

```
#include <iostream>

int add(int a, int b) {
    return a + b;
}

int main() {
    int x = 5, y = 10;
    int result = add(x, y);
    std::cout << "Sum: " << result << endl;
    return 0;
}
```

The preprocessor reads the `#include` contents of the `iostream` header (providing the functionality).

Any macro definitions (if there were any) would be expanded here.

The compiler verifies the syntax of the code and ensuring that `add()` is correct. It then generates an object file (e.g., `main.o`). The object code represents the compiled code, but it is not yet executable.

The linker resolves external references. For example, it finds the address of the `add` function. It also links in the standard library (e.g., `libc++_abi.so`) defined in the C++ standard library.

The loader assigns memory for the variables (e.g., `result`).

It also loads the program's code into memory.

```
< std::endl;
```



`#include <iostream>` directive and replaces it with the file (which contains declarations for input/output

re any) would be expanded here.

f the code, such as checking if `std::cout` is used properly
tly defined and called.

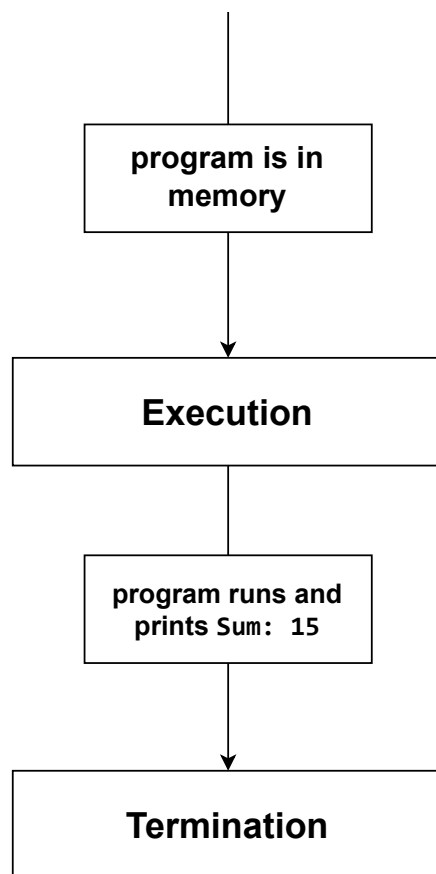
g., `example.o` or `example.obj`) with machine code. This
led version of your program but is not yet fully

nces, like the `std::cout` object and `add()` function. For
ie `add()` function and links it with the compiled code.

r functions like `std::cout` and `std::endl` , which are
ry.

e program's data (e.g., for the integer variables `x` , `y` , and

ito memory for execution.



Initializes the standard library runti

The program starts from `main()` .

The function `add(x, y)` is called, `(15)`.

The result `(15)` is stored in the var

The program then outputs `"Sum: "`

The `main()` function reaches its e

The operating system frees any re:
for the stack and heap.

ime if necessary.

The values `x = 5` and `y = 10` are stored in memory.

which adds `5` and `10` together and returns the result

riable `result` .

`15"` using `std::cout` .

nd and returns `0` , signaling successful completion.

sources allocated to the program, such as memory used