Entidade de Relacionamento Múltiplo e Ordenação por um Atributo

1 - Entidade de principal de Relacionamento Múltiplo

Nesta seção vamos prototipar a entidade Competição referenciando as entidades Jogador e Partida.

1.1 - Prototipando a entidade Competição em relacionamento [1:n] com a entidade Partida

Ao definirmos uma competição podemos cadastrar para ela, as partidas do time de volêi do Brasil, que ocorrem unicamente (relacionamento [1:n]) na competição. Ou seja, em uma competição são disputadas várias partidas, mas uma partida é disputada somente em uma única competição.

Neste caso, os conjunto de partidas, migram do módulo partida para a classe Competição.

Vamos aperfeiçoar o tipo de conjunto utilizado para armazenar as partidas. Toda vez que você tiver uma classe associada a um único identificador poderemos utilizar um dicionário para armazenar cada objeto dessa classe associado à sua chave. Desta forma, poderemos recuperar o objeto a partir do valor da sua chave, dado que ela é única (não se repete) no conjunto de objetos armazenados. Somente strings podem ser utilizados como chaves de um dicionário.

```
from util.gerais import compacta_nome

class Competição:

    def __init__(self, nome, tipo, técnico, data_inicial):
        self.nome = nome if nome in ('Campeonato Mundial de Vôlei', 'Copa do Mundo de Vôlei') else 'Sem Nome'
        self.tipo = tipo if tipo in ('masculino', 'feminino') else 'indefinido'
        self.técnico = técnico
        self.data_inicial = data_inicial
        self.id = self.título()
        self.partidas = {}

    def __str__(self):
        formato = '{:<43} {:<29} {:<11} {:<24} {:<10}'
        competição_formatada = formato.format(self.id, self.nome, self.tipo, self.técnico, str(self.data_inicial))
        return competição_formatada

    def título(self): return self.nome + ' ' + self.tipo.capitalize() + ' ' + str(self.data_inicial.ano)

    def inserir_partida(self, partida):
        id_partida = partida.id
        if id_partida not in self.partidas.keys(): self.partidas[id_partida] = partida
        else: print('Partida ' + id_partida + ' já tem cadastro na Competição')
```

No método __init__ o conjunto partidas está sendo inicializado com um dicionário vazio ({ }). O método inserir_partida, que estava definido inicialmente no módulo partida para inserir um objeto em uma lista, é atualizado para inserir um objeto em um dicionário. Inicialmente é obtida a chave do objeto partida: id_partida = partida.id. Então é testado se essa chave não faz parte do conjunto de chaves (self.partidas.keys()) que já foram utilizados para armazenar os objetos no dicionário self.partidas. Se a chave ainda não tiver sido utilizada, o objeto é inserido no dicionário: self.partidas[id_partida] = partida.

Utilizar dicionários facilita muito a recuperação de objetos posteriormente. Se você precisar recuperar o objeto a partir de sua chave no módulo de controle, basta utilizar um comando com essa sintaxe: partida = competição.get_partidas()['G-Itália'] .

O método título, foi implementado para gerar um título para competição, agregando informação relevante, sem formatar o string gerado. Utiliza a função capitalize, disponibilizada pelo Python, para converter um string de forma que todas as suas partes, separadas por espaços, tenham a primeira letra maiúscula e as demais minúsculas. Está sendo utilizada para converter 'feminino' em 'Feminino'.

1.2 - Prototipando a entidade Competição em relacionamento [n:n] com a entidade Jogador

Para objetos da classe Jogador, a situação não é a mesma do que para objetos da classe Partida, porque um mesmo objeto jogador pode estar envolvido em várias competições (relacionamento [n:n]). Ou seja, em uma competição participam vários jogadores, e cada jogador pode participar em várias competições.

A primeira diferença, neste caso, é que precisamos manter o conjunto jogadores no módulo jogador, por que esses objetos podem ser utilizados em várias competições. Mas pelo mesmo motivo que explicamos para o conjunto de partidas, vamos atualizar o conjunto de jogadores de lista para dicionário.

```
jogadores = {}

def get_jogadores(): return jogadores

def inserir_jogador(jogador):
    nome_jogador = jogador.nome
    if nome_jogador not in jogadores.keys(): jogadores[nome_jogador] = jogador
    else: print('Atleta ' + nome_jogador + ' já tem cadastro na Competição')
```

No entanto, precisamos armazenar também quais jogadores participaram de uma dada competição. No entanto, não há disperdício de memória, pois cada objeto da classe Jogador ocupa uma dada posição e espaço de memória, que no caso é referenciada tanto pelos objetos do dicionário jogadores armazenado no módulo jogador, quanto pelos obejtos do dicionário self.jogadores armazenado na classe Competição.

Para armazenar o dicionário self.jogadores, é implementado o método inserir_jogadores na classe Competição. Como todos os jogadores, que partiparam de todas as competições que serão cadastradas, serão cadastrados previamente no conjunto jogadores do módulo jogador, basta passar ao método inserir_jogadores a lista das chaves (nomes) dos jogadores que partiparam da competição que está sendo cadastrada. O método verifica se a chave pertence ao dicionário jogadores do módulo jogador e, então, obtém o objeto jogador a partir de sua chave (nome) e o insere no dicionário self.jogadores da classe Competição. A seguir mostraremos as mudanças na classe Competição.

```
class Competição:

    def __init__(self, nome, tipo, técnico, data_inicial):
        self.nome = nome if nome in ('Campeonato Mundial de Vôlei', 'Copa do Mundo de Vôlei') else 'Sem Nome'
        self.tipo = tipo if tipo in ('masculino', 'feminino') else 'indefinido'
        self.técnico = técnico
        self.data_inicial = data_inicial
        self.id = self.__str__()
        self.jogadores = {}
        self.partidas = {}

    def inserir_jogadores(self, chaves_jogadores):
        for nome_jogador in chaves_jogadores:
            if nome_jogador in get_jogadores().keys(): self.jogadores[nome_jogador] = get_jogadores()[nome_jogador]
            else: print('Atleta ' + nome_jogador + ' não tem cadastro')
```

