 doi:10.20944/preprints202012.0516.v1

Comparative Analysis of C++ and Python in Terms of Memory and Time

Farzeen Zehra[1], Darakhshan[2], Maha Javed[3], and Maria Pasha[4].

Department of Software Engineering
NED University of Engineering and Technology, Karachi, Pakistan

***Abstract*— In this era of technology, programming has become more significant than ever before. Python and C++ are both widely used programming languages. Python, the most popular programming language in today's world, is a high-level object-oriented language whereas C++, the language behind most operating systems, is a low-level object-oriented language. In this paper, we present a comparative study of Python and C++. This paper discusses the introduction to these languages, their memory management techniques, and the reasons behind their program execution speed. Furthermore, we analyzed the execution time and memory used by multiple algorithms in both the languages with best, average, and worst cases. They are also compared with respect to the benefits and issues related to them. Results indicate that C++ is faster than Python in execution speed but Python serves as a better language for beginners due to its simplicity. Moreover, for the best results, the language should be selected according to the type of project.**

***Keywords*— benefits, c++, comparative study, execution speed, memory management, python**

I. INTRODUCTION

Software development is proved to be the most demanding field as all domains now are linked with technology. Different languages are in use and new languages are being introduced from time to time. All languages have their pros and cons and they are being used for different purposes. In our research paper, we have compared two commonly used languages, Python, and C++. New programmers prefer Python because of its ease and experienced programmers prefer C++. We will compare these two languages on the basis of time and space utilized so as to know which language is important in which situation. The languages are compared using algorithms by considering the best case, worst case, and the average case of algorithms. We have used Visual Studio for the execution of C++ code and Pycharm for the execution of Python code. Along with the execution of algorithms, we have discussed how C++ and Python allocate memory, their background, and time utilization for the execution of programs.

In this paper, section [II] describes the related work, section [III] gives the overview of python and c++ with their brief history. Section [IV] explains the memory management techniques of both languages whereas performance analysis in terms of execution time is done in section [V]. Section [VI] shows the memory and time analysis of algorithms in both languages. Section [VII] discusses the benefits and issues of both C++ and Python and gives recommendations for beginners. Section [VIII] concludes the paper and section [IX] comprises references.

II. BACKGROUND

With the emerging time, many programming languages are being introduced in the field of computer science. Each language has its own importance, and they also have their drawbacks. But which language should be used by beginners and which language should be used for fast working and precise results? This query is faced by almost all developers. Zakaria Alomari, Oualid El Halimi, Kaushik Sivaprasad, and Chitrang Pandit compared C++, PHP, C#, Java, Python, and VB in this paper [1] on the basis of paradigms, application areas, execution strategy, typing strategy, memory management, and available IDEs. For analysis of all features of languages, the same algorithm has been executed in all languages and their results are being compared. From all the results, it has been deduced that C++ is the fastest language. All languages have their own domain for working where they work better than others. C# is best suited for GUI, for web development, Java and PHP are best. For event-driven programming, VB can be used. Python can be used for rapid prototyping and as C++ is proved to be the best language, that's why it can be used in a large variety of software applications development.

Similarly, Kristijan Stefanoski, Aleksandar Karadimche, and Ile Dimitrievski made a comparative analysis of the performance of two languages, a compiled language that is C++ and a scripting language that is Javascript [2]. Sorting and searching algorithms are used for comparison. The author first compared the time used by both languages in inserting different amounts of data in arrays for Javascript and in vectors for C++ (same as dynamic arrays). Results showed that C++ performed 7ms faster than Javascript. The difference between logging time is 15 ms and 22 ms respectively. The result is the same for different amounts of inputs. Similarly, Linear searching algorithms also showed that C++ is comparatively faster than Javascript. But the sorting algorithm

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

showed the efficiency of the Dynamic Javascript array in comparison to standard C++ arrays. This paper concluded that C++ is relatively faster than Javascript or any scripting language such as Python because C++ is a compiled language and doesn't require any time to be translated into machine code. On the other hand, scripting languages require an interpreter that first translates the code then executes it.

Another study about the comparison between programming languages namely, Python and C++ was conducted. The authors compared the languages based on efficiency. For this purpose, they utilized the sorting algorithms (Quicksort, Merge Sort, Bubble Sort, and Insertion Sort). They calculated the execution time of each sorting algorithm in both languages in two ways. First, without showing data in array, and second, with showing data in array. They also used two ways to compile the code namely, Command Prompt and Cygwin. The results showed that Python took shorter to complete in Cygwin and longer in CMD whereas the results of C++ were converse. Furthermore, both the languages run faster when not showing data in array [3].

In another study [4] conducted by Muhammad Ateeq, Hina Habib, Adnan Umer, Muzammil Ul Rehman, discussed the choices of programming languages for a beginner of programming and from the paper, it is concluded that python is easy to learn and user-friendly language which is much alike English, majority of beginners express satisfaction and comforts while learning python due to its syntax. Other languages like C++, Java, and many more have their importance at their places but students or beginners should always go with Python and don't learn C++ until it is required because C++ has more syntax rules and other programming conventions but in some cases, you can surely go for C++. C++ is the best option for game development and systems. Python is the most powerful programming language compared to any other programming language.

This paper presents a comparative study of Python and C++ under the characteristics of memory allocation, speed, advantages, and disadvantages. The objective of writing this paper is to analyze and compare Python with C++ and find out about their implementation due to a famous statement about Python that it is very slow. Hence, the research questions were: Is python slow? If yes, why? Which is better for beginners?

III. Overview of C++ and Python

A. *Overview of C++*

C++ is a general-purpose compiled language. It was developed in 1979 by Danish Computer scientist "Bjarne Stroustrup" It is an extension of the C language and considered a superset of the C language. Its first edition was released in 1985 [5].

