



A Comparative Study of C, C++, C# and Java Programming Languages: Underpinning Structure Equation Modeling, Data Structures and Algorithm

Sanja Michael Mutongwa¹, Silvanca Abeka²

¹School of Postgraduate, Department of Information Technology, University of Kigali, Rwanda

E-mail: sanja_michael@yahoo.com / msanja@uok.ac.rw

²Informatics and Innovative Systems, University of Science & Technology (JOOUST), Bondo, Kenya

E-mail: silvanca@gmail.com

Abstract Structure Equation Modeling technique has not been fully utilized in the field of Software analysis in handling complex data analysis and modeling on programming languages by a number of researches globally and specifically by Universities in East Africa, in a nut shell; secures programming practices and daily life application is still wanting. We present a comparative study of C, C++, C#, and Java Programming languages: Underpinning Structure Equation Modeling; with respect to: Runtime tests, compilation, Model comparison, Latent Curve model, rate of change over time, Runtime tests, sorting criteria and compilation. Findings indicated that Java was: (Var_Inter =0.81 and Var_slop =0.02). The overall model fit for Java had (RMR =0.000, CFI = 9.88, TLI = 1.0, and NFI = 0.99), while that of C-language was poorly fitted, its Latent Curve model results: (Var-Inter =0.41 and Var_Slop =0.31) hence not significant. Rule of thumb utilized was excellent; Java was rated higher in terms of Runtime tests, same applied to sorting criteria; Merge sort, applicabilication and compilation, While C was the least. C++ gave close results to C, Comparatively C# had better finding. On sorting criteria, Java indicated a more coherent than a C++ and C, but closer to C#, Java was rated as more simplified, Java applications are more than the 3 languages. Study recommends further investigation on C and C++, all the same there is need to carry out research on more than four languages and to use a larger sample size.

Keywords Structure Equation Modeling, C, C++, C#, Java Programming

Introduction

The object-oriented design concept has been popular for quite some time owing its success to the powerful features it offers for making program development easy and robust. On a more academic level, computer scientists search for ways to design programming languages that combine expressive power with simplicity and efficiency. The complexity of engineering software has increased dramatically in the past decade. In the early years most engineering applications were concerned solely with solving difficult numerical problems, and little attention was paid to man-machine interaction, data management, or integrated software systems. Now, computers are expected to solve a much wider variety of problems, particularly those in which numerical computations are less predominant. With the continuing increase in the variety, functionality, and complexity of engineering software, with its more widespread use, and with its increasing importance, more attention must be paid to programming language suitability so that rational decisions regarding language selection may be made. Object oriented languages like java and C# offer an instinctive way of developing programs and provide prevailing characteristics for supporting the program development Different languages support different



paradigms, provide different abstractions, and have different levels of expressive power. Some are better suited to express algorithms and others are targeting the non-technical users. The question is then what is the best programming language among the four for a particular problem. Aspects, like security and language safety, prototyping capabilities, language support for building distributed systems, and support for automating existing processes, and portability are also important issues to consider when choosing the programming language.

Overview of the Paper

The rest of the paper is organized as follows. First we introduce the programming languages under study, the language overview in terms of evolution, compact syntaxes, designing approaches and efficient execution characteristics have been explained and their development history including their important contributors have been displayed, versions, paradigms. Then we address how each language is evaluated and how memory is managed. We have so far highlighted the strengths and weaknesses for each language as well as, the applications domains in which these languages are mainly used, such as Software development (i.e. operating system, application software and antivirus (software's) creations. We have ponded strength on language compilations, their conversion source code towards executable code specifically to CPU architecture and OS. A couple of experiments have been examined; such include sorting criteria by mean of data structures and algorithm, Multiple Group Model Comparison, Latent Growth Curve conducted to compare efficiency.

The C Language: One of the most popular languages till date is the C programming language used by both novice and expert programmers. It was developed in 1972 by Dennis Ritchie and Ken Thompson at AT & T Bell Labs. Although a general-purpose programming language, its compact syntax and efficient execution characteristics have made it popular as a system programming language. It is an imperative systems implementation language. It was designed relatively straightforward compiler, C provide low-level access to memory, it also Provides language constructs that map efficiently to machine instructions and to require minimal run-time support.

