# 1 Programming in C and C++

* 1. Statically allocated variables are allocated in the data segment of memory at compile-time and remain in memory during the entire execution of the process. Dynamically allocated variables are allocated at run-time on the stack or the heap. If on the stack, the memory is deallocated when the variable goes out of scope. If on the heap, the memory is deallocated manually by the programmer with a call to a method like *free*, or by the language’s garbage disposal.



* 1. The *static* keyword in C can have two meanings. If used inside a function for a dynamic variable, its value will persist even over multiple function calls. If used outside of a function on a static/global variable, it means that the variable is only visible inside that particular translation unit.

For both dynamic and static variables, the *extern* keyword means that the definition for the variable will come from a different translation unit.





The benefit of this coding style is that all of the list-traversal logic is contained within the loop. It makes use of the convention of for loops containing an initialisation step, a checking step, and a step to move to the next iteration. Therefore, it is clear at a glance what the loop is doing. This leaves the body of the loop uncluttered, containing only the logic to be applied to each node in turn.

1. I think that they are trying to create two strings, one called s201 with a value of “201” and one called s202 with a value of “202”. One problem they will encounter is that both s201 and s202 are pointing to the same address in memory, so updating the latter will update the former. The second issue is that in C a string literal is an anonymous array, and modifying it is undefined behaviour.
2. One consideration is where the strings are stored. Presumably they will need to be stored on the heap, but if they are going to be accessed one after the other it might be worth trying to allocate them contiguously to make use of the processor’s cache.

Another consideration is deallocaion of the strings. Most of the strings stored will likely not need to be stored for the entire execution of the program. It would therefore be beneficial to implement some sort of garbage collector for the interpreter (perhaps using reference counting) so strings can be deallocated when they are no longer needed.

# 2 Programming in C and C++



The lookup function would work by iterating through the string pairs stored in the NVM, and if the first of the pair matches the key, then return the pointer to the second of the pair (copying its contents to volatile memory if required).

* 1. After the call to f1, the Java case sees p.x being equal to 1 as Objects are passed by reference, whereas the C++ case sees p.x still being equal to 99 as p is being passed by value.

When q is created in the Java case, both p and q reference the same Object, and so when f2 is called, p.x = q.x = 1, and p.y = q.y = 2. In C++, q is created using a copy constructor so p and q are distinct objects each with x=y=99. When p is assigned q using an assignment operator, a bitwise copy is performed so both are still distinct objects with x=y=99. When f2 is called, q is passed by reference so its y value is updated to 2.

To summarise:

Java: p.x = 1, p.y = 2, q.x = 1, q.y = 2.

C++: p.x = 99, p.y = 99, q.x = 99, q.y = 2.



# 5 Further Java





1. The Function reference could be, for example, Squared, which expects its params to be an instance of SquaredParams. However, a different function, for example, Pow, could expect its params to be an instance of PowParams which might contain two fields, a base and an exponent.

You could refactor the client to have the Function class use <R extends Result, Params> as its generics rather than any class which inherits from Params. Its run method could the *instanceof* keyword to check at runtime whether the provided parameter is of the correct type. This check, though defined in the client, would run on the server.

1. The server would be executing arbitrary code from the client which can be dangerous. The code could contain viruses or otherwise malicious operations. Ways to combat this would include static analysis on the bytecode (i.e. an algorithm which looks at the bytecode and tries to determine whether it contains anything malicious, similar to a virus scanner) but this would be slow and prone to errors. A better way to address the vulnerabilities would be to limit the JVM permissions on the server side (e.g., limiting the number of resources it can use, disallowing it from opening sockets, disallowing it from accessing the filesystem).
2. The Result class could contain a String field named to contain an error message and an int field to contain an error number. The meaning of these (and assignment to them) could be implemented in the Function.run method, or in a catch block surrounding the server-side call to Function.run.

# 2 Economics, Law, and Ethics

* 1. I believe that Proposal 1 would be largely unethical, or at least currently impractical to implement in an ethical way. In this context, I posit that an ethical admissions process is one in which candidates are treated with respect, and in which the most deserving candidates are offered places. Proposal 1 violates both of these requirements.

The proposed admissions process disrespects candidates by forcing them through the arduous and time-consuming process of writing a personal statement which will never be read by human eyes. Current natural language processing algorithms are imperfect and could not possibly understand the nuances of such an essay which indicate the intelligence of the candidate. In essence, a personal statement is plea from the candidate to urge the reader to understand their worthiness of a place at the college. Removing the human listener to that plea could be considered a violation of the admissions staff’s Kantian duty to only act on maxims that you’d like to be universal. This is to say, it is unlikely that anybody would like to write an application which would be judged by an algorithm.

Furthermore, due to the flawed nature of natural language processing, the algorithm’s decision would likely largely come down to data collected on the applicants. For example, their grades and other accolades. This opens the door to a variety of biases. In particular, it would likely favour candidates with wealthier backgrounds who had access to more resources in early life. This would violate the second requirement for an ethical admissions process, as the most deserving candidates would not necessarily be granted places. It is in these situations that a human moderator is helpful to assess the applicant’s potential, factoring in their background.

Another way in which this proposal could be unfair is if the algorithm is designed to mimic a human agent in its decision-making. This means that existing biases which have historically been a part of the admissions process (and which a human could work to reduce) would remain in place. Indeed, if an applicant is exceedingly unique, unlike any previous application, the distributional shift (when an input to a machine learning algorithm is significantly different to its training data) could cause the algorithm to badly misjudge the applicant, with no justification for the decision being given.

