What is optimization?

Ans.:- Optimization in Python refers to improving code efficiency to reduce runtime, memory usage, or resource consumption. This can be achieved through algorithm improvements, using built-in libraries like `NumPy`, `Pandas`, and `multiprocessing`, or optimizing loops and data structures. Profiling tools like `cProfile` help identify bottlenecks, enabling developers to enhance performance while maintaining readability.

Define types of optimization.

Ans:- Optimization in Python can be categorized into several types:

1. Algorithmic Optimization: Choosing efficient algorithms and data structures.

2. Code Optimization: Improving code efficiency through refactoring (e.g., avoiding redundant calculations).

3. Memory Optimization: Using memory-efficient structures (e.g., `generators` instead of `lists`).

4. Multithreading/Multiprocessing: Leveraging parallelism to optimize CPU-bound tasks.

5. Using C Extensions: Speeding up performance-critical sections using C extensions like Cython.

6. Library Utilization: Utilizing optimized libraries (e.g., `NumPy`, `Pandas`).

7. Compiler Optimization: Using tools like PyPy for faster execution.

Minimize the function using python f(x,y)=x^2+y^2+3x+4y+5.

Ans.:-
```
import numpy as np

from scipy.optimize import minimize

def f(x):

   X, Y = x

   return X**2 + Y**2 + 3*X + 4*Y + 5

initial = [0, 0]

result = minimize(f, initial)

print("Optimal values for X and Y:", result.x)

print("Minimum value of the function:", result.fun)
```

Output:-
```
Optimal values for X and Y: [-1.49999997 -2.00000001]

Minimum value of the function: -1.2499999999999991
```