A standards-compliant and portably written C program can be compiled for a very wide variety of computer platforms and operating systems with little or no change to its source code. Is suitable for systems-programming applications, hardware related applications, embedded device, chip designing, and industrial automation products. Study by [1].The research C is suitable for systems-programming applications; hardware related applications, embedded device, chip designing, and industrial automation products.

Research indicates that C is a minimalistic programming language since it can be compiled in a straightforward manner by a relatively simple compiler. C offers low-level access to memory via pointers and the ability to access specific hardware addresses. C generates only a few instructions of machine languages for each of its core language elements and does not require extensive run-time support [1]. It can be concluded that C language is suitable for many systems-programming applications that had traditionally been implemented in assembly languages' is structured oriented programming language and focuses on the procedural programming paradigm, it is relatively hard to control the large-scale program.

As C language has high level and machine level mixed programming capacity, it is used in most hardware related applications. It is very suitable for writing programs in embedded device, chip designing, industrial automation products and so forth and soon. Meanwhile, Software such as "Unix", "windows", and other antivirus can also be created by C language. It is clear that algorithm is implemented in C language easily [1].

The C++ Language: C++ is a fairly complicated object-oriented language derived from C. The syntax of C++ is a lot like C, with various extensions and extra keywords needed to support classes, inheritance and other OO features. C++ was originally developed as an extension to C. Like any other programming language. The language offers a very broad range of OOP features: multiple inheritances, strong typing, dynamic memory management, templates (generics), polymorphism, exception handling, and overloading. Its systems also offer run-time type identification and separate namespaces. The language also supports: a variety of data types including strings, arrays and structures, full I/O facilities, data pointers and type conversion [1-2].



The C++ Standard Template Library (STL) offers a set of collection and abstract data type facilities. Because it is derived from C, C++ has a number of features that support unsafe and defective software. The more recent C++ standards do support safe casts, but this feature is not yet universally available or employed. Also, C++ has dynamic memory allocation, but does not have garbage collection; this allows programs to misuse and leak memory. C++ also supports dangerous raw memory pointers and pointer arithmetic. These low-level facilities are useful in some situations, but can increase the time needed for software development. C++ is widely available on the WWW, but language has no official home on the Web. Many C++ implementations exist; some of them follow the old tradition of translating C++ into C, while others are native compilers.

C++ provides the cryptic C shortcuts that run counter to clarity, and they are commonly used. When C++ is being used to create object-oriented code, the programmer has good object-oriented features to facilitate maintainability. However, the C problem of little inherent support for maintainability still remains in other C++ language features. Mixed language support: C++ will readily use object files produced by any language compiler as it composes an application. This is easy because C++ requires no consistency checking among these separate files. While that makes the object files easy to use, it does not provide specific support for properly interfacing the languages or for verifying correct exchange of data across the established interface. C++ improves on C with better language constructs for facilitating language interfacing.

C++ does not yet have an existing standard, but, when it does, it will probably not alter the C characteristics in this respect. Common practice will not necessarily adhere to the standard. However, C++ does encourage the encapsulation of dependencies, a feature which facilitates portability. C++ tools and tool sets are also widely available on many platforms. C++ improves considerably on the language characteristics of C for supporting reliability with features such as encapsulation, as well as improved expression

Comparison of Programming Languages

The C# Language: (pronounced "See Sharp") is a multi-purpose computer programming language suitable for all development needs. "The primary architects of C# were Peter Golde, Eric Gunnerson.

The principal designer of the C# language was Anders. Hejlsberg, a lead architect at Microsoft." C# was designed to be a pure object oriented programming language. "C# debuted in the year 2000 at the Professional Developers Conference (PDC) where Microsoft founder Bill Gates was the keynote speaker. At the same time, Visual Studio.NET was announced." Although C# is derived from the C programming language, it has features such as *garbage collection* that allow beginners to become proficient in C# more quickly than in C or C++.