* 1. The college would need to implement a system for allowing applicants to access and update any data stored about them, or to withdraw from the application process, ensuring that all of their data is deleted. This could be achieved via an online portal, which also includes a privacy notice informing applicants of what data is being stored and why.

Furthermore, applicants would need to be informed about the nature of the automated admissions process, including how the algorithm makes decisions and what the consequences of those decisions might be.

The college would also need to ensure that application data is pseudonymised wherever possible.

The data would need to be stored securely to prevent theft or loss. This could be achieved by internal audits of the data processing, and perhaps appointing a data protection officer to oversee the operation.

* 1. Option a) might be preferable from a classical utilitarian perspective. This is because each student will rent the “nicest” (of course this depends on the preference of the individual) apartment for which they are willing and able to do so, thereby maximising their own individual utility. Classical utilitarian welfare defines welfare as the sum of each individual’s utility. Therefore, option a) maximises utilitarian welfare.

Option a) might also be preferable in terms of Rawlsian welfare. It is likely that the cheapest apartment available with option a) will be cheaper than the fixed price of all of the apartments with option b). Therefore, option b) will increase the likelihood that some students will be priced out of the market and be forced to look for alternative accommodation. Since Rawlsian welfare is defined as the minimum across the utility of each individual, option b) makes it significantly more likely for the Rawlsian welfare to be incredibly low when compared to option a).

In addition, option a) would likely lead to an increase in the utility from the point of view of the college. It allows students who can afford to do so to live more comfortably and be more productive.

* 1. The college may decide to let the price mechanism set the prices for each grade of apartment. That is to say, rent each apartment out at the maximum price at which students are willing and able to rent it.

However, the free market may not be the optimal tool for the college, as the students’ welfare may strongly impact the college’s welfare. The college may wish to ensure that as many students as possible are living in college-owned accommodation to improve the quality of life (and subsequently, the productivity) of the student body. They may therefore decide to artificially deflate the price from the market value, to allow and encourage more students to live in the apartments.

Similarly, the college may choose to price the higher-grade apartments close to the market value, but artificially lower the price of the lower-grade ones. This would allow more students to participate in the market, but still generate a lot of revenue.

However, creating a price ceiling like this might create a surplus of demand, which would not be optimal for the college. It may however be acceptable depending on the elasticity of supply (the extent to which the college could buy up more apartments in response to increased demand).

# 3 Economics, Law, and Ethics

1. The main intellectual property aspect which governs code is copyright. The writer of a non-trivial piece of code (as with other media) is granted automatic copyright, without having to claim such in writing. Most open-source code repositories contain a license stating what is and is not a permitted use of the code, and the copyright holder has sole authority to determine these rules.

Licenses such as MIT are very permissive, allowing almost any use of the code. However, if the program you are writing is to be closed-source, you would not be allowed to include open-source code from a project licensed under, for example, GNU GPLv3 which states that all projects which use the code must also be open-source.

It is also possible that the writer of the code is not in fact the copyright-holder. For example, this would be the case if they were commissioned to write some code, in which case the copyright-holder would likely be whoever commissioned them. In such a case, the author may have included a license with their codebase which is invalid, as they do not have the authority to declare a license.

Though patents are typically reserved for protecting physical inventions, if the code (or a component thereof) can be considered a novel invention, it is possible that it is protected by a patent. There is precedent for certain methods of user authentication being protected by patents (e.g., US patent #11,212,283).

1. One economic argument for making code open-source is that it improves welfare. With the ability to build off of existing ideas, small firms are able to create products of great value, and thus improve the welfare of society. Furthermore, this would increase the output of the economy, boosting GDP.

Another economic argument for making code open-source is that is encourages a free market. If tech giants are allowed to lock down their source code and algorithms, they are able to monopolise the market and set their own prices, free of competition. On the other hand, if they release their source code, other firms could provide similar services at a lower price.

However, one potential effect of this is that it discourages innovation. A firm is disincentivised from creating novel algorithms and features if they are not able to reap the benefits of having done so. Another firm would be able to provide the same service for a fraction of the trouble or expense.

1. An ethical approach would involve 3 major steps. The first would be to patch the vulnerability. As a software company you would have a duty to your clients to keep software vulnerability-free to the best of your knowledge and ability.

The next step would be to inform your clients about the vulnerability and explain how to patch it. If you are selling directly to the public, releasing a public statement about the vulnerability would be the appropriate action. This statement should be accurate, clear, and should inform people how to know whether or not they have been affected by the vulnerability, and what action they can take if so.

If, however, you sell to businesses, reaching out privately to each customer with the same information would be a more appropriate action. This is because a public statement might bring undue negative publicity to the businesses who use your software. You should also advise the businesses on how to responsibly disclose the vulnerability to their affected clients.

Once the vulnerability is patched, you should pay the reporter an amount equal to the value of the information they’ve given you (regardless of the actual amount they’ve demanded). Since they have undeniably given you something of value, it would be unethical to pay less than that value but paying more is also unethical as it encourages the reporter to further extort other software companies.

1. The lawful response would include reporting the data breach to a regulator within 72 hours of being notified of the vulnerability. If the breached data includes or may include high-risk data, the data subjects must also be notified of the breach.

The reporter is likely guilty of violating section 2 of the Computer Misuse Act of 1990, as they have had unauthorised access to data with the knowledge that it was unauthorised, with the intent to commit extortion. It would therefore be legally responsible for the software company to report the hacker to the relevant authorities, e.g., the police.