1.3 - Alterando a classe Jogador para competições femininas ou masculinas

```
competição = None

def salvar_competição_interesse(competição1):
    global competição
    competição = competição1

class Jogador:

    def __init__(self, nome, posição, peso, altura, data_nascimento):
        self.nome = nome
                self.posição = posição if posição in ('central', 'levantadora', 'líbero', 'oposta', 'ponteira') else 'indefinida'
        self.peso = peso
        self.altura = altura
        self.data_nascimento = data_nascimento

    def __str__(self):
        formato = '{:<23} {:<13} {:<9} {:<7} {:<4}'
        data_referência = competição.data_inicial
        jogador_formatado = formato.format(self.nome, self.str_posição(),
                                           str(self.data_nascimento.calcular_idade(data_referência)) + ' anos',
                                           f'{self.altura:.2f}' + 'm', str(self.peso) + 'kg')
        return jogador_formatado

    def str_posição(self):
        if competição.tipo == 'feminino': return self.posição
        if self.posição in ['central', 'líbero']: return self.posição
        if self.posição == 'levantadora': return 'levantador'
        if self.posição == 'oposta': return 'oposto'
        if self.posição == 'ponteira': return 'ponteiro'
```

O módulo competição importa o módulo jogador. Para não haver circularidade (A importa B e B importa A), o módulo jogador não pode importar o módulo competição. Uma alternativa para que a classe Jogador, definida no módulo jogador, possa ter acesso ao atributo data_inicial da classe Competição, é definir uma variável global no módulo jogador, para armazenar o objeto competição.

O método salvar_competição_interesse é utilizado para armazenar o objeto competição em uma variável global, para que seja possível saber a idade do jogador na época em que ele estava participando da competição que interessa para aquele jogador. Quando você altera um dado global em uma função é necessário utilizar a palavra reservada global para informar que o parâmetro não é local (válido somente no escopo da função). Observe que a palavra reservada global só é necessária para alterar o valor de uma variável global; para ler o valor você pode não precisa utilizá-la.

No Tutorial 1, os valores para posição das jogadoras foram definidos nos feminino. Agora, no Tutorial 2 podemos corrigir a posição para levar em conta se o tipo da competição é masculino ou feminino. Para fazer essa correção, foi definido o método str_posição na classe Jogador. Se o tipo da competição é feminino, ou a posição for central ou líbero (substantivos que são utilizados nos dois gêneros), o método retorna o próprio valor da posição. Caso contrário, o método retorna a posição convertida para o masculino.

2 - Ordenação por um Atributo : Especificação e Prototipagem

Inicialmente vamos especificar o comportamento de uma função que possa ser utilizada para ordenar, em ordem decrescente, um conjunto de objetos com base em um de seus atributos. A seguir, vamos implementar essa especificação.

2.1 - Ordenação de Jogadores em ordem descrescente de altura

Vamos especificar um algoritmo que possa ordenar objetos de um conjunto, a partir de um atributo qualquer desse objeto. Por exemplo, poderíamos querer ordenar, de forma descrescente um conjuntos de jogadores baseados em suas alturas: do mais alto para o menos alto. Vamos ilustrar a funcionalidade para ordenar jogadores em ordem descrescente de altura em três tipos de detalhamentos: (a) uma descrição textual; (b) a especificação do algoritmo correspondente à descrição; e (c) a implementação correspondente à descrição algoritmo.

Uma forma de especificar a ordenação de jogadores em função descrescente de sua altura, por exemplo, é a de utilizar dois conjuntos de jogadores: (a) o conjunto de jogadores desordenados, inicializado com os jogadores que se deseja ordenar; e (b) o conjunto de jogadores já ordenados inicialmente vazio. Então, cada jogador do conjunto de jogadores ordenados é comparado com cada um dos jogadores do conjunto de jogadores já ordenados.

Enquanto a altura do jogador desordenado for menor que altura do jogador ordenado, continue comparando com o próximo jogador ordenado. Quando a altura for maior, significa que o jogador desordenado deve ser inserido no conjunto de jogadores já ordenados na posição do jogador ordenado (que tem altura menor que a sua), deslocando o jogador ordenado comparado e os seus próximos jogadores de uma posição. Se a altura do jogador desordenado for menor que altura de todos os jogadores já ordenados, ele deve ser apendado no final do conjunto.

Para especificar o algoritmo, vamos citar um comparador da altura de dois jogadores, no entanto como veremos na próxima seção, este algoritmo é genérico e poderiamos utilizar um comparador para qualquer atributo do objeto.

Algoritmo de ordenação descrescente por um atributo

- inicialize o conjunto de jogadores ordenados com uma lista vazia
- para cada um dos jogadores do conjunto de jogadores : obtenha o jogador desordenado
  - indique inicialmente o jogador desordenado ainda não foi inserido como ordenado
  - para cada dos jogadores do conjunto de jogadores ordenados : obtenha o jogador ordenado e seu índice no conjunto
    - se comparador de atributos indica que a altura do jogador não ordenado é maior do que a altura do jogador ordenado
      - insira o jogador desordenado, no conjunto de jogadores ordenados, ocupando a posição do jogador já ordenado : deslocando os demais jogadores ordenados para suas próximas posições
      - indique que o jogador desordenado já foi ordenado
      - saia do loop de jogadores ordenados : para obter o próximo jogador desordenado
  - se após concluir o loop de jogadores ordenados, o jogador desordenado ainda não foi inserido
    - apende o jogador ao final do conjunto de ordenados
- retorne o conjunto de jogadores ordenados