Similar to Java, it is object-oriented, comes with an extensive *class library*, and supports exception handling, multiple types of polymorphism, and separation of interfaces from implementations. Those features, combined with its powerful development tools, multi-platform support, and *generics*, make C# a good choice for many types of software development projects: rapid application development projects, Projects implemented by individuals or large or small teams, Internet applications, and projects with strict reliability requirements. Its strong typing helps to prevent many programming errors that are common in weakly typed languages [2].

The JAVA Language: The Java language was developed in 1991 by James Gosling of Sun Microsystems and released in 1995. The latest release of Java is the Java SE 8. Java is a write once run anywhere type of programming language. It is a secure, high performance and portable object oriented programming language. Java was started as a project called "Oak" by James Gosling in July 1991. Java is portable OO-language, simple, and designed at Sun Microsystems labs by research staff and originally developed by James Gosling. Virtual Machine (JVM) regardless of computer architecture Java has three different forms, Java2 Standard Edition (J2SE), Java2 Micro Edition (J2ME), and Java2 Enterprise Edition (J2EE). J2SE is suitable for the desktop applications. J2ME is proper for embedded systems development for mobile phones, wireless application and PDA programming.

Java was started as a project called "Oak" by James Gosling in July 1991. Java is portable OO-language, simple, and designed at Sun Microsystems labs by research staff and originally developed by James Gosling. Java syntax fairly similar to C++, also Java borrows ideas from Mesa, Objective-C and Modula-3. Some features of Java, Java has inheritance, exception handling, modularity, strong type checking, garbage collection,



polymorphism and etc. The newest version of Java is Java platform 6, specially this version includes nested classes, reflection, and persistence as well as many additional standard libraries. The *class* in java is the fundamental structure component.

Java standard library includes extensive I/O facilities, date/time support, cryptographic security classes, distributed computation support, GUI toolkit, and system interfaces. Additionally to normal application development by java, Java is used to develop embedded programs, called 'applets', for web browsers and other Java enabled platforms. This capability is an important part of Java, and the standard library packages include a security manager to restrict the capabilities of Java applets. These applet facilities were important to Java's widespread adoption and popularity.

Its compiler implementations should be commensurate with the current state of technology. Java is very sophisticated for such a new language, and it is driving some of the current technology trends. Its Sun compiler implementation is commensurate with current technology. Appropriate software engineering-based supporting tools and environments should be available. Current Java tool kits contain primarily tools to support code creation, although some software engineering-based tool kits are beginning to appear. Readability: Java is strictly object oriented, so its form is very well defined.

The code suffers somewhat from the cryptic C syntax forms. Many features of Java support maintainability, such as those which support code clarity, encapsulation, and object orientation. Object-oriented capabilities can have both good and bad effects on maintainability, but, if used properly, object-oriented programming will improve maintainability. Mixed language support: Java provides for interfacing with other languages by providing wrappers around the code from the other languages.

Java was built for complete portability. Its compiler produces source code in a platform-independent bytecode. The byte code is then translated at runtime into native machine code for the given platform. Reliability: Java requires the specification of information, the omission of which can make a program unreliable, such as type specifications. Java supports reusability with language features supporting code clarity (making code understandable), encapsulation (making code adaptable), maintainability, and portability. Safety: Java was not developed for safety-critical systems, and its capabilities in that area are unproven.

Analysis by Latent Growth Curve

Finding on variances as indicated in Figure 1. ($Var_{Inter} = 0.81$ and $Var_{slope} = 0.02$) were statistically significant, indicating significant individual variability in the initial level and rate of change (growth) in the use of SYSTEM across the four waves of measurement. The covariance ($Cov_{InterceptSlope} <---> = 0.44$) between the two latent factors is highly Significant at the two-tail level, $p < 0.000$, indicating the initial level and rate of change over time are strongly related

