



이 자습서에서는 프로그램 실행 중에 Python에서 메모리를 해제하거나 지우는 방법을 살펴 봅니다. 프로그램이 대용량 파일을 처리해야 하는 경우 많은 양의 데이터를 처리하거나 데이터를 메모리에 보관해야 합니다. 이러한 유형의 시나리오에서 프로그램은 종종 메모리가 부족할 수 있습니다.

 프로그램의 메모리 부족을 방지하려면 프로그램에서 더 이상 필요하지 않은 변수 또는 데이터를 지워 메모리를 비우거나 지워야 합니다. 다음 방법을 사용하여 Python에서 메모리를 지울 수 있습니다.

`gc.collect()` 메소드를 사용하여 Python에서 메모리 지우기

`gc.collect(generation=2)` 메소드는 Python에서 참조되지 않은 ^{여러 상의 데이터} 메모리를 지우거나 해제하는 데 사용됩니다. 참조되지 않은 메모리는 액세스 할 수 없고 사용할 수 없는 메모리입니다. 선택적 인수 `generation` 은 값 범위가 0에서 2까지 인 정수입니다. `gc.collect()` 메소드를 사용하여 수집 할 객체 생성을 지정합니다.

 `gc.collect()` 메소드는 프로그램 실행 중에 메모리 사용량을 줄이고 참조되지 않은 메모리를 지우는 데 도움이 될 수 있습니다. 메모리의 액세스 할 수 없는 데이터를 삭제하여 프로그램의 메모리 부족 및 충돌을 방지 할 수 있습니다.

아래 예제 코드는 `del` 문을 사용하여 변수를 삭제하는 방법을 보여줍니다.

```
import numpy as np

a = np.array([1,2,3])
del a
```

변수를 삭제 한 후 사용하거나 액세스하려고한다고 가정합니다. 이 경우 프로그램은 액세스하려는 변수가 변수 네임 스페이스에 더 이상 존재하지 않기 때문에 `NameError` 예외를 반환합니다.

예제 코드 :

```
import numpy as np

a = np.array([1,2,3])
del a
print(a)
```

출력:

```
NameError: name 'a' is not defined
```

`del` 문은 네임 스페이스에서 변수를 제거하지만 반드시 메모리에서 지워지는 것은 아닙니다. 따라서 `del` 문을 사용하여 변수를 삭제 한 후 `gc.collect()` 메서드를 사용하여 메모리에서 변수를 지울 수 있습니다.

아래 예제 코드는 `gc.collect()` 메소드와 함께 `del` 문을 사용하여 Python에서 메모리를 지우는 방법을 보여줍니다.

```
import numpy as np
import gc

a = np.array([1,2,3])
del a
gc.collect()
```

`del df` will not be deleted (if there are any reference to the `df`) at the time of deletion. So you need to delete all the references (to it) (with `del df`) (to release the memory.)

So all the instances bound to `df` should be deleted to trigger garbage collection.

Use `objgraph` to check which is holding onto the objects.