With the passage of time, C++ was updated and new features were introduced in it such as classes, inheritance, static and constant member functions, templates, libraries, and namespaces. C++ is standardized by the International Organization Standardization (ISO) and published in 2011.

A number of programming languages developed on the basis of C++. C# uses C++ syntax and its major features like classes. In fact, the most popular programming language, Java is obtained from C++ but excludes its features like a pointer, operator overloading, and multiple inheritance just to keep the language simple [1].

C++ supports both object-oriented and procedure-oriented programming but actually, it is an Object-Oriented Programming language that is largely used in real-world applications. It is considered to be an efficient language in terms of memory and speed as compared to other programming languages like Java, python, etc [6].

B. *Overview of Python*

In the late 1980s or early 1990s, at the National Research Institute for Mathematics and Computer Science, Netherlands Guido van Rossum developed a programming language called Python. It is derived from many other languages, including Modula-3, C, C++, Algol-68, and other scripting languages.

Python is currently one of the most popular dynamic programming languages, along with others such as Perl, Ruby, etc. It is a popular high-level programming language with high code readability. Unlike other languages such as C++, C, Java, etc, it uses indentation instead of brackets and semicolons [1]. It started gaining popularity in 2003 and now in 2020 according to PYPL PopularitY of Programming Language Index, Python tops the chart. There are two broad versions of Python namely, 2.x and 3.x. It supports programming paradigms such as functional, procedural, reflective, imperative, and object-oriented approaches [8].

IV. Memory Management

The memory system is a prime determinant of performance [9]. The process of allocating, deallocating, and managing memory effectively is called memory management. It basically refers to the allocation of memory when required by the program and then frees it when it's no longer required. Since, how much memory is required for running an application cannot be estimated therefore, additional code is required to manage changing memory requirements. There are two related tasks of memory management:

1) *Allocation:* Allocation of memory on the request of the program for the block of memory. This block of memory is received from the operating system. The part of the memory manager that performs this, is known as the allocator.

2) *Recycling:* Once the memory blocks have been allocated, but no longer required, then they are recycled either by manual memory management in which the programmer decides this or by automatic memory management in which the memory manager does this work [10]

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

.

A. Memory Management in C++

Managing memory is the most crucial part of the programming as it is one of the factors to measure the efficiency of a programming language. In order to write an effective program, a programmer must be careful about the way memory is consumed by the program. Some languages like Java provide their own garbage collector but in C++, a programmer has to manage the memory [11]

.

Kinds of Memory Management:

1. Stack Allocation (Automatic Management):

Stack is the place in computer memory where the programs and all the values returned as a result of the execution of the program will be stored. Stack is maintained throughout the program. Whenever a function is called, its address is pushed in the stack and popped out when the function does its job.

Whenever a C++ program executes, an activation record is created where memory is allocated for all local variables and methods and is deallocated with the variable's lifetime.

It means there is no need to explicitly allocate or deallocate memory in the stack as the compiler manages all stack objects. However, not all the memory requirements are fulfilled by the stack. Some objects like strings need additional memory allocation for the fulfillment of their task [11].

2. Static Memory Allocation:

Static means fixed. Programmers should allocate a fixed size of memory to a particular variable, program or application at compile time. Once the memory is allocated to a variable, it is not used for any other purpose even if most of the memory is not needed by that variable at a particular instant.

For Example, Declare a static array of size 6 to store an integer as shown in Figure 1,

```
int arr [6] = {1,2};
```

In a 64-bit Windows operating system, where the integer data type is of 4 bytes, the memory required by this array is 6x4=24 bytes. It means, no matter array is empty or consists of only 2 integers, 24 bytes are reserved for this array in memory which is not used by any other program

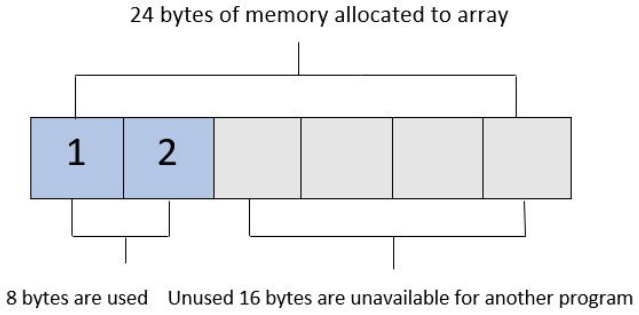

Fig. 1. Static Memory Allocation

This technique is called static memory allocation but this is not an efficient way of memory management as this results in lots of memory wastage. No doubt, it is faster and easier to allocate memory statically but still as a programmer, one should not compromise on memory management [12].

3. Dynamic Memory Allocation (Heap Allocation)

Shortage of memory can only be avoided when memory is used in an efficient way. Dynamic memory allocation is also called manual memory allocation. The most important mechanism about this kind of allocation is that here required bytes of memory are allocated only when the program demands and memory is released or deallocated when it is no longer needed. Moreover, the released memory then once again can be available for other purposes. Using dynamic allocation, a programmer can have full control or access to a memory at run time. This space comes from a large powerful area of dynamic storage called heap memory.

C++ provides an efficient technique that allows the dynamic allocation of memory. C++ uses new and delete keywords for dynamic memory allocation and deallocation respectively. As mentioned above, in C++, a programmer has to manage the memory. This means that once a memory is allocated explicitly using a new keyword, it cannot be reused until and unless it is deallocated with a delete keyword. Pointers are used to handle dynamic allocation.

An array is dynamically allocated as:

```
type* myArray=new type[5];
```

Deallocated as:

```
delete [] myArray;
```

As shown in Figure 2, dynamic memory is allocated for 5 elements of type double in heap [12].

