Why CS students *must* learn C++ as their main programming language

Vardan Grigoryan (vardanator) Follow
Aug 22, 2017 · 8 min read

Photo from pexels.com

. . .

Disclaimer: This post aims to motivate students to study C++. Professional programmers may find it uninteresting or painful (especially if you code in C# or Java or JavaScript).

Disclaimer 2: Scroll down to see some C++ questions asked in interviews.

Disclaimer 3: Listen to this article's audio recording provided by Listle.

. . .

C++ is the hardest language for students to master, mostly because they have to think much. Really much. We don't claim that C# is easy, or Java is easy, but in comparison, yep, they are easy. Many other popular languages provide some cool "features" allowing developers to concentrate on their actual problem, instead of worrying about language-specific quirks (agree, C++ has so many of them). In Java/C# you have automatic memory management out of the box so you don't have to worry about memory deallocation ever. In

JavaScript, you also have freedom of using **var**, no **int**'s, no **double**'s, no **floats** and **char pointers** at all. "Just store the value somewhere and somehow". The levels of abstraction help programmers solve problems faster than before, which lead many of them to the concept "know less, do more, don't worry, make money". For sure, you don't need to worry about memory management or efficient types or whatever else comes packed in a fancy title "performance". It's just you and the problem you are trying to solve. Language is just a tool. It must help you, not hurt you and your little feelings. And your job title (senior, right?).

What would become a programming student who will study, for example, JavaScript as their first and only language? Definitely not a programmer. Just an advanced user, or you might say a "StackOverflow copy-paster". Why so? Let me introduce you Alice, she's a lawyer, she's good with computers, she can install software, setup a network connection, she's able to distinguish WiFi from a 3g connection, but she doesn't know programming. She is familiar with different operating systems for desktop and mobile. She even uses two operating systems, an OS X installed on her MacBook and a Windows, installed on her office desktop. She even knows that Mac applications couldn't be installed on Windows. She strives to master a software for her job, some lawyer-specific soft, with many menu items, buttons and dialogs. To master this software, she took a class for "SupaLoya2012-Ultimate", she learned difficult queries to request the court cases, she learned the protocol of using the SupaLoya2012-Ultimate, e.g. the order of buttons and menu items to click to get desired results. She definitely is not a programmer, you can call her someone who knows how to work on a computer better than many

others and she knows some complex and specific software, its protocols, queries and stuff.

And then comes Bob. Bob loves computers, he watches too many movies about hackers, programmers, startupers and hackers again. Did I mention hackers? He is good with computers like Alice, although Bob doesn't know SupaLoya2012-Ultimate, he knows JavaScript. Well, he kind of knows it, he took classes, he learned much, he skipped Promises, but is able to make some AJAX requests. So he can write some "code", which, in turn could be used in "production" environment. Actually, Bob is not a programmer, but he claims the opposite to his mom. Yeah, Bob's mom doesn't know that Bob is not aware of memory addressing, virtual machines, just-in-time compilation or "is-a"/"has-a" relationship. Worse , Bob uses $i++$ instead of $++i$ everywhere. Meh, what's the difference. Anyway, does Bob differ from Alice? You can teach Alice the "protocols" of JavaScript, the "difficult queries" to "request" something and she would probably end up like Bob, i.e. calling herself a programmer. Although s he wouldn't, she is humble enough.

So, why is it a MUST for students to study C++ as their first programming language (and spend a lot of time doing it). Because, by learning C++, students have to:

- worry about manual memory management;
- know the difference between a compiler, linker and loader;
- find out that compilers make some optimizations (compilers code better than you);

- get familiar with meta-programming;
- distinguish compile time from run-time;
- really understand the low-level implementation of polymorphism (such as virtual tables and virtual table pointers, or dynamic type identification);
- understand pointer arithmetics, which could be a good base for understanding node-based data structures (linked lists, trees or graphs);
- find out that the compiler generates platform-specific code, and discover that there are many other platforms besides Windows on x86;
- find out that there are ELFs and PEs and other executable file formats, each of which has a bunch of sections you should at least partially be familiar with;
- find out that the size of data types is something you have to worry about (sometimes);
- implement some function pointers to understand how callbacks work under the hood;
- dive deeper into generic programming;
- use and understand iterators, implement containers supporting various categories of iterators;
- … the list goes on and on and on..

These are some skills and knowledge that are a MUST for any CS student, at least for any CS student who is willing to become a good programmer. Mastering C++ guarantees the required experience to

master almost any other programming language. Why C++? Easy to answer to this one. Tell me what:

1. you want to be a world-class developer working on really interesting stuff…
2. …or you just want to make some money while coding routine tasks with some CurrentlyPopularJSFramework?