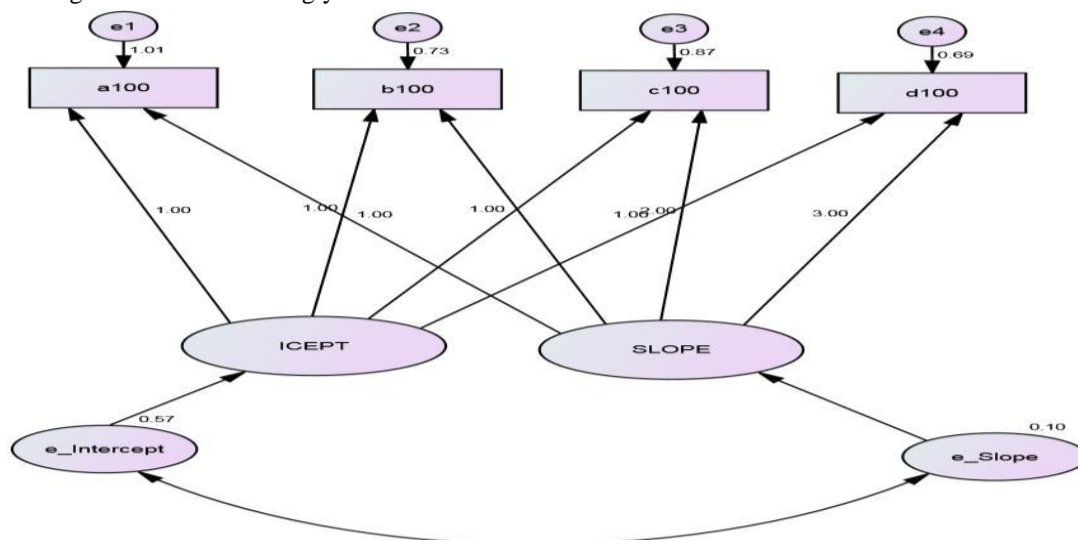


Figure 1: Latent Growth Curve (Source: Sanja., M., M., (2017) Testing SMAUIT Model



We analyzed the Sub-Models of IT Infrastructure by means of latent growth curve it indicating that the mean intercept value obtained was 1.35. COITI sub model indicated that the average starting period of the system use was 1.35 units. The mean slope value was 0.11 units. The correlation between the intercepts and the slopes was 2.09 which signifies that there was an association between Intercept and slope. Results indicate that the means were *statistically significant* when tested with the null hypothesis that their true values are zero in the population collected from Ministry _C, were the sample was drawn.

Study done by [5, 9, 10] while testing SMAUIT model, utilized, with a technique of Impulsive Decision Making Scale, were by Java Programming in the system was paramount ,since it indicated strong significant, it also showed overlap on security parameter for the system. C# categorically was also significant as a language ,the finding conquers with that of [6, 9] in their study, in both studies Impulsivity was said to a predisposition toward rapid, unplanned reactions to internal or external stimuli of the system software’s with diminished regard to the negative consequences of these reactions to the impulsive on system uses [6, 7] While multiple scales have been developed to assess self-reported impulsivity, the Barratt Impulsiveness Scale (BIS) is arguably the most widely used, Another study by [11] whose aim was to investigate in detail the relationships between actual and potentially problematic use of the mobile phone with regard to the various components of the UPPS impulsivity model, gave similar results with this research.

