C/C++ Assignment

PG-DAC | Feb 2026 Batch Topics: Data Types · Operators · Functions · Scope · Pointers · References

Instructions

- Write clean, compilable C/C++ code with meaningful variable names and comments.
- Compile and run every program; write the **actual output** below your code.
- Do **not** copy lecture examples — apply the concepts in the new scenarios given.

Q1. Grade Calculator

A student appeared in 5 subjects. Store the marks of each subject as a `float`. Compute the total and percentage. Use the **ternary operator** (no `if-else` allowed) to determine and print the grade according to the table below.

Percentage	Grade
>= 75	A
>= 60	B
>= 45	C
< 45	Fail

Print all subject marks, total, percentage, and grade.

Q2. Type Size Explorer

Write a C program that prints the **size in bytes** of each of the following types using `sizeof`:
`short int`, `int`, `long int`, `long long int`, `unsigned int`, `float`, `double`, `long double`, `char`

Then answer in a comment inside your code:
Which two types have the same size on your machine? Does `unsigned int` have a different size than `int`?

Q3. Temperature Converter

Write a C program with the following **two functions** — declare prototypes before `main()`, define them after `main()`:

- `float celsiusToFahrenheit(float c)` — converts Celsius to Fahrenheit
  (Formula: F = (C × 9/5) + 32)
- `float fahrenheitToCelsius(float f)` — converts Fahrenheit to Celsius
  (Formula: C = (F − 32) × 5/9)

In `main()`, call both functions with sample values and print the results formatted to 2 decimal places.

Q4. Min-Max via Reference Parameters

Write a C++ function:

```
void findMinMax(int a, int b, int c, int &minVal, int &maxVal);
```

The function must find the minimum and maximum of three integers **without using any library function** and store the results into `minVal` and `maxVal` through reference parameters.

In `main()`, declare two variables `int lo, hi`, pass them to `findMinMax`, and print the results. Verify that the original variables `lo` and `hi` are updated after the call.

Q5. Spot the Bug — Call by Value Trap

The following program is meant to double the value of `n` inside `doubleIt()` and see the change reflected in `main()`. It does **not** work as expected.

```
#include <stdio.h>

void doubleIt(int n) {
    n = n * 2;
    printf("Inside doubleIt: %d\n", n);
}

int main() {
    int num = 7;
    doubleIt(num);
    printf("In main after call: %d\n", num);
    return 0;
}
```

1. Compile and run it. Write the output.
2. Fix the function using **call by reference** (C++ style with `&`) so that `num` in `main()` is actually doubled.
3. Write the corrected program and its output.

Q6. Pointer Swap

Declare three integer variables: `x = 10`, `y = 20`, `z = 30`.
Create three pointers `px`, `py`, `pz` pointing to them.
Using **only pointer dereferencing** (no direct use of `x`, `y`, `z` after declaration):

- Swap the values of `x` and `z`.
- Print all three variables before and after the swap.
- Also print the addresses stored in each pointer to confirm they did not change.

Q7. `const` Constant in Functions

Write two C++ functions that use a `const float PI = 3.14159f` declared at the **global scope**:

- `float circleArea(float radius)` — returns area of a circle.
- `float circlePerimeter(float radius)` — returns circumference.

In `main()`, call both with `radius = 7.0` and print results to 4 decimal places.

Additionally, add this line inside `main()` and describe what happens when you compile:

```
PI = 3.0f;   // attempt to modify const
```

Do **not** delete the line — comment it out after observing the error and write the compiler error message as a comment beside it.

Q8. (Challenge) Absolute Value & Clamp — No `if-else`
Write two C++ functions using **only the ternary operator** (no `if`, no `else`, no standard library):

1. `int absolute(int n)` — returns the absolute value of `n`.
2. `int clamp(int val, int lo, int hi)` — returns:
   - `lo` if `val < lo`
   - `hi` if `val > hi`
   - `val` otherwise

Test with the following cases in `main()` and print results in a table format:

val	lo	hi	absolute(val)	clamp(val, lo, hi)
-15	-10	10	?	?
0	-10	10	?	?
25	-10	10	?	?
-3	0	5	?	?

All the best!