Implementação do algoritmo com a função ordenar_jogadores_por_um_atributo. Vamos utilizar esse nome, por que no Tutorial 3, definiremos adicionalmente a função ordenar_jogadores_por_dois_atributos.

```
def comparador_altura(jogador1, jogador2): return jogador1.altura > jogador2.altura

def ordenar_jogadores_por_um_atributo(jogadores, comparador):
    jogadores_ordenados = []
    for jogador_desordenado in jogadores:
        ordenou_jogador = False
        for índice, jogador_ordenado in enumerate(jogadores_ordenados):
            if comparador(jogador_desordenado, jogador_ordenado):
                jogadores_ordenados.insert(índice, jogador_desordenado)
                ordenou_jogador = True
                break
        if not ordenou_jogador: jogadores_ordenados.append(jogador_desordenado)
    return jogadores_ordenados
```

O métofo insert recebe como parâmetro uma posição e um objeto. Sua execução então desloca para frente os objetos da lista, a partir dessa posição, para poder inserir o objeto na posição que ficou vaga.

A chamada da função ordenar_jogadores_por_um_atributo no módulo de controle seria da seguinte forma:

```
jogadores_ordenados = ordenar_jogadores_por_um_atributo(competição.jogadores.values(), comparador_altura)
```

Essa chamada só é possível, porque Python suporta a passagem de uma função como parâmetro. O método values retorna os objetos do dicionário de jogadores.

Mas Python também permite que você passe como parâmetro uma função anônima (sem nome). Deste forma, você nem precisa definir a função comparador_altura. Esta é a opção que vamos utilizar na nossa implementação, por ser a mais compacta. A sua chamada no módulo de controle ficará da seguinte forma:

```
jogadores_ordenados = ordenar_jogadores_por_um_atributo(jogadores=competição.jogadores.values(),
        comparador=lambda jogador1, jogador2: jogador1.altura > jogador2.altura)
```

A palavra reservada lambda é utilizada para definir uma função anônima, com a seguinte sintaxe: lambda lista_parâmetros : corpo. A função anônima, passada como parâmetro tem:

- como lista de parâmetros: jogador1 e jogador2;
- e como corpo, a comparação: jogador1.altura > jogador2.altura.

No entanto, para passar uma função anônima como parâmetro, seu corpo deve ter um único comando. Quando o corpo da função tem vários comandos, você deverá definir uma função nomeada (ex: calcular_altura_média_jogadores) e, então, passá-la como parâmetro.

No Tutorial 3, veremos que Python disponibiliza uma função de ordenação. Mas no Tutorial 2, o objetivo é trabalharmos a especificação de textual de uma função, de seu algoritmo e de sua implementação, e a função de ordenação que apresentamos foi utilizada para esse propósito.

2.2 - Ordenação de Jogadores em ordem descrescente de outros atributos

Provavelmente você deve estar se perguntando, porque em vez de passar uma função como parâmetro, a comparação de alturas não foi feita diretamente na função ordenar_jogadores_por_um_atributo.

A passagem da função de comparação como parâmetro, é muito versátil. Inicialmente, imagine que você agora deseja ordenar os jogadores por altura de forma crescente: do mais baixo para o mais alto. Neste caso, basta você alterar o teste realizado na função utilizada como comparador, invertendo a comparação > para <, sem realizar nenhuma alteração na função ordenar_jogadores_por_um_atributo. Neste a caso, a chamada da função ordenar_jogadores_por_um_atributo no módulo ficaria sendo:

```
jogadores_ordenados = ordenar_jogadores_por_um_atributo(jogadores=competição.jogadores.values(),
                            comparador=lambda jogador1, jogador2: jogador1.altura < jogador2.altura)
```

Indo mais além, suponha agora que você deseja realizar o ordenação decrescente de jogadores por idade. Basta alterar o corpo da função anônima passada como parâmetro para:

```
data_referência = competição.data_inicial
jogadores_ordenados = ordenar_jogadores_por_um_atributo(jogadores=competição.jogadores.values(),
    comparador=lambda jogador1, jogador2:
        jogador1.data_nascimento.calcular_idade(data_referência) > jogador2.data_nascimento.calcular_idade(data_referência))
```

Mas como o atributo da classe Jogador agora foi alterado de idade para data_nascimento, que alterações seriam necessárias para utilizar diretamente a data de nascimento sem precisar calcular a idade? Para podermos comparar uma data diretamente, é necessário acrescentar métodos na classe Data, para permitir que objetos criados a partir dela sejam comparáveis, ou seja possam ser comparados diretamente a partir dos comparadores: ==, !=, >, <, >=, <=. Para utilizar diretamente esses comparadores será necessário acrescentar um método padronizado para cada comparador. Na classe Data, este possibilidade é muito útil, para que você possa comparar se dois objetos da classe Data tem a mesma data, ou se a data de uma deles é superior ou inferior à data do outro.

Precisamos acrescentar os seguintes métodos padronizados de comparação na classe Data:

- __eq__ (equal : igual) para == ;
- __ne__ (not equal : diferente) para != ;
- __gt__ (greater than : maior que) para > ;
- __ge__ (greater or equal: maior ou igual) para >= ;
- __lt__ (lesser than : menor que) para < ;
- __le__ (lesser or equal : menor ou igual) para <= .