If your answer is “ye-yeah, I want to be a rock-star developer, ” then C++ is your choice. If you are the guy who claims “oh, come on, language is just a tool, I know React, I can do a lot of stuff, I make money, man!” Sure you are right, no one tells you shouldn’t use other tools, no one tells you shouldn’t solve problems with a tool created just for that particular problem. Finally, no one says you should code some website’s front-end with C++. It’s your choice, but remember this, JavaScript runs on engine written in C/C++ (Google V8), .NET Framework CLR is written in C++, even MS Windows is written in C/C++. Java JVM is written in C++, MongoDB, Redis, web-browsers, Linux, MySQL, (I’m trying to bring a wide range of software), Adobe Photoshop, Illustrator, Nginx, OS X is written in a mix of language, but a few important parts are C++, many Google internal/external products (including Google Search), Microsoft Visual Studio, even C# compiler itself is written in C++. (You can find more by googling).

```
Quoting Kurt Guntheroth (from the book "Optimized C++")
```

> *At the dawn of the 21 st century, C++ was under assault. Fans of C pointed to C++ programs whose performance was inferior to*

> *supposedly equivalent code written in C. Famous corporations with big marketing budgets touted proprietary object-oriented languages, claiming C++ was too hard to use, and that their tools were the future. Universities settled on Java for teaching because it came with a free toolchain. As a result of all this buzz, big companies made big-money bets on coding websites and operating systems in Java or C# or PHP. C++ seemed to be on the wane. It was an uncomfortable time for anyone who believed C++ was a powerful, useful tool.*
>
> *Then a funny thing happened. Processor cores stopped getting faster, but workloads kept growing. Those same companies began hiring C++ programmers to solve their scaling issues. The cost of rewriting code from scratch in C++ became less than the cost of the electricity going into their data centers. All of a sudden, C++ was popular again.*

. . .

So, do you want to make some noise? Then learn C++!

Here are some interesting/regular/just-for-fun C++ questions you should answer as a CS student (helpful for a C++ interview preparation).

 # Write a template function `my_alloc()` taking the type as a template parameter and allocating memory using `malloc()` :

```
template <typename T>
T* my_alloc()
{
// your code goes here
}
```

 # What is the output the following code:

```
int i = 4;
int j = i++;
int k = ++j;
std::cout << i << j << k << std::endl;
```

 # What is the output:

```
int arr[] = {1, 2, 3, 4};
int* p = arr;
int* k = p;
std::cout << (*(k + 2) + 1[p] + *(0 + arr)) << std::endl;
```

 # What will be printed:

```
int n = 5;
void *p = &n;
int *pi = static_cast<int*>(p);
++*pi;
std::cout << *pi << std::endl;
```

 # Here is some struct

```
struct Mruct {
 char c1;
 int x;
 char c2;
 int y;
```

```
 char c3;
 int z;
 char c4;
 int k;
};
```

what is the `sizeof(Mruct)` and how to reorder data-members to decrease the `sizeof(Mruct)` ?

 # What is the output of the this program (see questions in the comments):

```
#include <iostream>
class A {
  int m;
public:
  void sayHi() { std::cout << “Hi” << std::endl; }
};
class B : private A {
private:
  void useHi() { sayHi(); }
public:
  B() { useHi(); }
};
int main() {
  B obj; // Questions 1: What will be printed on the screen
  obj.sayHi(); // Question 2: What will be printed on the
screen
  return 0;
}
```

What is the output of this program:

```
#include <iostream>
#define SQUARE(x) (x * x)
int main() {
    int a = SQUARE(2);
```

```
    int b = SQUARE(a++);
    int c = SQUARE(a + b);
    std::cout << a << " " << b << " " << c << std::endl;
    return 0;
}
```

 # Bring an example of Argument-Dependant Lookup

 # What is Empty Base Optimization

 # In the following code:

```
class A {};
class B : public A {};
```

`sizeof(B)` is equal 1, why?

 # In the following code:

```
class A {};
class B : public A {
  char ch;
};
```

`sizeof(B)` is equal to 1, why?

 # In the following code

```
class A {};
```

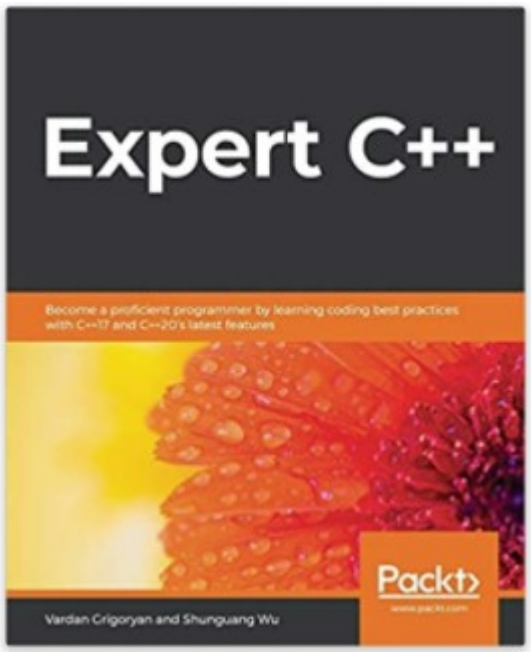
```
class B : public A {
  A m;
};
```

`sizeof(B)` equals 2, why?

 # Why and when you override private virtual functions?

Have more questions like this? Post in comments.

. . .



UPD 2020. This article is my first article about C++. Since then, I've moved towards understanding more low-level details in C++ and programming in general, which led to writing this book. I suggest you to check it out as it contains a lot of explanation about the inner details of concepts in programming.

https://www.amazon.com/gp/product/1838552650/

Programming Cpp C Programming Computer Science Teaching Computer Science