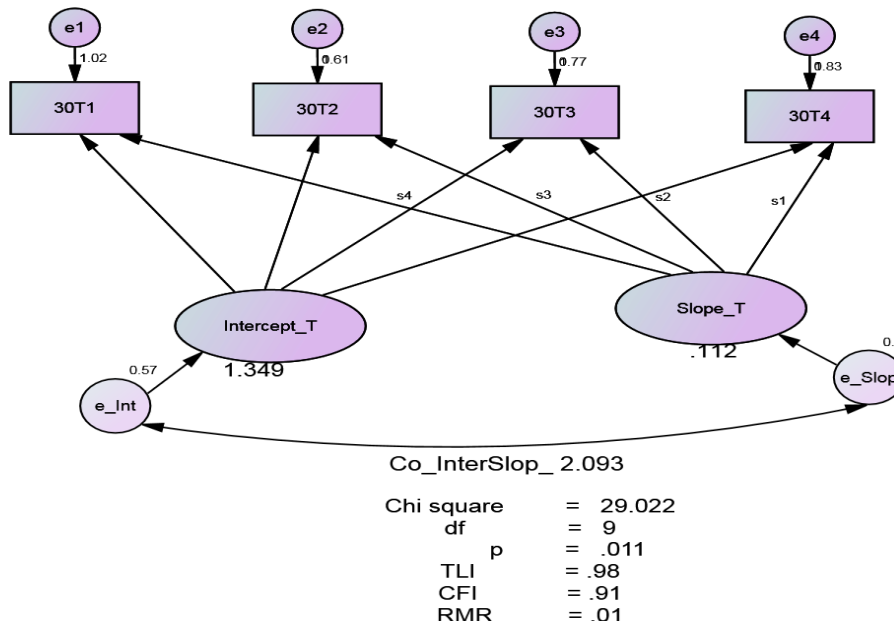


Figure 2: Latent Growth Curve (Source: Sanja., M., M., (2017) Testing SMAUIT Model)

We contend that the use of system at is expected to increase by .11 each operated time period, beginning with an average score of 1.35. Such a trend basically indicate higher rate of increase compared to the use of system at the counties. Study done by [4],was similar to this study , the technique of Impulsive Decision Making Scale, of Java Programming predominately indicated reliable results which indicated strong effect on system performance ,unlike C# security on Java was good it had a significant different ,a departure from p=0.05 Java language was strong whose p =0.01 ,the finding conquers with that of [4, 6] in their study, in both studies Impulsivity was said to a predisposition toward rapid, unplanned reactions to internal or external stimuli of the system software’s with diminished regard to the negative consequences of these reactions to the impulsive on system uses [8]

Table 1: Latent Growth Curve

Intercept	Estimate	S.E	C.R	P
Intercept_T	1.349	0.045	29.737	0.0001
Slop_T	0.112	0.020	5.67	0.0001
Covariance	Estimate	S.E	C.R	P
e_slop ↔ e_Intercept	2.093	2.165	0.967	0.334

Analysis by Multiple Group Model Comparison

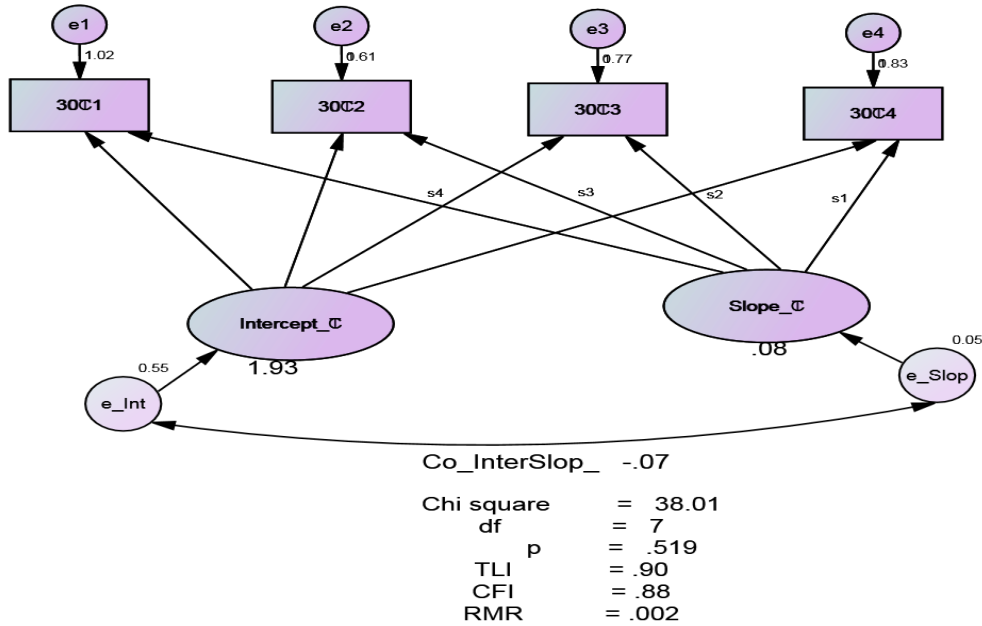


Figure 3: Latent Growth Curve (Source: Sanja., M., M., (2017) Testing SMAUIT Model)