A seguir a implementação dos seis métodos de comparação.

```
class Data:

    def __eq__(self, data):
        if self.dia == data.dia and self.mês == data.mês and self.ano == data.ano: return True
        return False

    def __ne__(self, data): return not self == data

    def __gt__(self, data):
        if self.ano > data.ano: return True
        elif self.ano < data.ano: return False
        if self.mês > data.mês: return True
        elif self.mês < data.mês: return False
        if self.dia > data.dia: return True
        elif self.dia < data.dia: return False
        return False

    def __lt__(self, data):
        if self.ano < data.ano: return True
        elif self.ano > data.ano: return False
        if self.mês < data.mês: return True
        elif self.mês > data.mês: return False
        if self.dia < data.dia: return True
        elif self.ano > data.ano: return False
        return False

    def __ge__(self, data):
        if self < data: return False
        else: return True

    def __le__(self, data):
        if self > data: return False
        else: return True
```

Com esses métodos na classe Data, é possível comparar diretamente as datas de nascimentos dos jogadores no módulo de controle.

```
jogadores_ordenados = ordenar_jogadores_por_um_atributo(jogadores=competição.jogadores.values(),
                                comparador=lambda jogador1, jogador2: jogador1.data_nascimento < jogador2.data_nascimento)
```

Observe que para ordenar em ordem descrescente por data de nascimento, é necessário utilizar o comparador <, pois quanto maior a data de nascimento (ou seja mais atual), mais novo será o jogador.

Além da função de ordenação, vamos ilustrar uma função com passagem de parâmetro para alterar sua funcionalidade. Suponha que você deseje obter as partidas da competição nas quais o time de vôlei do Brasil foi vencedor ou perdedor. Podemos utilizar a seguinte implementação na classe Partida.

```
def obter_partidas_mesmo_resultado(partidas_brasil, vencedor):
    partidas_mesmo_resultado = []
    for partida in partidas_brasil:
        sets_vencidos_brasil, sets_vencidos_adversário = partida.placar.split('x')
        brasil_vencedor = True if sets_vencidos_brasil > sets_vencidos_adversário else False
        if vencedor == brasil_vencedor: partidas_mesmo_resultado.append(partida)
    return partidas_mesmo_resultado
```

Neste projeto, o atributo placar foi cadastrado para exibir o primeiro resultado sempre para o Brasil. Portanto, se o primeiro resultado é superior ao segundo resultado, o Brasil é vencedor; do contrário é perdedor.

2.3 - Generalizando para Ordenar Objetos

Implementar a função ordenar_jogadores_por_um_atributo foi mais didático para você visualizar como ela funciona com atributos de jogadores. Agora que você já entendeu como ela funciona, vamos generalizá-la para a função ordenar_objetos_por_um_atributo, que suporta a ordenação de objetos de qualquer classe em função de um de seus atributos numéricos. Como essa função é genérica não faz sentido mantê-la no módulo jogador do diretório entidades e, portanto, ela será definida no módulo gerais no diretório util.

```
def ordenar_objetos_por_um_atributo(objetos, comparador):
    objetos_ordenados = []
    for objeto_desordenado in objetos:
        ordenou_objeto = False
        for índice, objeto_ordenado in enumerate(objetos_ordenados):
            if comparador(objeto_desordenado, objeto_ordenado):
                objetos_ordenados.insert(índice, objeto_desordenado)
                ordenou_objeto = True
                break
        if not ordenou_objeto: objetos_ordenados.append(objeto_desordenado)
    return objetos_ordenados
```

Obviamente, com essa função genérica, a função ordenar_jogadores_por_um_atributo passa a ser totalmente desnecessária e pode ser removida do módulo jogador do diretório entidades.

3 - Controle do Projeto

A seguir, vamos ilustrar o módulo competições_brasil_volêi_mundial no diretório controle. Com a definição da classe Competição poderemos criar mais de uma competição. Para não ficar muito extenso, vamos criar duas competições, Liga das Nações e Campeonato Mundial, ambas com times femininos e ocorrendo no ano de 2022.

O fato das duas competições ocorrem no mesmo ano permite cadastrar menos jogadoras para ambas as competições. De fato, de uma competição para outra houve mudança somente de duas jogadoras. Então vamos cadastrar 16 jogadoras, sendo que em cada competição foram convocadas 14.