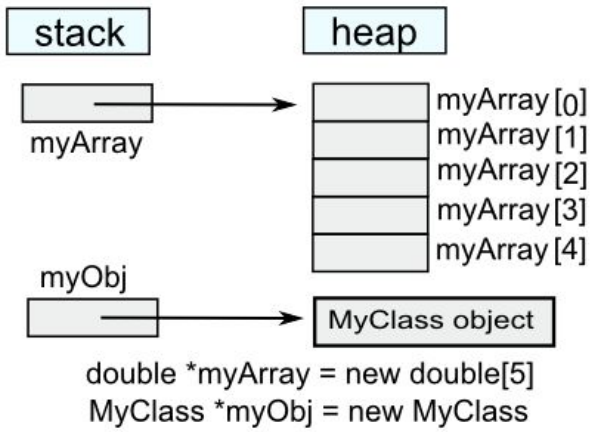

Fig. 2. Dynamic Memory Allocation

Additionally, C++ also uses another function to dynamically allocate memory for the requested bytes that is malloc () and releases memory using the free () function.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

As a result of malloc () failure, a null pointer is returned whereas the new operator throws an exception. So better to use a new operator for memory allocation in C++ [12].

B. Memory Management in Python

Python is a high-level programming language implemented in C language. It handles everything in the form of objects e.g., list, dictionary, tuple, integers are all stored as an object. All these data structures and Python objects are stored in a private heap managed by the Python memory manager.

Four concepts related to the Python memory manager are:

1. *Heap:* It is the collection of all Python managed memory.
2. *Arenas:* These are the largest chunks of memory which has a fixed size of 256 KB each. Python requests them from the system and they are the objects that make up the heap.
3. *Pools:* These are the chunks of memory that make up the arenas of 4 KB each. They are simply arrays due to their fixed sizes.
4. *Blocks:* Python objects are stored in blocks. They have a specific format based on their data types. For instance, an integer takes up more space than a character, for efficiency a different block size is used.

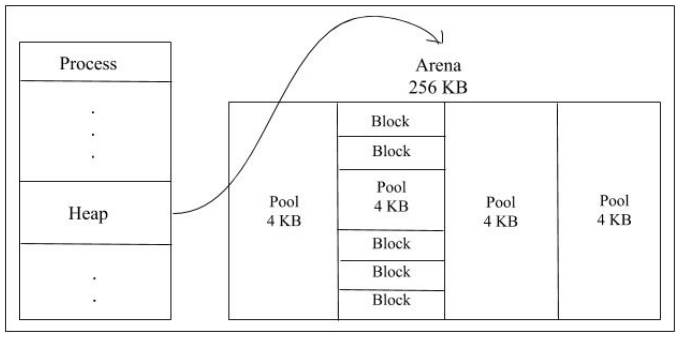

Fig. 3. Relationship between Heap, Arenas, Pools, and Blocks

Memory can only be freed to OS in the form of an arena therefore, once an arena is empty, it is freed. Due to the same reason, whenever Python requests memory, the arena that is already filled will be given if possible instead of an empty one. However, whenever a block is freed, it will not be released to OS but will be kept for further memory allocation. This deallocation of memory is performed automatically by Python through a garbage collector [13].

Deallocating memory with a garbage collector is a process opposite to that of manual memory management in which the programmer has to specify which objects to deallocate [14]. By default, the thresholds for the Python garbage collector are set to 700, 10, 10 for the three generations of collected objects. It will automatically collect once the thresholds have been reached. It also keeps track of all the memory allocations and deallocations, and if the number of allocations minus the number of deallocations reaches 700, the object is either deleted if it's not referenced anymore or it is moved to the next generation if it still has a reference. The same is repeated for generations 2 and 3, but with the lower thresholds of 10 [15].

This process further extends to the lower level as depicted by Figure 1 where the interpreter (CPython, PyPy, etc) interacts with the OS's memory manager to deal with the memory.

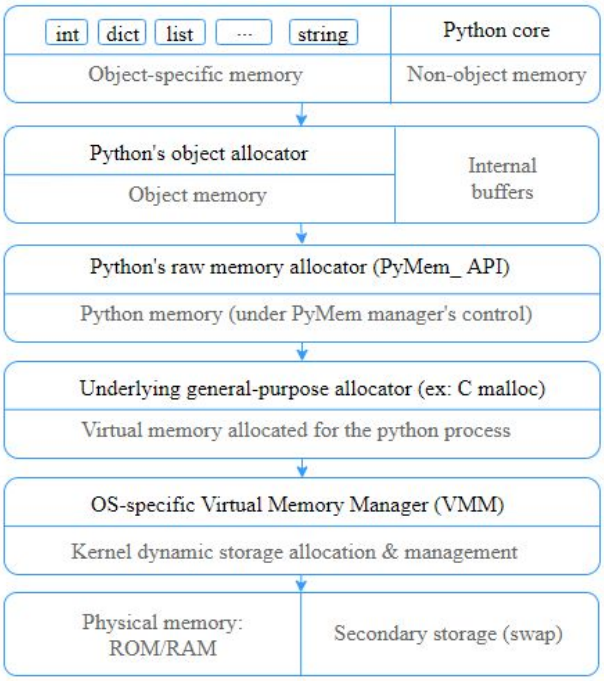

Fig. 4. Python hierarchical layers
Source: Adapted From [27]

In short, the whole process can be stated as to when any program written in python executes, it compiles down to an intermediate code called bytecode which is then interpreted by the Python Runtime Environment (PRE) to machine code. Furthermore, the allocation and deallocation of memory are handled by the garbage collector run by the PRE. Whenever a new object is created, the needed memory is allocated, and once the object goes out of its scope, the memory becomes eligible for Garbage Collection (GC), and eventually would be released by GC [1].

