یادداشت‌های درسی (۲)

ما هرگز بلندای قامت خود را نمی‌دانیم،
مگر ما را فرا خوانند که برخیزیم و قد بر افزاریم
و آنگاه، اگر به طرح وجود خویش وفادار مانده باشیم
قامت ما از آسمانها خواهد گذشت.

اِمیلی دیکنسن (۱۸۸۶-۱۸۳۰)

مثال ۱: پارامترهای پیش‌فرض

برنامه‌نویس می‌تواند هنگام تعریف تابع سازنده، مقادیری اولیه برای پارامترهای آن نیز تعیین کند. به این مقادیر، **پارامترهای پیش‌فرض** (default parameters) گفته می‌شود. اگر در هنگام فراخوانی تابع سازنده، مقادیری متفاوت برای پارامترهای آن مشخص نشود، مقادیر اولیه مورد استفاده قرار خواهند گرفت.

```
1.  class Dog:
2.      def __init__(self, breed="German Shepherd", color="Tan Black"):
3.          self.breed = breed
4.          self.color = color
5.      def dog_breed(self):
6.          print(f"Dog Breed is {self.breed}")
7.      def dog_color(self):
8.          print(f"Dog Color is {self.color}")
9.  def main():
10.     babloo = Dog()
11.     babloo.dog_breed()
12.     babloo.dog_color()
13. if __name__ == "__main__":
14.     main()
```

- از خط ۱ تا خط ۸ کلاس Dog تعریف شده است.
- در این کلاس، دو صفت (attribute) برای اشیاء عضو کلاس (سگ‌ها) در نظر گرفته شده است: breed و color .

- تابع سازنده __init__() (خطوط ۲ و ۳ و ۴) علاوه بر پارامتر self ، دو پارامتر breed (نژاد) و color (رنگ) را نیز به عنوان ورودی گرفته و مقداری اولیه نیز برای آنها مشخص کرده است. خطوط ۳ و ۴ بیانگر این است که مقادیر صفات breed و color اشیایی که با فراخوانی تابع سازنده ایجاد خواهند شد، برابر با مقداری قرار خواهند گرفت که در هنگامه فراخوانی تابع سازنده مشخص می‌شوند. و اگر در هنگامه فراخوانی تابع سازنده، مقداری برای پارامترها مشخص نشود، مقادیر اولیه آنها مورد استفاده قرار گیرند.
- توابع dog_breed() (در خطوط ۵ و ۶) و dog_color() (در خطوط ۷ و ۸) ، هر یک خطی را چاپ می‌کنند که بیانگر نژاد و رنگ سگی است که آنها را فرا خوانده است!
- در خط ۱۰ ، تابع سازنده فراخوانی شده است و شیءای از کلاس Dog ایجاد شده است به نام babloo . از آنجا که تابع سازنده بدون پارامتر فراخوانی شده است، مقادیر صفات breed و color اشیاء عضو کلاس، برابر با مقدار اولیه پارامترهای تابع سازنده قرار خواهند گرفت.
- در خطوط ۱۱ و ۱۲ ، شیء babloo توابع dog_breed() و dog_color() را فراخوانی کرده است. این توابع مقادیری را که با فراخوانی تابع سازنده در خط ۱۰ ، برای صفات نژاد و رنگ یک سگ مشخص شده است، چاپ می‌کنند.

مثال ۲: استفاده از اشیاء به عنوان پارامترهای توابع

اولین پارامتر تابع سازنده و دیگر توابع عضو هر کلاس، self است و بنابراین، آنها عملاً و همیشه یک شیء را به عنوان اولین ورودی خود می‌پذیرند (گرچه پذیرش این شیء صریحاً در هنگام فراخوانی تابع مشخص نمی‌شود). در حالت کلی، یک شیء می‌تواند صریحاً به عنوان ورودی به تابعی که عضو کلاس نیست نیز، خورانده شود. در این حالت، آن تابع به تمام صفات (داده‌های) آن شیء دسترسی خواهد داشت.

```
1.  class Track:
2.      def __init__(self, song, artist):
3.          self.song = song
4.          self.artist = artist
5.  def print_track_info(vocalist):
6.      print(f"Song is '{vocalist.song}'")
7.      print(f"Artist is '{vocalist.artist}'")
8.  singer = Track("The First Time Ever I Saw Your Face", "Roberta Flack")
9.  print_track_info(singer)
```