Multiple Group Model Comparison

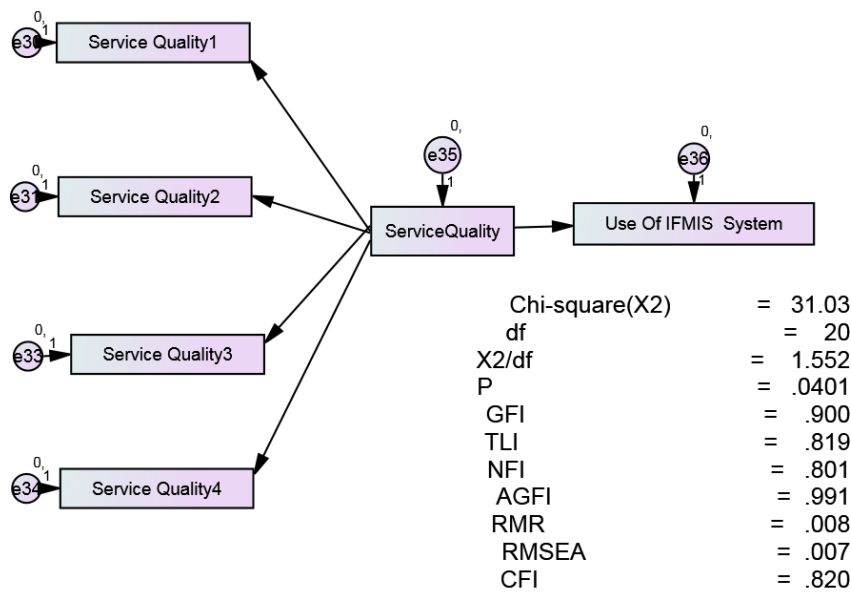


Figure 3: Multiple Group Model Comparison (Source: Sanja., M., M., (2017) Testing SMAUIT Model)

Table 2: Summary of Overall Model Comparison

Scale	Model_1 TOS	Model_2 MgtS	Model_3 COITI	Model_4 InfQ	Model_5 SystQ	Model_6 ServQ
Chi-Sq(X ²)	26.913	20.013	29.014	33.87	20.145	31.03
Df	7	9	5	27	13	20
P	0.013	0.000	0.643	0.000	0.0301	0.0401
X ² /df	3.844	2.224	5.8028	1.254	1.550	1.552
GFI	0.919	0.810	0.812	0.932	0.911	0.900
TLI	0.991	0.900	0.890	0.944	0.913	0.819
NFI	0.951	0.912	0.900	0.802	0.899	0.801
AGFI	0.813	0.899	0.802	0.971	0.903	0.991

RMR	0.019	0.031	0.067	0.014	0.011	0.008
RMSEA	0.022	0.001	0.055	0.005	0.020	0.007
CFI	0.901	0.907	0.801	0.909	0.990	0.820

Analysis by Sort Criteria

Selection Sort by C#

An Array Class Test Bed: To examine these algorithms, we did a test bed in which to implement a class that encapsulates the normal operations performed with an array—element insertion, element access, and displaying the contents of the array. Here's the code:

```
class CArray {
private int [] arr;
private int upper;
private int numElements;
public CArray(int size) {
arr = new int[size];
upper = size-1;
numElements = 0;
}
public void Insert(int item) {
arr[numElements] = item;
numElements++;
}
public void DisplayElements() {
for(int i = 0; i <= upper; i++)
Console.Write(arr[i] + " ");
}
public void Clear() {
for(int i = 0; i <= upper; i++)
arr[i] = 0;
numElements = 0;
}
}
static void Main() {
CArray nums = new CArray();
for(int i = 0; i <= 49; i++)
nums.Insert(i);
nums.DisplayElements();
}
```

The output looks like this:

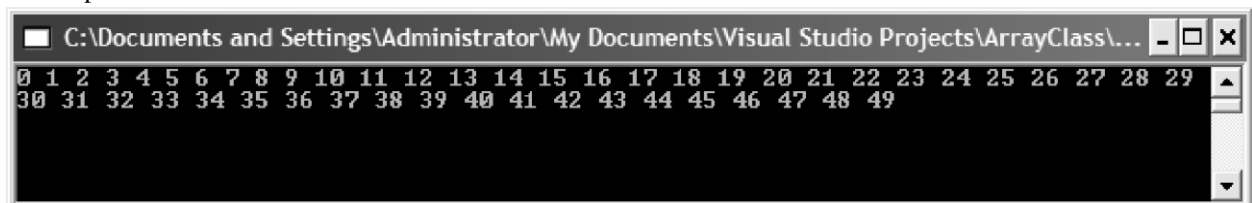


Figure 4: Sorting by C#

Sorting by C# is best achieved by using a random number generator to assign each array element to the array. Results on C# sorting is about data which is in array random order C# creates class hence generates random objective, results also show that C# easily stores numbers. Study by [11] indicate that, C# Sorts arrays by more sophisticated rules, such rules include **delegate to anonymous method**. We content that generic delegate



`Comparison<T>` is declared as public delegate `int Comparison<T>(T x, T y)`. It points to a method that compares two objects of the same type. It should return less than 0 when $X < Y$, zero when $X = Y$ and greater than 0 when $X > Y$. The method (to which the delegate points) can be also an anonymous method [11].

We demonstrate how to **sort an array of custom type** using the delegate to anonymous comparison method.

The custom type in this case is a class `User` with properties: `Name`, `Custom type` and `Age`.

[C#]

```
// array of custom type
User[] users = new User[3] { new User("Betty", 23), // name, age
                             new User("Susan", 20),
                             new User("Lisa", 25) };
```

[C#]

```
// sort array by name
Array.Sort(users, delegate(User user1, User user2) {
    return user1.Name.CompareTo(user2.Name);
});
// write array (output: Betty23 Lisa25 Susan20)
foreach (User user in users) Console.WriteLine(user.Name + user.Age + " ");
```

[C#]

```
// sort array by age
Array.Sort(users, delegate(User user1, User user2) {
    return user1.Age.CompareTo(user2.Age); // (user1.Age - user2.Age)
});
// write array (output: Susan20 Betty23 Lisa25)
foreach (User user in users) Console.WriteLine(user.Name + user.Age + " ");
```

Selection Sort By Java

```
public static void selectionSort(int[] list)
{
    for (int index = 0; index < list.length - 1; index++)
    {
        int minIndex = findMinimum(list, index);
        if (minIndex != index)
            swap(list, index, minIndex);
    }
}
```

Analysis by Merge Sort On Java

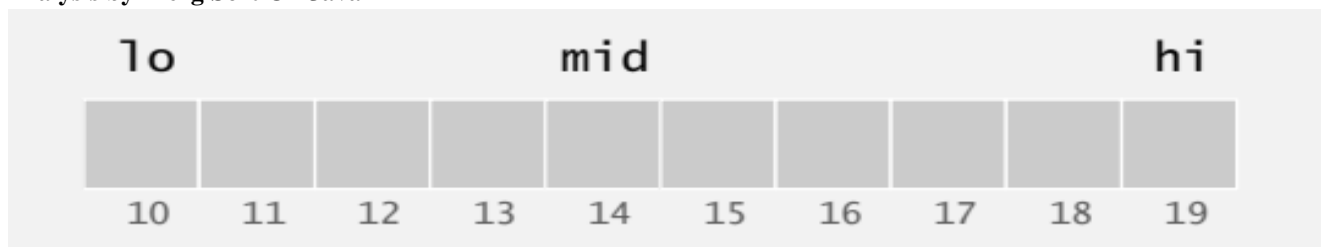


Figure 5: Sorting by Java

Finding with Java indicate that, it performs both merge sort, easily handles recursive calls, takes quick curve numbers