V. TIME ANALYSIS

Analysis of the program execution time is extremely important when making a complex software system. Program running time is measured from the start of the program providing inputs to the termination of the program delivering output [16]. Some languages are slower than others. The reason behind it lies in its implementation. Usually, High-level languages are slower than low-level languages. The reasons behind the speed of executing programs in C++ and Python are discussed below.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

A. Time Analysis in C++

Writing code in C++ is somehow difficult as compared to Python but when it comes to performance, C++ gives excellent results. Research proves that languages like Python, PHP, Java, and C# cannot compete with C++ in terms of speed and memory efficiency. Following are the reasons behind the fast performance of C++.

1. Statically Typed Language:

C++ uses a static typing method. Data types are fixed like integer, float, char, and string. Every variable is stored with its data type. The compiler already knows the data type of the variable and chooses an appropriate process for that particular variable. This means that in C++, there is no need to check the validation of type conversion at run time. This enhances the performance of C++ as it favors the execution process at run time.

2. C++ is a Compiled Language:

Unlike interpreted languages like Python in which the source code is first translated into the byte code by an interpreter and then converted into machine code at run time, C++ allows direct conversion of source code into machine code as it is a compiler-based language. This results in a high improvement in execution speed [1].

3. Dynamic Memory Allocation:

Since C++ facilitates the programmer through the dynamic allocation of memory and enables him to declare and control the lifetime of objects, it doesn't use a garbage collector.. This removes the additional time requirement in which a Java Virtual Machine or Python interpreter must stop for a while and trace the object's lifetime. Hence, your C++ code runs faster [17].

4. Use of Templates:

C++ provides a strong feature of templates. These templates improve run time performance because they mostly do computation at compile time. This may increase the compilation time but still run time performance is much better than Python. For large numerical problems, expression templates are used. They solve vectors and matrices efficiently. They perform most of the matrix operations such as transposition without creating an intermediate matrix which saves memory and leads to better performance.

B. Time Analysis in Python

Python is a dynamically typed language. It has a simple syntax and is flexible. The programmer has to do less work when coding in Python which means the computer has to identify more when executing that program which makes it slow [18].

1. Reasons behind Slow Execution

The reasons behind its slow execution speed are mentioned below:

- *Dynamically Typed Language*

A dynamically typed language means it doesn't require the variable type to be mentioned. A single python variable can store any data type be it list, dictionary, int, string, etc. Therefore, when the program is executing, the interpreter checks the variable's data type every time it performs any operation to ensure the operation can be performed on that variable. This process requires more time making the execution slow [19].

- *Global Interpreter Lock (GIL)*

GIL is required in Python to prevent multiple threads to alter memory, at the same time corrupting memory. Since the Python memory manager is not thread-safe therefore, it requires GIL which prevents memory corruption by only executing a single thread at a time which slows down the speed of program execution [20].

- *High-level language*

Python is a high-level language which means it is closer to human language. It requires less effort to write code because of several reasons such as automatic memory management, garbage collection, etc [21].

- Garbage Collection

Since Python uses a garbage collector, it takes time for it to identify what memory to free. This process increases the runtime of the program.

2. Different Kinds of Python Implementation:

After all these reasons, the reason that your python code is executing slowly can be the interpreter you are using. Rick van Hattem presented a comparison of three different interpreters in his book "Mastering Python" to show the difference between the speed of execution by different Python interpreters on a test code. He used 3 interpreters namely, Python 3, Python 2, PyPy. Python 2 and 3 are CPython and concluded that for this test code, the used version of PyPy interpreter is more than 4 times faster than the CPython version used [22].

Similarly, we carried out a test. We used multiple different sets of codes and got the execution time of code for CPython and PyPy (Python 3.7) and we concluded that PyPy is usually faster than CPython due to JIT (Just In Time) Compiler which directly converts source code to machine code whereas CPython converts the source code to byte code and then it is converted to machine code.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

Table I.
Comparison between PyPy and CPython (Test Case 1)

	PyPy	CPython
Source Code	[i*j for i in range(1000) for j in range(i)]	[i*j for i in range(1000) for j in range(i)]
Execution Time	14.6 msec	49.8 msec

Table II.
Comparison between PyPy and CPython (Test Case 2)

	PyPy	CPython
Source Code	[str(i) for i in range(10000)]	[str(i) for i in range(10000)]
Execution Time	287 usec	2.97 msec

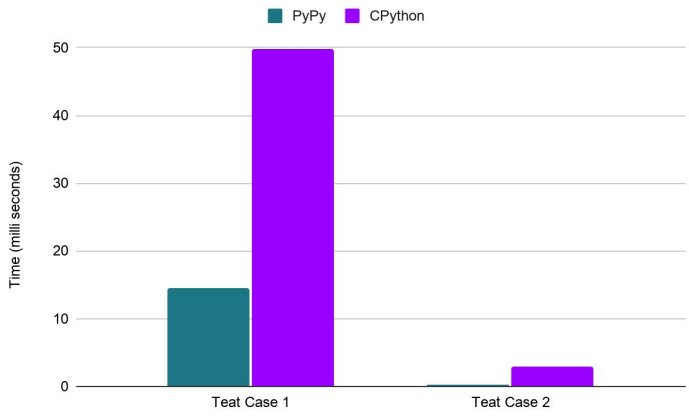

Fig. 5. Graph between PyPy and CPython speed for test cases 1 & 2

VI. MEMORY AND RUN TIME EVALUATION

A. *Algorithms:*

We have used certain algorithms in order to compare memory and time of Python and C++. Algorithms that have been compared in this research paper include sorting algorithm, searching algorithm, insertion and deletion algorithms. For the best analysis of algorithms, we have compared the best case, worst case and average case of searching, sorting, insertion and deletion algorithms.