Em relação ao cadastro das partidas, precisaremos de duas funções distintas, pois o conjunto de partidas e seus resultados é distinto de uma competição para outra. No corpo principal, precisaremos de um loop para tratar, de forma genérica, o conjunto de competições cadastradas.

```
from util.gerais import imprimir_objetos, ordenar_objetos_por_um_atributo
from util.data import Data
from entidades.competição import Competição, inserir_competição, get_competições
from entidades.partida import Partida, obter_partidas_mesmo_resultado
from entidades.jogador import Jogador, salvar_competição_interesse, inserir_jogador

def cadastrar_jogadores():
    inserir_jogador(Jogador(nome='Ana Carolina Silva', posição='central', peso=73, altura=1.83, data_nascimento=Data(8, 4, 1991)))
    inserir_jogador(Jogador('Ana Cristina Souza', 'ponteira', 79, 1.93, Data(7, 4, 2004)))
    inserir_jogador(Jogador('Caroline Gattaz', 'central', 87, 1.92, Data(27, 7, 1981)))
    inserir_jogador(Jogador('Gabriela Guimarães', 'ponteira', 65, 1.80, Data(19, 5, 1994)))
    inserir_jogador(Jogador('Júlia Bergmann', 'ponteira', 83, 1.91, Data(21, 2, 2001)))
    inserir_jogador(Jogador('Julia Kudiess', 'central', 76, 1.92, Data(2, 1, 2003)))
    inserir_jogador(Jogador('Kisy Nascimento', 'oposta', 81, 1.91, Data(28, 1, 2000)))
    inserir_jogador(Jogador('Lorena Viezel', 'central', 76, 1.90, Data(21, 7, 1999)))
    inserir_jogador(Jogador('Lorenne Teixeira', 'oposta', 73, 1.87, Data(8, 1, 1996)))
    inserir_jogador(Jogador('Macris Carneiro', 'levantadora', 60, 1.78, Data(3, 3, 1989)))
    inserir_jogador(Jogador('Natália Araújo', 'líbero', 59, 1.62, Data(10, 4, 1997)))
    inserir_jogador(Jogador('Nyeme Costa', 'líbero', 66, 1.75, Data(11, 10, 1998)))
    inserir_jogador(Jogador('Priscila Daroit', 'ponteira', 71, 1.83, Data(10, 8, 1988)))
    inserir_jogador(Jogador('Roberta Ratzke', 'levantadora', 68, 1.85, Data(28, 4, 1990)))
    inserir_jogador(Jogador('Rosamaria Montibeller', 'ponteira', 74, 1.85, Data(9, 4, 1994)))
    inserir_jogador(Jogador('Tainara Santos', 'ponteira', 81, 1.90, Data(9, 3, 2000)))


def cadastrar_partidas_liga_nações_2022(competição):
    competição.inserir_partida(Partida(fase='G', adversário='Alemanha', data=Data(31, 5, 2022), placar='3x1',
                               sets='29–27 23–25 27–25 25–21'))
    competição.inserir_partida(Partida('G', 'Polônia', Data(2, 6, 2022), '3x0', '25–23 25–21 25–22'))
    competição.inserir_partida(Partida('G', 'República Dominicana', Data(3, 6, 2022), '3x1', '25–9 16–25 25–18 25–17'))
    competição.inserir_partida(Partida('G', 'Estados Unidos', Data(4, 6, 2022), '0x3', '21–25 20–25 18–25'))
    competição.inserir_partida(Partida('G', 'Turquia', Data(15, 6, 2022), '3x1', '19–25 25–23 25–23 25–23'))
    competição.inserir_partida(Partida('G', 'Países Baixos', Data(16, 6, 2022), '3x0', '25–16 25–15 25–23'))
    competição.inserir_partida(Partida('G', 'Itália', Data(18, 6, 2022), '1x3', '17–25 15–25 25–14 14–25'))
    competição.inserir_partida(Partida('G', 'Sérvia', Data(19, 6, 2022), '3x0', '25–21 25–9 25–21'))
    competição.inserir_partida(Partida('G', 'China', Data(28, 6, 2022), '3x2', '25-20 25–23 18–25 21–25 15–11'))
    competição.inserir_partida(Partida('G', 'Coreia do Sul', Data(30, 6, 2022), '3x0', '25–17 25–19 25–13'))
    competição.inserir_partida(Partida('G', 'Bulgária', Data(1, 7, 2022), '3x0', '25–21 25–20 25–18'))
    competição.inserir_partida(Partida('G', 'Tailândia', Data(2, 7, 2022), '3x1', '25–18 26–24 23–25 25–23'))
    competição.inserir_partida(Partida('Q', 'Japão', Data(13, 7, 2022), '3x1', '29–27 28–26 20–25 25–14'))
    competição.inserir_partida(Partida('S', 'Sérvia', Data(16, 7, 2022), '3x1', '14–25 25–18 26–24 25–19'))
    competição.inserir_partida(Partida('F', 'Itália', Data(17, 7, 2022), '0x3', '23–25 22–25 22–25'))


def cadastrar_partidas_campeonato_mundial_2022(competição):
    competição.inserir_partida(Partida(fase='G', adversário='Chéquia', data=Data(24, 9, 2022), placar='3x1',
                               sets='25–20 25–16 22–25 25–18'))
    competição.inserir_partida(Partida('G', 'Argentina', Data(26, 9, 2022), '3x0', '25–19 25–13 25–21'))
    competição.inserir_partida(Partida('G', 'Colômbia', Data(28, 9, 2022), '3x0', '25–14 25–12 25–20'))