- از خط ۱ تا خط ۴ کلاس Track تعریف شده است. این کلاس، تنها شامل تابع سازنده است و در بدنه تابع سازنده آن، دو صفت song (آواز) و artist (هنرمند) برای اشیاء عضو آن (قطعات موسیقی) تعریف شده است.
- از خط ۵ تا خط ۷ ، تابع print_track_info(vocalist) در خارج از کلاس تعریف شده است. این تابع، شیءای را به عنوان ورودی می‌پذیرد و مقادیر دو صفت آن شیء (ترانه و خواننده هر قطعه موسیقی) را چاپ می‌کند.
- در خط ۸ ، تابع سازنده فراخوانی شده است و شیءای از کلاس Track ایجاد شده است به نام singer .
- در خط ۹ ، تابع print_track_info(vocalist) در حالتی که شیء singer (خواننده) را به عنوان ورودی گرفته، فراخوانی شده است. تابع وقتی شیء singer را در اختیار می‌گیرد، عملاً خواهد توانست مقدار صفات (داده‌های) آن شیء را چاپ کند.

مثال ۳: متغیرهای کلاس و متغیرهای نمونه

مقادیر یک صفت اشیاء عضو یک کلاس، ممکن است متفاوت با یکدیگر باشند یا ممکن است یکسان باشند. به صفاتی که تنها یک مقدار می‌توانند داشته باشند، class attribute یا class variable گفته می‌شود، چون مقدار آن صفت برای همه اشیاء عضو کلاس یکسان خواهد بود.

و به صفاتی که مقدار آنها متفاوت با یکدیگر است، data attribute یا instance variable گفته می‌شود، چون مقدار آنها به ازای هر نمونه‌ای از کلاس (هر شیء) احتمالاً متفاوت خواهد بود.

```
1.  class Dog:
2.      kind = 'canine'
3.      def __init__(self, name):
4.          self.dog_name = name
5.  d = Dog('Fido')
6.  e = Dog('Buddy')
7.  print(f"Value for Shared Variable or Class Variable 'kind' is '{d.kind}'")
8.  print(f"Value for Shared Variable or Class Variable 'kind' is '{e.kind}'")
9.  print(f"Value for Unique Variable or Instance Variable 'dog_name' is '{d.dog_name}'")
10. print(f"Value for Unique Variable or Instance Variable 'dog_name' is '{e.dog_name}'")
```

- از خط ۱ تا خط ۴ کلاس Dog تعریف شده است. در خط ۲ ، متغیری تعریف شده است به نام kind و مقدار آن 'canine' تعیین شده است. این متغیر یک class variable است، چون گونه همه سگ‌ها یکسان است. (واژه canine به گونه‌ای از سگ‌سانان گفته می‌شود.)

- متغیرهای کلاس را باید بیرون از بدنه تابع‌ها تعریف کرد. شاید مناسب‌ترین مکان برای تعریف آنها، بعد از خط آغازین تعریف کلاس و قبل از تعریف تابع سازنده باشد.
- در خط ۴ ، متغیر dog_name در داخل تابع سازنده تعریف شده است. این متغیر یک instance variable است، چون هر سگ (هر شیء) یک نام مخصوص به خود را دارد.
- در خطوط ۵ و ۶ ، دو بار تابع سازنده فراخوانی می‌شود و دو شیء از کلاس Dog ، به نام‌های d و e ایجاد می‌شوند.
- در خطوط ۷ و ۸ ، مقادیر «صفت کلاسِ» هر یک از دو شیء، یعنی d.kind و e.kind (گونه سگ‌های d و e) چاپ می‌شوند.
- در خطوط ۹ و ۱۰ ، مقادیر «صفت نمونه» هر یک از دو شیء، یعنی d.dog_name و e.dog_name (نام‌های سگ‌های d و e) چاپ می‌شوند.

*****

تمرین : برنامه‌ای شیءگرا بنویسید برای حل این مسأله که آیا سه نقطه دو بعدی، روی یک خط قرار دارند یا خیر.