1. *Searching Algorithm:*

In this algorithm, all the elements present in an array have been gone through to find the required element's index [23]. There are different types of searching algorithms, what we are going to compare is binary search algorithms. Middle element is used for the searching of required elements in binary searching [24]. It divides the array in half until the array is reduced to one element

Table III.
Comparison of Time and Memory of Searching Algorithm

	Time in Python (ms)	Time in C++ (ms)	Memory in Python (Kb)	Memory in C++ (Kb)
Best Case	0.19	0.094	19.54	9.65
Worst Case	0.077	0.134	19.53	10.19
Average Case	0.118	0.108	19.53	10.2

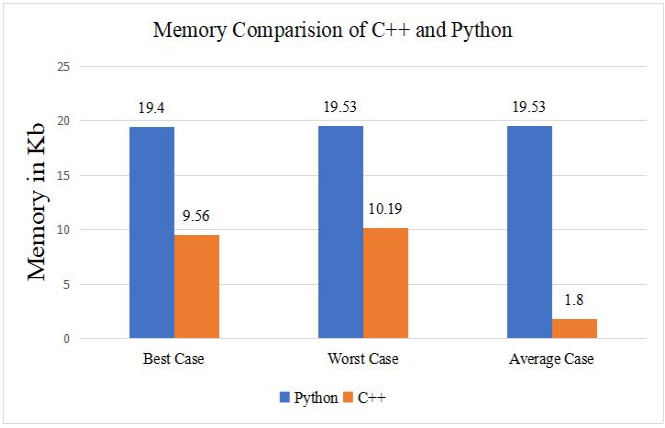

Fig. 6. Comparison of Memory Consumption of Searching Algorithm

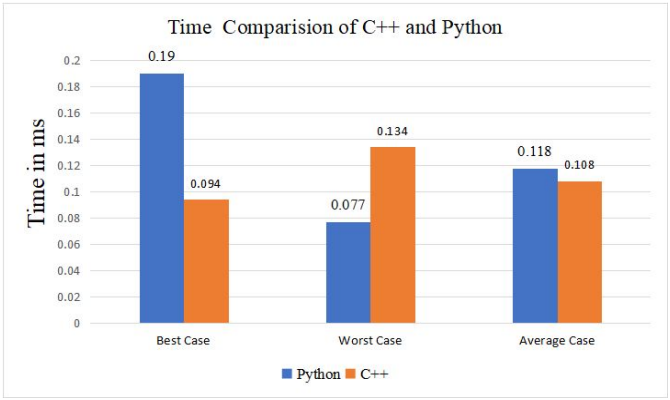

Fig. 7. Comparison of Time Utilization of Searching Algorithm

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

2. *Sorting Algorithm:*

Arranging a given set of data in a predefined manner is sorting [25]. It can be done using different algorithms, one of which is bubble sort algorithm. In bubble sorting, first two elements are sorted first and then they are compared with the third element and get sorted and so on [26].

Table IV.
Comparison of Time and Memory of Sorting Algorithm

	Time in Python (ms)	**Time in C++ (ms)**	**Memory in Python (Kb)**	**Memory in C++ (Kb)**
Best Case	0.56	0.006	11.6	4.01
Worst Case	1.06	0.011	11.71875	4.04
Average Case	0.93	0.008	11.71785	5.43

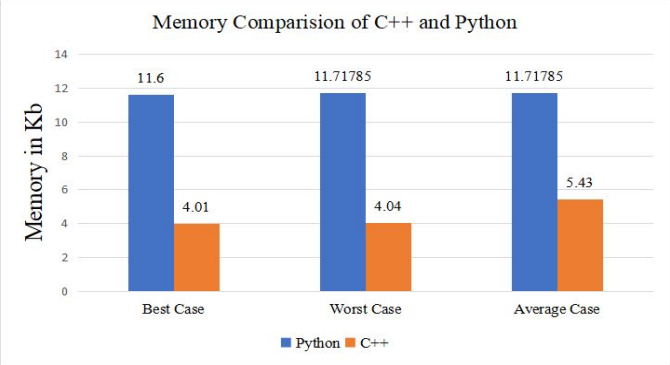

Fig.8. Comparison of Memory Consumption of Sorting Algorithm

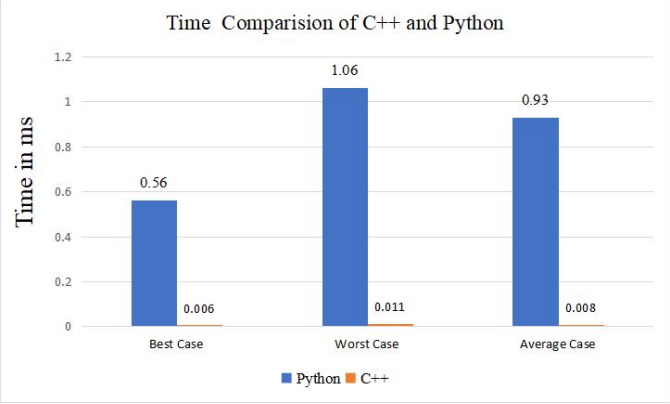

Fig. 9. Comparison of Time Utilization of Searching Algorithm

3. *Insertion Algorithm:*

Insertion is the process of adding a new element in data structure at desired location. It can be added in beginning, end, mid or any other location.