    competição.inserir_partida(Partida('G', 'Japão', Data(30, 9, 2022), '1x3', '22–25 19–25 25–17 20–25'))
    competição.inserir_partida(Partida('G', 'China', Data(1, 10, 2022), '3x1', '23–25 25–17 25–22 25–22'))
    competição.inserir_partida(Partida('G', 'Itália', Data(4, 10, 2022), '3x2', '25–20 22–25 22–25 25–21 17–15'))
    competição.inserir_partida(Partida('G', 'Porto Rico', Data(6, 10, 2022), '3x0', '25–15 25–19 25–21'))
    competição.inserir_partida(Partida('G', 'Países Baixos', Data(7, 10, 2022), '3x0', '25–19 25–19 25–20'))
    competição.inserir_partida(Partida('G', 'Bélgica', Data(8, 10, 2022), '3x1', '26–28 25–17 25–11 25–16 '))
    competição.inserir_partida(Partida('Q', 'Japão', Data(11, 10, 2022), '3x2', '18–25 18–25 25–22 27–25 15–13'))
    competição.inserir_partida(Partida('S', 'Itália', Data(13, 10, 2022), '3x1', '25–23 22–25 26–24 25–19'))
    competição.inserir_partida(Partida('F', 'Sérvia', Data(15, 10, 2022), '0x3', '24–26 22–25 17–25'))
```

```
def cadastrar_competições():
    competição = Competição(nome='Liga das Nações de Vôlei', tipo='feminino', técnico='José Roberto Guimarães',
                            data_inicial=Data(31, 5, 2022))
    competição.inserir_jogadores(chaves_jogadores=['Ana Carolina Silva', 'Ana Cristina Souza', 'Gabriela Guimarães',
        'Júlia Bergmann', 'Julia Kudiess', 'Kisy Nascimento', 'Lorena Viezel', 'Lorenne Teixeira', 'Macris Carneiro',
        'Natália Araújo', 'Nyeme Costa', 'Priscila Daroit', 'Roberta Ratzke', 'Rosamaria Montibeller'])
    cadastrar_partidas_liga_nações_2022(competição)
    inserir_competição(competição)
    competição = Competição('Campeonato Mundial de Vôlei', 'feminino', 'José Roberto Guimarães', Data(23, 9, 2022))
    competição.inserir_jogadores(['Ana Carolina Silva', 'Caroline Gattaz', 'Gabriela Guimarães', 'Julia Kudiess',
        'Kisy Nascimento', 'Lorena Viezel', 'Lorenne Teixeira', 'Macris Carneiro', 'Natália Araújo', 'Nyeme Costa',
        'Priscila Daroit', 'Roberta Ratzke', 'Rosamaria Montibeller', 'Tainara Santos'])
    cadastrar_partidas_campeonato_mundial_2022(competição)
    inserir_competição(competição)


if __name__ == '__main__':
    cadastrar_jogadores()
    cadastrar_competições()
    imprimir_objetos('Competição : nome, tipo, técnico, data_inicial', get_competições().values())
    for competição in get_competições().values():
        print('\n\n=== ' + competição.título() + ' ===')
        salvar_competição_interesse(competição)
        jogadores_competição = competição.jogadores.values()
        imprimir_objetos('Jogador : nome, posição, idade, altura, peso', jogadores_competição)
        partidas_competição = competição.partidas.values()
        imprimir_objetos("Partida : data, fase, placar, adversário, sets", partidas_competição)
        jogadores_ordenados = ordenar_objetos_por_um_atributo(objetos=jogadores_competição,
            comparador=lambda jogador1, jogador2: jogador1.altura > jogador2.altura)
        imprimir_objetos('Jogador : nome, posição, idade, altura, peso -- por ordem decrescente de altura', jogadores_ordenados)
        data_referência = competição.data_inicial
        jogadores_ordenados = ordenar_objetos_por_um_atributo(jogadores_competição, lambda jogador1, jogador2:
            jogador1.data_nascimento.calcular_idade(data_referência) > jogador2.data_nascimento.calcular_idade(data_referência))
        imprimir_objetos('Jogador : nome, posição, idade, altura, peso -- por ordem decrescente de idade', jogadores_ordenados)
        jogadores_ordenados = ordenar_objetos_por_um_atributo(jogadores_competição,
            lambda jogador1, jogador2: jogador1.data_nascimento < jogador2.data_nascimento)
        imprimir_objetos('Jogador : nome, posição, idade, altura, peso'
            + ' -- por ordem crescente de data de nascimento (decrescente de idade)', jogadores_ordenados)
        partidas_vencidas = obter_partidas_mesmo_resultado(partidas_brasil=partidas_competição, vencedor=True)
        imprimir_objetos('Partida : data, fase, placar, adversário, sets -- vencidas pelo Brasil', partidas_vencidas)
        partidas_perdidas = obter_partidas_mesmo_resultado(partidas_competição, False)
        imprimir_objetos('Partida : data, fase, placar, adversário, sets -- perdidas pelo Brasil', partidas_perdidas)
```

