

Lab 1 — C++ Age Calculator

Zayden

October 7, 2025

Part 1

Problem

Construct an algorithm to find and print out the sum of the first five positive, non-zero integer numbers.

Original Pseudocode

```
SET x to 0
SET n to 0
WHILE n < 5
  ADD n to x
Print out x
```

Questions

- Is the solution proposed logically correct? If not, what is wrong and how would you correct it?
- How might you be able to actually prove her solution works?
- What suggestions do you have to actually prove her solution works?

Response

This solution is not correct. Since **n** never increases, **x** never increases, and 0 is printed. **n** also starts at 0 so if **n** was added correctly, the sum would include 0 as a natural number. Here is a corrected example:

```
SET x to 0
SET n to 1
WHILE n < 5
  ADD n to x
  ADD 1 to n
PRINT out x
```

We cannot prove her solution works since it doesn't.

Part 2

Your friend is trying to write a pseudocode solution to outline the steps required to convert a given distance entered by someone in kilometers to miles. In an effort to help you judge the accuracy of the algorithm below, your friend tells you the following information: - She prefers the metric system - A kilometer is 1000 meters - One kilometer is less than one mile - Kilometers per mile is approximately 1.6093 kilometers

```
DISPLAY PROMPT
"PLEASE ENTER IN THE MILES YOU WANT TO CONVERT:"
READ MILES
KILOMETERS = MILES * 1.3069      DISPLAY MILE
DISPLAY "MILES = "
DISPLAY KILOMETER                DISPLAY "KILOMETERS"
```

Questions

- We know that 15 miles is approximately 24.14 kilometers. Using this information how might you be able to actually prove her solution works? (i.e., is the solution proposed below logically correct?) If there is a problem below, fix it!
- What suggestions do you have to improve the overall readability of her pseudocode?

Response

This solution is not correct. The actual conversion factor is 1.6093. Here is a corrected version with formatting fixes.

```
DISPLAY PROMPT
"Please enter in the miles you want to convert"
READ miles
kilometers = miles * 1.6093
DISPLAY miles " Miles = " kilometers " Kilometers"
```

Variables are now lowercase, the conversion factor is corrected, and readability is improved.

Part 3

Write a C++ Program to calculate how old you are in days.

My code:

```
#include <iostream>
#include <ostream>

int main()
{
    int age;

    std::cout << "enter your age" << std::endl;
    std::cin >> age;

    int leapYears = age / 4;

    int ageInDays = (age * 365) + leapYears;

    std::cout << ageInDays;
    return 0;
}
```

ChatGPT's response:

Excellent start — your program already demonstrates correct use of variables, constants, input/output, and simple arithmetic. Let's go through a detailed evaluation and improvement process.

Pseudocode (what the program does in plain English)

1. Start program
2. Declare an integer variable `age`
3. Ask the user to enter their age
4. Read the user's input into `age`
5. Declare a **constant** for the number of days in a year (`DAYS_IN_YEAR = 365`)
6. Estimate the number of leap years as `age / 4`

7. Compute total days lived as `(age * DAYS_IN_YEAR) + leapYears`
 8. Display the result
 9. End program
-

Feedback and Suggestions

What's good: - Clear structure and good variable naming - Uses both `cin` and `cout` properly - Logical leap year approximation

What can be improved:

1. **Add a constant** for `DAYS_IN_YEAR` (as the prompt suggests).
 2. Include clear output messages (e.g., "You are approximately ____ days old.").
 3. Format output with spaces and context for readability.
 4. Optional: Validate input (ensure `age > 0`).
 5. You don't need `#include <ostream>` — it's redundant since `<iostream>` already includes it.
-

Improved Version (Rewritten Code)

```
#include <iostream>

int main() {
    // Declare variables
    int age;
    const int DAYS_IN_YEAR = 365;

    // Input
    std::cout << "Enter your age in years: ";
    std::cin >> age;

    // Optional: basic input validation
    if (age < 0) {
        std::cout << "Age cannot be negative!" << std::endl;
        return 1;
    }

    // Calculation
    int leapYears = age / 4; // approximate leap years
    int ageInDays = (age * DAYS_IN_YEAR) + leapYears;

    // Output
    std::cout << "You are approximately " << ageInDays << " days old." << std::endl;

    return 0;
}
```

Improvements applied:

- Added `const int DAYS_IN_YEAR`
- Removed unnecessary include
- Added user-friendly output
- Optional error check for negative input

Would you like me to make it even more accurate (e.g., accounting for leap years more precisely using the current date)?