Table V.
Comparison of Time and Memory of Insertion Algorithm

	Time in Python (ms)	**Time in C++ (ms)**	**Memory in Python (Kb)**	**Memory in C++ (Kb)**
Best Case	0.024	0.005	7.2	4.01
Worst Case	0.03	0.008	7.4	4.04
Average Case	0.028	0.006	7.31	5.43

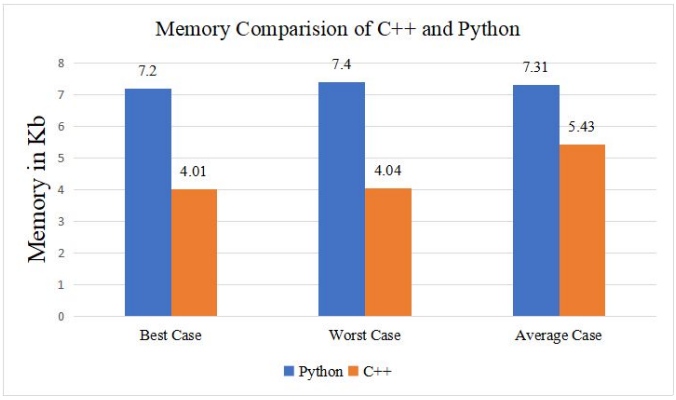

Fig.10.Comparison of Memory Consumption of Insertion Algorithm

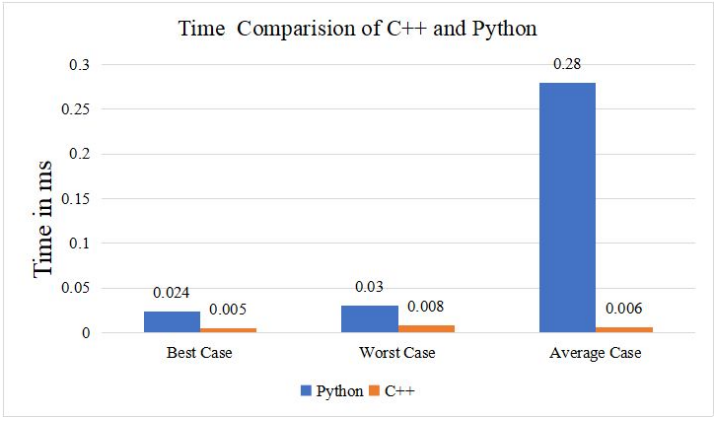

Fig.11. Comparison of Time Utilization of Searching Algorithm

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

4. *Deletion Algorithm:*

Deletion is the process in which elements are removed from the data structures and leaving an empty space for new elements. It can be done from the start of data structure, from the end, mid or other location.

Table VI.
Comparison of Time and Memory of Deletion Algorithm

	Time in Python (ms)	**Time in C++ (ms)**	**Memory in Python (Kb)**	**Memory in C++ (Kb)**
Best Case	0.010	0.004	3.906	5.67
Worst Case	0.021	0.005	7.8125	6.42
Average Case	0.013	0.0042	7.742	6.12

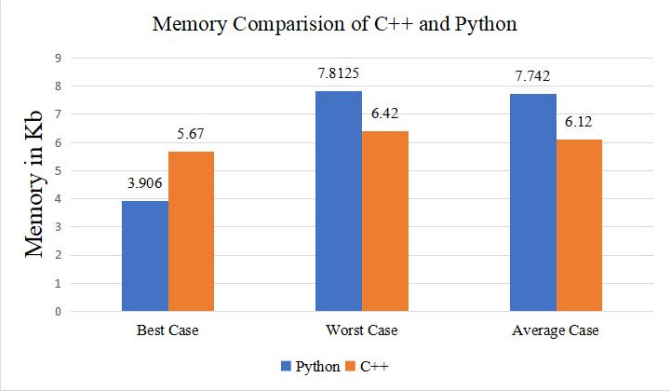

Fig. 12. Comparison of Memory Consumption of Deletion Algorithm

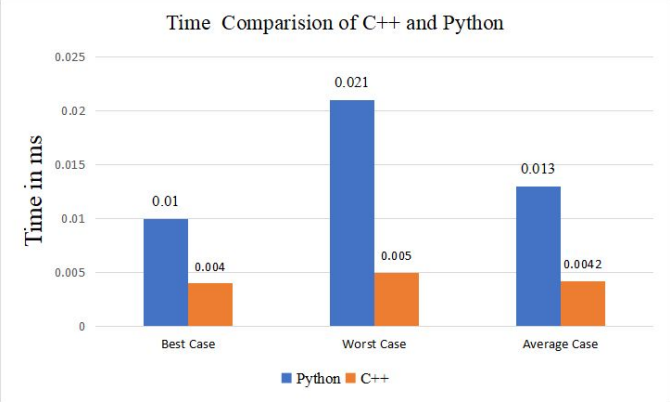

Fig. 13. Comparison of Time Utilization of Deletion Algorithm

B. *Analysis:*

Python is an advanced language and it is very easy to use. But, along with being easy and having short code, it takes more time for execution as compared to C++. C++ converts whole code into a machine code at once and then executes it. This method takes less time as compared to Python which converts code line by line to byte code and then converts it to a machine code. After conversion, it executes the program line by line. This method takes more time and makes Python slow. Moreover, Python utilizes more memory than C++ which also makes it slow. All these arguments are satisfied with the above charts and graphs.

VII. C++ OR PYTHON: WHAT SHOULD A BEGINNER PREFER?

A. *C++, It's significance and issues*

As we discussed earlier, C++ is the most used programming language. Every beginner is curious to know which language would be preferable for them to learn and which would be easier. Merits and demerits of any language should be known by the beginner so that they can start their programming journey. Benefits and issues are important to know as it can help them to choose the better one.
Everything has two or maybe more than two sides, let's reveal the faces of C++, through discussing its benefits and issues that it may cause.

1. *Significance*

- Portability

C++ enables its users to run a similar program on different operating systems making it a portable language.