4 - Execução do Projeto

O resultado obtido na execução do projeto é ilustrado a seguir.

```
Competição : id, nome, tipo, técnico, data_inicial
1 -   Liga das Nações de Vôlei Feminino 2022      Liga das Nações de Vôlei      feminino    José Roberto Guimarães   31/05/2022
2 -   Campeonato Mundial de Vôlei Feminino 2022   Campeonato Mundial de Vôlei   feminino    José Roberto Guimarães   23/09/2022


Jogador : nome, posição, idade, altura, peso
1 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
2 -   Ana Cristina Souza      ponteira      18 anos   1.93m   79kg
3 -   Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
4 -   Júlia Bergmann          ponteira      21 anos   1.91m   83kg
5 -   Julia Kudiess           central       19 anos   1.92m   76kg
6 -   Kisy Nascimento         oposta        22 anos   1.91m   81kg
7 -   Lorena Viezel           central       22 anos   1.90m   76kg
8 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
9 -   Macris Carneiro         levantadora   33 anos   1.78m   60kg
10 -  Natália Araújo          líbero        25 anos   1.62m   59kg
11 -  Nyeme Costa             líbero        23 anos   1.75m   66kg
12 -  Priscila Daroit         ponteira      33 anos   1.83m   71kg
13 -  Roberta Ratzke          levantadora   32 anos   1.85m   68kg
14 -  Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
```

```
Partida : data, fase, placar, adversário, sets
1 -   31/05/2022   Grupos            3x1   Alemanha              sets: 29-27 23-25 27-25 25-21
2 -   02/06/2022   Grupos            3x0   Polônia               sets: 25-23 25-21 25-22
3 -   03/06/2022   Grupos            3x1   República Dominicana  sets: 25-9 16-25 25-18 25-17
4 -   04/06/2022   Grupos            0x3   Estados Unidos        sets: 21-25 20-25 18-25
5 -   15/06/2022   Grupos            3x1   Turquia               sets: 19-25 25-23 25-23 25-23
6 -   16/06/2022   Grupos            3x0   Países Baixos         sets: 25-16 25-15 25-23
7 -   18/06/2022   Grupos            1x3   Itália                sets: 17-25 15-25 25-14 14-25
8 -   19/06/2022   Grupos            3x0   Sérvia                sets: 25-21 25-9 25-21
9 -   28/06/2022   Grupos            3x2   China                 sets: 25-20 25-23 18-25 21-25 15-11
10 -  30/06/2022   Grupos            3x0   Coreia do Sul         sets: 25-17 25-19 25-13
11 -  01/07/2022   Grupos            3x0   Bulgária              sets: 25-21 25-20 25-18
12 -  02/07/2022   Grupos            3x1   Tailândia             sets: 25-18 26-24 23-25 25-23
13 -  13/07/2022   Quartas de final  3x1   Japão                 sets: 29-27 28-26 20-25 25-14
14 -  16/07/2022   Semifinal         3x1   Sérvia                sets: 14-25 25-18 26-24 25-19
15 -  17/07/2022   Final             0x3   Itália                sets: 23-25 22-25 22-25
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de altura
1 -   Ana Cristina Souza      ponteira     18 anos   1.93m   79kg
2 -   Julia Kudiess           central      19 anos   1.92m   76kg
3 -   Júlia Bergmann          ponteira     21 anos   1.91m   83kg
4 -   Kisy Nascimento         oposta       22 anos   1.91m   81kg
5 -   Lorena Viezel           central      22 anos   1.90m   76kg
6 -   Lorenne Teixeira        oposta       26 anos   1.87m   73kg
7 -   Roberta Ratzke          levantadora  32 anos   1.85m   68kg
8 -   Rosamaria Montibeller   ponteira     28 anos   1.85m   74kg
9 -   Ana Carolina Silva      central      31 anos   1.83m   73kg
10 -  Priscila Daroit         ponteira     33 anos   1.83m   71kg
11 -  Gabriela Guimarães      ponteira     28 anos   1.80m   65kg
12 -  Macris Carneiro         levantadora  33 anos   1.78m   60kg
13 -  Nyeme Costa             líbero       23 anos   1.75m   66kg
14 -  Natália Araújo          líbero       25 anos   1.62m   59kg
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de idade
1 -   Macris Carneiro         levantadora  33 anos   1.78m   60kg
2 -   Priscila Daroit         ponteira     33 anos   1.83m   71kg
3 -   Roberta Ratzke          levantadora  32 anos   1.85m   68kg
4 -   Ana Carolina Silva      central      31 anos   1.83m   73kg
5 -   Gabriela Guimarães      ponteira     28 anos   1.80m   65kg
6 -   Rosamaria Montibeller   ponteira     28 anos   1.85m   74kg
7 -   Lorenne Teixeira        oposta       26 anos   1.87m   73kg
8 -   Natália Araújo          líbero       25 anos   1.62m   59kg
9 -   Nyeme Costa             líbero       23 anos   1.75m   66kg
10 -  Kisy Nascimento         oposta       22 anos   1.91m   81kg
11 -  Lorena Viezel           central      22 anos   1.90m   76kg
12 -  Júlia Bergmann          ponteira     21 anos   1.91m   83kg
13 -  Julia Kudiess           central      19 anos   1.92m   76kg
14 -  Ana Cristina Souza      ponteira     18 anos   1.93m   79kg
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de data de nascimento (mas informando a idade)
1 -   Priscila Daroit         ponteira      33 anos   1.83m   71kg
2 -   Macris Carneiro         levantadora   33 anos   1.78m   60kg
3 -   Roberta Ratzke          levantadora   32 anos   1.85m   68kg
4 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
5 -   Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
6 -   Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
7 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
8 -   Natália Araújo          líbero        25 anos   1.62m   59kg
9 -   Nyeme Costa             líbero        23 anos   1.75m   66kg
10 -  Lorena Viezel           central       22 anos   1.90m   76kg
11 -  Kisy Nascimento         oposta        22 anos   1.91m   81kg
12 -  Júlia Bergmann          ponteira      21 anos   1.91m   83kg
13 -  Julia Kudiess           central       19 anos   1.92m   76kg
14 -  Ana Cristina Souza      ponteira      18 anos   1.93m   79kg
```

```
Partida : data, fase, placar, adversário, sets -- vencidas pelo Brasil
1 -   31/05/2022   Grupos             3x1   Alemanha               sets: 29-27 23-25 27-25 25-21
2 -   02/06/2022   Grupos             3x0   Polônia                sets: 25-23 25-21 25-22
3 -   03/06/2022   Grupos             3x1   República Dominicana   sets: 25-9 16-25 25-18 25-17
4 -   15/06/2022   Grupos             3x1   Turquia                sets: 19-25 25-23 25-23 25-23
5 -   16/06/2022   Grupos             3x0   Países Baixos          sets: 25-16 25-15 25-23
6 -   19/06/2022   Grupos             3x0   Sérvia                 sets: 25-21 25-9 25-21
7 -   28/06/2022   Grupos             3x2   China                  sets: 25-20 25-23 18-25 21-25 15-11
8 -   30/06/2022   Grupos             3x0   Coreia do Sul          sets: 25-17 25-19 25-13
9 -   01/07/2022   Grupos             3x0   Bulgária               sets: 25-21 25-20 25-18
10 -  02/07/2022   Grupos             3x1   Tailândia              sets: 25-18 26-24 23-25 25-23
11 -  13/07/2022   Quartas de final   3x1   Japão                  sets: 29-27 28-26 20-25 25-14
12 -  16/07/2022   Semifinal          3x1   Sérvia                 sets: 14-25 25-18 26-24 25-19

