeto. Verifica se o mesmo existe. Retorna os seguintes dados: coordenador do projeto, nome dos usuários participantes seguido do tipo de cada um e número de atividades do projeto. Para cada atividade imprime o número de tarefas. Além disso, retorna o status do projeto.
3) atividadesId(): recebe uma string no formato PxAy, onde x indica o número do projeto e y indica o número da atividade. Verifica se x e y existem no banco. Se existir, imprime o código do projeto, o número total de atividades dele, coordenador, status e o número de tarefas da atividade.

Classe Menu.

Métodos:

1) addProjeto(): método criado para adicionar projetos. Verifica se o mesmo já existe, emitindo true (existe) ou false (não existe). Se não existir, cria o projeto adicionando as informações básicas. Se existe, emite uma mensagem de erro e retorna false.
2) addAtividade(): método criado apra adicionar atividades. Antes, verfica em qual projeto deseja adicionar. Verifica a existência do projeto. Se não existir, emite mensagem de erro. Se existir, prossegue, adicionando as informações básicas e o número de tarefas. Retorna true.

Classe Associacao.

Métodos:

1) nomeUsuario(): recebe um nome. Retorna true se existir e false se não existir. Caso exista, imprime na tela os projetos nos quais

participa mostrando o código do mesmo, o nome coordenador e o nome dos envolvidos em cada projeto seguido de qual tipo de usuário é, as atividades e os participantes de cada uma, as tarefas de cada atividade e os participantes de cada uma. Por fim imprime o status do projeto.

Classe Main.

Métodos:

1) main: chama **Menu** e **Associação.**