- Object-Oriented Programming

Since C++ is an object-oriented language, it includes concepts like structures, inheritance, classes, polymorphism, etc. This enables programmers to reuse the code easily and better to understand. Not only this, but it also helps us to deal with the real-world problems by considering real-world things as an object.

- Memory Management

C++ provides whole control of memory to the programmer. It can have its pros and cons, either way, you take it, here the programmer has the whole command whether to clear past memory or to save it, the work of the garbage collector is done by the programmer itself. By using dynamic memory allocation with the help of pointers the memory management took place.

- Compatibility with C

C++ is way more compatible with C language. A person with the basics of the C language can easily start coding in C++. An error free code of the C language can easily be run in the compiler of C++.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 December 2020 doi:10.20944/preprints202012.0516.v1

- Scalability

Basically, the ability to scale a program is known as scalability. So, in C++ we can run either small or big programs. Code of any size, in C++ compilers, is runnable.

- Multi-paradigm

The term 'Paradigm' refers to logics, structures, etc. which is basically a style to represent a code. Hence, C++ is a multi-paradigm language because it can be represented into various styles.

2. Issues

- Complex Syntax

In comparison to other languages like Python, etc. the syntax of C++ is way more complex and lengthier.

- Security Issues

Though C++ is the best language of object-oriented programming as it has objects but due to global variables, pointers, and public specifiers, security issues still exist.

- Use of Pointers

Pointers consume a lot of memory and are hard to implement and hard to learn. Abundant use of pointers can produce wild pointers which can crash the whole program too.

- Garbage Collection

Garbage collector as discussed before in C++, is absent. The programmer must manage and clean unnecessary memory on his own.

- Lack of Rapid prototyping

The size of the code is larger in C++ due to which rapid prototyping cannot take place [8].

B. Python, It's significance and issues

Python is derived from many other languages like C, C++, Linux, etc. Nowadays, Python is most likely used language which is being preferable due to its syntax. Python got a way different syntax from C, Java, and C++. We usually do not use semicolons and brackets and when it comes to data representation like int, float, string, etc, we do not explicitly declare the data type.

1. Significance

- Easy to read, write, maintain, and learn

Python has clearly defined syntax and easy to read structure. Beginners can easily pick its keywords because the majority of them are from the English language which can easily be learned, it is English-like syntax. Python's code is minimum in size which can easily be maintained. These are the main reasons why everyone around recommends Python to beginners.

- Portability

Just like C++, Python is platform independent. You only need to write the code once and run it anywhere you desire like any platform you want just make sure that code is error-free.

- Scalable

It is scalable too. Any size of code either large or small, either of a single line would be runnable in Python.

- Vast Standard Library

The standard library in Python is massive, the functions a person can be in need of, all can be easily found in Python's library. This is a great advantage to Python's users.

- Productivity Improvement

As Python is easy to understand so the programmer focuses more on his productivity instead of checking the syntax errors and managing syntax issues. They write less code, and more functions are being performed which avoids complexity and code is manageable through which the rate of productivity is increased.

2. Issues

- Gets Slow in Speed

One of the biggest disadvantages of using Python is that it executes through an interpreter instead of the compiler like other languages which causes slow execution. The projects where time is an important part, Python is being avoided because it runs backend work at the same time of execution due to which its speed is affected and being delayed.

- Weak in Mobile Computing

Mobile applications are not often made by using Python, the reason behind this is that Python is not a good memory-efficient language and it also consumes much time which is not much suitable for mobile computing.

- Less Memory Efficient

Python consumes a large amount of memory while compiling. If memory optimization is considered so in that case, Python is avoided. To provide simplicity to developers this is a little trading developer did, a time and memory-consuming language.

- Issues in Using Other Languages

Since python is so easy to use, therefore, programmers become used to it and it becomes difficult for them to learn or work on other low-level languages. Python users may consider the addition of curly braces or semicolons as a troublesome task in other languages like C++, C#, etc [7].

C. Recommendation of Programming Language

In our opinion, Python leads in comparison with C++ and the language that we would recommend for beginners will be

Python. It is better for newcomers in terms of its easy-to-learn, and simple syntax. Furthermore, Web designing and backend-development are done with ease in python. Whereas C++ is not a very good choice for web development of any type.

Python also leads to data analysis and machine learning. While comparatively, C++ for ML is neither a good option. If we think of simplicity, Python is easier regarding every aspect and when it comes to Artificial Intelligence and Machine Learning there is no other language better than Python. A beginner could have a better understanding and can get command of this particular language easily because of its English-like syntax.

VIII. Conclusion

In this paper, we made a comparison between Python and C++ as these are the two most popular languages in today's era. We tried to figure out which language is better in terms of time and memory. First, we discussed how both the languages manage their memory. Discussion showed that C++ handles memory efficiently. It permits dynamic memory allocation and allows the programmer to have full control over the memory. A programmer can be able to allocate and deallocate memory whenever he wants. On the other hand, Python has a garbage collector that is handled by Python Runtime Environment (PRE) which collects the memory which is not going to be used in the future with the help of reference count. It reduces the work of the programmer but increases the work of the interpreter.

Then we examine time consumption by both the languages. C++ is a statically typed compiled language which is the main reason for its performance efficiency. On the other hand, a Python interpreter takes time to translate the source code into bytecode and then into machine code. Other than that, Python works on dynamic typing and solves all the type conversion issues at run time which also makes it slow.

Furthermore, we compared both the languages on the basis of sorting, searching, insertion and deletion algorithms on array data structure. We tested their best, worst and average case complexities in terms of time and space. We used Pycharm for testing Python code and Visual Studio for C++. It was evident from the graphs that Python cannot compete with C++ when it comes to memory and performance but when their simplicity in writing, understanding, and learning code was discussed, it was concluded that Python is a very powerful tool for beginners and for coding in less time, but for larger projects that need to complete processing really quick C++ is suitable.

