HW0: A + B

Ungraded smoke test | C17 | Individual submission

Overview

HW0 is an ungraded smoke test to run after setting up your programming environment. It does not count toward your course grade. It checks that you can compile a C program, read standard input, and submit source code to Gradescope. Given two integers x and y, print their sum.

Input

Read two integers x and y from standard input, separated by whitespace. Each integer satisfies 0 <= x, y <= 32767.

Output

Print one integer: x + y. Do not print prompts, labels, or explanations. Leading and trailing whitespace is ignored.

Example

Input	Output
`123 500`	`623`

Submission

Submit one file named main.c to the HW0 Programming Assignment on Gradescope using file upload. Do not submit a ZIP archive, a compiled executable, or a Git repository. The source file must be nonempty and no larger than 128 KiB. Check Gradescope for the release time and deadline.

Autograder feedback (ungraded)

The autograder compiles your program as C17 with the command below. Compiler warnings are treated as errors. The 20 test cases produce a diagnostic score out of 100 (5 per passing test), solely to check your setup and program. This score has zero weight in the course grade. Each test allows 1 CPU second (with a 2-second wall-clock timeout) and up to 64 KiB of output. Compilation failure earns 0 points; a failed test earns 0 points for that test.

Compile and test locally

```
gcc main.c -o main -std=c17 -Wall -Wextra -Wpedantic -Werror -O2
./main < sample.in
```

The starter ZIP includes a C file to complete and the sample input/output. Compare your output with sample.out. Passing the sample alone does not guarantee that all diagnostic tests will pass.

Input safety: check the return value of scanf

scanf returns the number of input items successfully converted and assigned. This call reads two integers, so check that it returns 2 before using x and y. If input fails, one or both variables may remain uninitialized; returning early avoids using those values.

```
int x, y;
if (scanf("%d %d", &x, &y) != 2) {
    return 1;
}
printf("%d\n", x + y);
```

Place this fragment inside main and include <stdio.h>. In some compiler and C library configurations, scanf is declared with the warn_unused_result attribute. Ignoring its return value then triggers -Wunused-result. With this assignment's -Werror option, that warning becomes a compilation error, reported as [-Werror=unused-result]. Whether the warning appears depends on your environment; its absence does not mean that input succeeded. Use the return-value check above rather than disabling the warning.

Reference: cppreference on scanf

The return-value section states: "Number of receiving arguments successfully assigned".

cppreference explains that scanf reads from standard input and stores values according to the format string. Its return value counts successful assignments, not characters read. A mismatch before any assignment returns 0; an input failure before any assignment returns EOF. For this two-integer call, only 2 means that both values were assigned. The %d conversion skips leading whitespace, including spaces, tabs, and newlines.

Source: cppreference: scanf / fscanf / sscanf (C), Return value and Notes.

Online handout and starter files | Submit through Gradescope.

For compiler setup or submission issues, contact the course staff.