Partida : data, fase, placar, adversário, sets -- perdidas pelo Brasil
1 -   04/06/2022   Grupos             0x3   Estados Unidos         sets: 21-25 20-25 18-25
2 -   18/06/2022   Grupos             1x3   Itália                 sets: 17-25 15-25 25-14 14-25
3 -   17/07/2022   Final              0x3   Itália                 sets: 23-25 22-25 22-25
```

```
=== Campeonato Mundial de Vôlei Feminino 2022 ===

Jogador : nome, posição, idade, altura, peso
1 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
2 -   Caroline Gattaz         central       41 anos   1.92m   87kg
3 -   Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
4 -   Julia Kudiess           central       19 anos   1.92m   76kg
5 -   Kisy Nascimento         oposta        22 anos   1.91m   81kg
6 -   Lorena Viezel           central       23 anos   1.90m   76kg
7 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
8 -   Macris Carneiro         levantadora   33 anos   1.78m   60kg
9 -   Natália Araújo          líbero        25 anos   1.62m   59kg
10 -  Nyeme Costa             líbero        23 anos   1.75m   66kg
11 -  Priscila Daroit         ponteira      34 anos   1.83m   71kg
12 -  Roberta Ratzke          levantadora   32 anos   1.85m   68kg
13 -  Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
14 -  Tainara Santos          ponteira      22 anos   1.90m   81kg
```

```
Partida : data, fase, placar, adversário, sets
1 -   24/09/2022   Grupos             3x1   Chéquia              sets: 25-20 25-16 22-25 25-18
2 -   26/09/2022   Grupos             3x0   Argentina            sets: 25-19 25-13 25-21
3 -   28/09/2022   Grupos             3x0   Colômbia             sets: 25-14 25-12 25-20
4 -   30/09/2022   Grupos             1x3   Japão                sets: 22-25 19-25 25-17 20-25
5 -   01/10/2022   Grupos             3x1   China                sets: 23-25 25-17 25-22 25-22
6 -   04/10/2022   Grupos             3x2   Itália               sets: 25-20 22-25 22-25 25-21 17-15
7 -   06/10/2022   Grupos             3x0   Porto Rico           sets: 25-15 25-19 25-21
8 -   07/10/2022   Grupos             3x0   Países Baixos        sets: 25-19 25-19 25-20
9 -   08/10/2022   Grupos             3x1   Bélgica              sets: 26-28 25-17 25-11 25-16
10 -  11/10/2022   Quartas de final   3x2   Japão                sets: 18-25 18-25 25-22 27-25 15-13
11 -  13/10/2022   Semifinal          3x1   Itália               sets: 25-23 22-25 26-24 25-19
12 -  15/10/2022   Final              0x3   Sérvia               sets: 24-26 22-25 17-25
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de altura
1 -   Caroline Gattaz         central       41 anos   1.92m   87kg
2 -   Julia Kudiess           central       19 anos   1.92m   76kg
3 -   Kisy Nascimento         oposta        22 anos   1.91m   81kg
4 -   Lorena Viezel           central       23 anos   1.90m   76kg
5 -   Tainara Santos          ponteira      22 anos   1.90m   81kg
6 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
7 -   Roberta Ratzke          levantadora   32 anos   1.85m   68kg
8 -   Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
9 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
10 -  Priscila Daroit         ponteira      34 anos   1.83m   71kg
11 -  Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
12 -  Macris Carneiro         levantadora   33 anos   1.78m   60kg
13 -  Nyeme Costa             líbero        23 anos   1.75m   66kg
14 -  Natália Araújo          líbero        25 anos   1.62m   59kg
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de idade
1 -   Caroline Gattaz         central       41 anos   1.92m   87kg
2 -   Priscila Daroit         ponteira      34 anos   1.83m   71kg
3 -   Macris Carneiro         levantadora   33 anos   1.78m   60kg
4 -   Roberta Ratzke          levantadora   32 anos   1.85m   68kg
5 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
6 -   Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
7 -   Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
8 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
9 -   Natália Araújo          líbero        25 anos   1.62m   59kg
10 -  Lorena Viezel           central       23 anos   1.90m   76kg
11 -  Nyeme Costa             líbero        23 anos   1.75m   66kg
12 -  Kisy Nascimento         oposta        22 anos   1.91m   81kg
13 -  Tainara Santos          ponteira      22 anos   1.90m   81kg
14 -  Julia Kudiess           central       19 anos   1.92m   76kg
```

```
Jogador : nome, posição, idade, altura, peso -- por ordem descrescente de data de nascimento (mas informando a idade)
1 -   Caroline Gattaz         central       41 anos   1.92m   87kg
2 -   Priscila Daroit         ponteira      34 anos   1.83m   71kg
3 -   Macris Carneiro         levantadora   33 anos   1.78m   60kg
4 -   Roberta Ratzke          levantadora   32 anos   1.85m   68kg
5 -   Ana Carolina Silva      central       31 anos   1.83m   73kg
6 -   Rosamaria Montibeller   ponteira      28 anos   1.85m   74kg
7 -   Gabriela Guimarães      ponteira      28 anos   1.80m   65kg
8 -   Lorenne Teixeira        oposta        26 anos   1.87m   73kg
9 -   Natália Araújo          líbero        25 anos   1.62m   59kg
10 -  Nyeme Costa             líbero        23 anos   1.75m   66kg
11 -  Lorena Viezel           central       23 anos   1.90m   76kg
12 -  Kisy Nascimento         oposta        22 anos   1.91m   81kg
13 -  Tainara Santos          ponteira      22 anos   1.90m   81kg
14 -  Julia Kudiess           central       19 anos   1.92m   76kg
```

```
Partida : data, fase, placar, adversário, sets -- vencidas pelo Brasil
1 -   24/09/2022   Grupos             3x1   Chéquia                sets: 25-20 25-16 22-25 25-18
2 -   26/09/2022   Grupos             3x0   Argentina              sets: 25-19 25-13 25-21
3 -   28/09/2022   Grupos             3x0   Colômbia               sets: 25-14 25-12 25-20
4 -   01/10/2022   Grupos             3x1   China                  sets: 23-25 25-17 25-22 25-22
5 -   04/10/2022   Grupos             3x2   Itália                 sets: 25-20 22-25 22-25 25-21 17-15
6 -   06/10/2022   Grupos             3x0   Porto Rico             sets: 25-15 25-19 25-21
7 -   07/10/2022   Grupos             3x0   Países Baixos          sets: 25-19 25-19 25-20
8 -   08/10/2022   Grupos             3x1   Bélgica                sets: 26-28 25-17 25-11 25-16
9 -   11/10/2022   Quartas de final   3x2   Japão                  sets: 18-25 18-25 25-22 27-25 15-13
10 -  13/10/2022   Semifinal          3x1   Itália                 sets: 25-23 22-25 26-24 25-19

Partida : data, fase, placar, adversário, sets -- perdidas pelo Brasil
1 -   30/09/2022   Grupos             1x3   Japão                  sets: 22-25 19-25 25-17 20-25
2 -   15/10/2022   Final              0x3   Sérvia                 sets: 24-26 22-25 17-25
```

Acabamos de completar a ilustração da Etapa 2 do projeto de referência.