**STDARG – ASSERT**

**Thư viện stdarg**

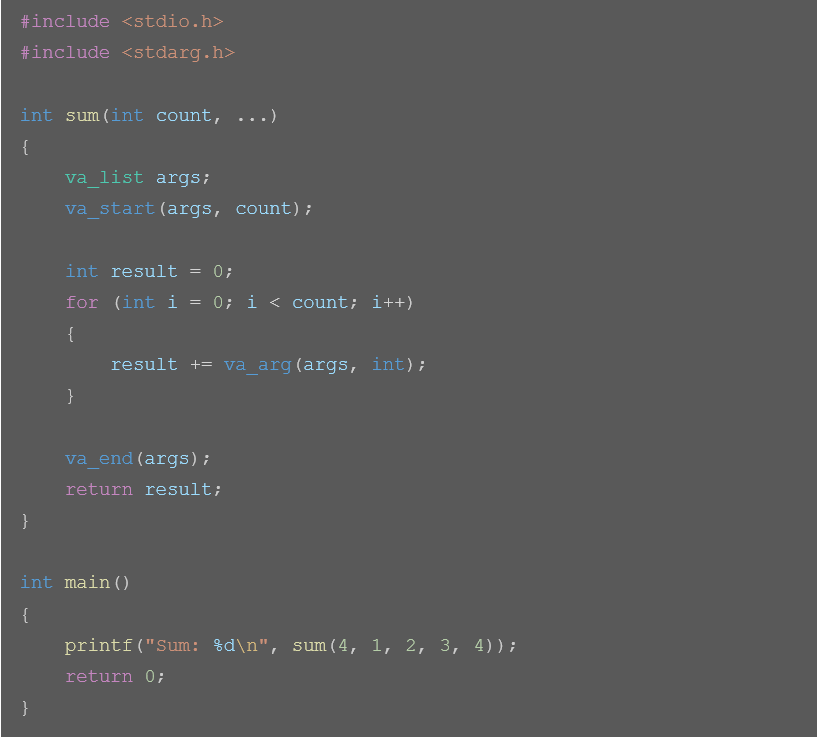
**Mục đích**

* Cung cấp các phương thức để làm việc với các hàm có số lượng **input parameter không cố định**.
* Các hàm như **printf** và **scanf** là ví dụ điển hình.

**Các macro chính trong stdarg.h**

* **va\_list**: là một kiểu dữ liệu để đại diện cho danh sách các đối số biến đổi.
* **va\_start**: Bắt đầu một danh sách đối số biến đổi. Nó cần được gọi trước khi truy cập các đối số biến đổi đầu tiên.
* **va\_arg**: Truy cập một đối số trong danh sách. Hàm này nhận một đối số của kiểu được xác định bởi tham số thứ hai
* **va\_end**: Kết thúc việc sử dụng danh sách đối số biến đổi. Nó cần được gọi trước khi kết thúc hàm.
* **va\_copy**: Sao chép dữ liệu giữa 2 biến cùng kiểu va\_list.

**Các ví dụ minh hoạ**

****

**Kết quả**

Value at 0 : 5

Value at 1 : 8

Value at 2 : 15

Value at 3 : 10

Value at 4 : 13



**Kết quả**

Data.x at 0 is: 2

Data.y at 0 is: 5.000000

Data.x at 1 is: 10

Data.y at 1 is: 57.000000

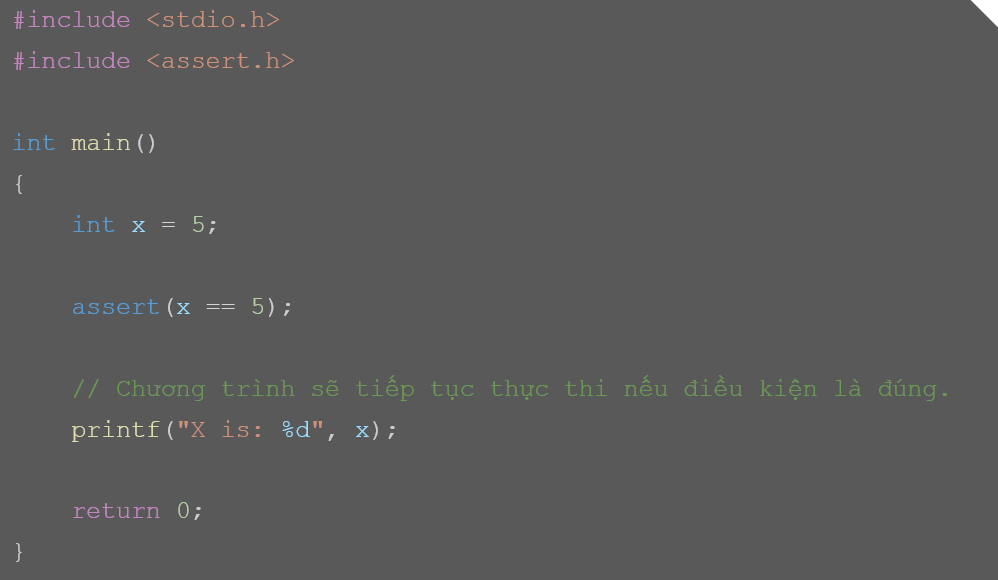
Data.x at 2 is: 29

Data.y at 2 is: 36.000000

**Thư viện ASSERT.H**

* Cung cấp macro assert.
* Macro này được sử dụng để kiểm tra một điều kiện.
* Nếu điều kiện đúng (true), không có gì xảy ra và chương trình tiếp tục thực thi.
* Nếu điều kiện sai (false), chương trình dừng lại và thông báo một thông điệp lỗi.
* Dùng trong debug, dùng #define NDEBUG để tắt debug

**Cách sử dụng cơ bản :**

****

assert(x == 5); // TRUE ⇒chương trình tiếp tục

**Các ví dụ khác**

**a. Macro kiểm tra điều kiện và log**

#define LOG(condition, cmd) assert(condition && #cmd)

**b. Kiểm tra biến nằm trong khoảng**

#define ASSERT\_IN\_RANGE(val, min, max) assert((val) >= (min) && (val) <= (max))

Gọi: setLevel(5); đảm bảo giá trị nằm từ 1 đến 10.

**c. Kiểm tra kích thước kiểu dữ liệu**

#define ASSERT\_SIZE(type, size) assert(sizeof(type) == (size))

Ví dụ: ASSERT\_SIZE(int, 4);