In short, this paper concluded that both the languages hold their own importance. Beginners should prefer Python because of its syntax simplicity, readability, portability and scalability and avoid C++ as it is a complex language and hard to learn. But for speedy software and better performance like operating systems, gaming applications, one must consider C++.

IX. References

[1] Z. Alomari, O. E. Halimi, K. Sivaprasad, and C. Pandit, "Comparative studies of six programming languages," 2015.

[2] K. Stefanoski, A. Karadimche, and I. Dimitrievski, "PERFORMANCE COMPARISON OF C++ AND JAVASCRIPT (NODE.JS – V8 ENGINE)," 2019.

[3] J. Tait, T. Ripke, L. Roger and T. Matsuo, "Comparing Python and C++ Efficiency Through Sorting," 2018 International Conference on Computational Science and Computational Intelligence (CSCI), Las Vegas, NV, USA, 2018, pp. 864-871, doi: 10.1109/CSCI46756.2018.00172.

[4] M. Ateeq, H. Habib, A. Umer and M. Rehman, "C++ or Python? Which One to Begin with: A Learner's Perspective," 2014 International Conference on Teaching and Learning in Computing and Engineering (LaTiCE), Kuching, Malaysia, 2014 pp. 64-69. doi: 10.1109/LaTiCE.2014.20

[5] Z. Ashraf, "Introduction" in *How to program in C++ with 100 examples,* (volume-I), 2016, ch.1, pp.1.

[6] H. E. Ahmed, "Evaluation of C++ as object-oriented programing language compared to Java," 2018.

[7] S. J. Humer and E. C. Foster. "A COMPARATIVE ANALYSIS OF THE C++, JAVA, AND PYTHON LANGUAGES.", 2014.

[8] P. T. Bukie, C. L. Udeze, I. O. Obono, and E. B. Edim, "Comparative Analysis of Compiler Performances and Program Efficiency.", 2019.

[9] W. Wolf, "Programs" in *High-Performance Embedded Computing - Architectures, Applications, and Methodologies*, 2nd ed., 2014, ch. 3, pp. 170.

[10] G. V. Umoh., "MEMORY MANAGEMENT IN COMPUTER SYSTEM.", 2014

[11] G. K. Sengottaiyan, "MEMORY MANAGEMENT IN C++ AND JAVA." International Journal Of Engineering And Computer Science, 2015.

[12] P. Irabashetti and N. Patil, "Dynamic Memory Allocation: Role in Memory Management," International Journal of Current Engineering and Technology, 2014.

[13] R. v. Hattem, "Performance – Tracking and Reducing Your Memory and CPU Usage" in *Mastering Python*, April 2016, ch. 12, pp. 378-,379

[14] S. Romanazzi, "From Manual Memory Management to Garbage Collection", 2018.

[15] R. v. Hattem, "Performance – Tracking and Reducing Your Memory and CPU Usage" in *Mastering Python*, April 2016, ch. 12, pp. 375

[16] W. Wolf, "Programs" in *High-Performance Embedded Computing - Architectures, Applications, and Methodologies*, 2nd ed., 2014, ch. 3, pp. 185.

[17] D. Rassokhin, "The C++ programming language in cheminformatics and computational chemistry." Journal of Cheminformatics, 2020.

[18] K. R. Srinath, "Python – The Fastest Growing Programming Language", 2017.

[19] R. v. Hattem, "Decorators – Enabling Code Reuse by Decorating" in *Mastering Python*, April 2016, ch. 5, pp. 130

[20] R. v. Hattem, "Async IO – Multithreading without Threads" in *Mastering Python*, April 2016, ch. 7, pp. 176

[21] K. R. Srinath, "Python – The Fastest Growing Programming Language", 2017.

[22] R. v. Hattem, "Performance – Tracking and Reducing Your Memory and CPU Usage" in *Mastering Python*, April 2016, ch. 5, pp. 346

[23] M. Joseph and P. Keshwani, "Comparison Between Linear Search and Binary Search Algorithms", 2018.

[24] A. R. Chadha, R. Misal, and T. Mokashi, "Modified Binary Search Algorithm.", International Journal of Applied Information Systems, 2014.

[25] P. Kunjwal, "BUBBLE SORT," 2015.

[26] H. Rohil and Manisha, "Run Time Bubble Sort – An Enhancement of Bubble Sort," 2014.

[27] A. Golubin, "Memory management in Python." https://rushter.com/blog/python-memory-managment/.

X. BIBLIOGRAPHY

Farzeen Zehra is an undergraduate student of Software Engineering at NED University of Engineering and Technology. She completed her Matriculation with A+ and got 10th position in Intermediate from Karachi Board. She has a keen interest in programming and aims to become a web developer in the future.

Darakhshan is an undergraduate student of Software Engineering at NED University Of Engineering and Technology. She got 8th position in Intermediate from Karachi board. She always wants to learn new stuff as she possesses a keen interest in front-end web development. Apart from this, she also desires to become a graphic designer.

Maha Javed is a student at NED University of Engineering and Technology studying Software Engineering. She got her Matriculation degree from Federal Board with A+ and completed Intermediate from Karachi Board also with A+. Being a software developer is her dream and she works hard to fulfill her dream.

Maria Pasha completed her secondary school in Pre-Engineering and is now an undergraduate student at NED University of Engineering and Technology pursuing Software Engineering. She has experience in coding in Python, C#, and C++. Maria got a keen interest in Mathematics and art and designing but due to some circumstances, she ended up studying software engineering. She now aims to develop an interest in programming.