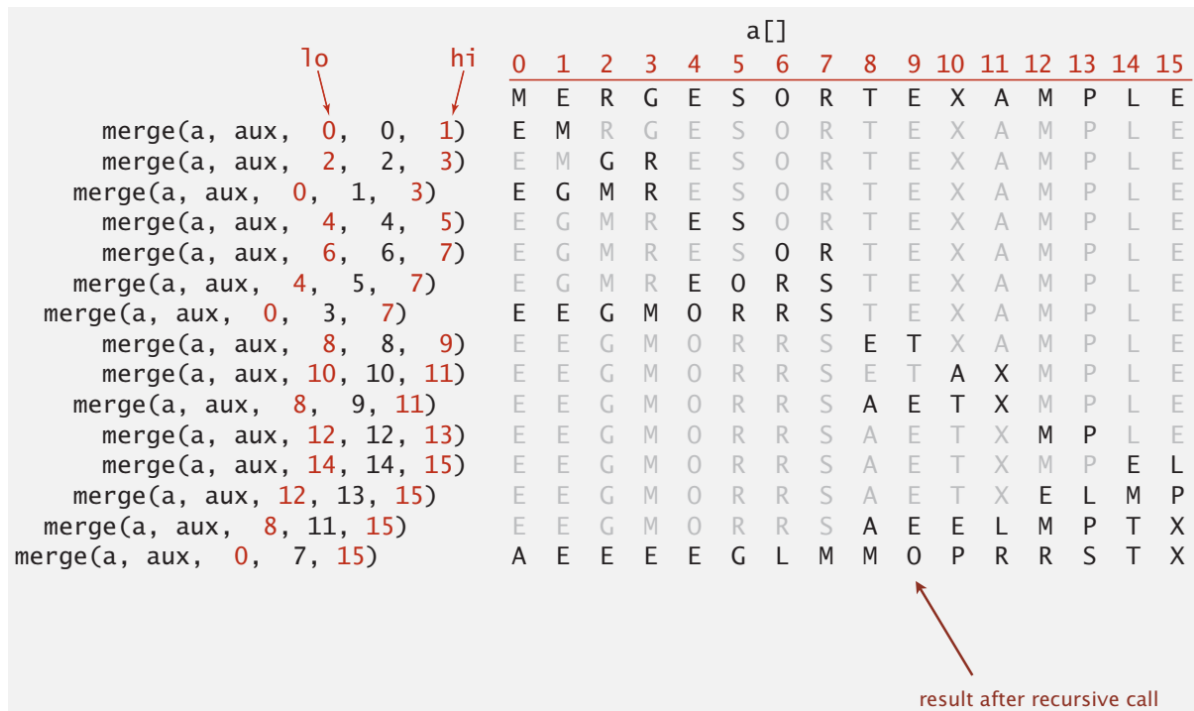


Figure 6: Sorting by Java

Java practically simplifies, Sort the growth rates from slowest to fastest growth

$O(n)$	$O(n \log(n))$
$O(n^3)$	$O(2^n)$
$O(n^n)$	$O(\sqrt{n})$
$O(\log(n))$	$O(n\sqrt{n})$
$O(n^2 \log(n))$	$O(n^{\log(n)})$

Practical set and sorting of the above shows that If 3 000 1 000 9 , , 2 2 = , the algorithm would take nine times as long, or 45 seconds, to handle a data set of 3,000 records. Java is able to remove all duplicates from an array, Counts number times it occurs, with the **Big-Oh** time estimate array are able to method in terms of n , the length of a ? Use the “light bulb pattern

Conclusion

Generally, the rule of thumb utilized was excellent, it fits for the model of Java and C# ; Java was rated higher in terms of Runtime tests same applies to Sorting techniques; Merge sort, applicability and compilation, while C was the least. C++ gave close results to C: (RMR =0.15, CFI = 7.08, TLI = 8.8, and NFI =0.90), it was below the threshold: On sorting criteria Java indicated a more coherent than a C++ and C language. Java was rated as more simplified, with operation rates dually progressively increasing higher from low to high growth; it is also able to remove duplicates from arrays, hence able to perform method for application on electrical appliances such as “light bulb pattern.

Utilizing a technique of Impulsive Decision Making Scale, Java Programming in the system was paramount, since it indicated strong significant, it also showed overlap on security parameter for the system .Java is highly standardized, strongly typed, and has a rich set of APIs that make it easy to write programs. However, Java is quite verbose. Further, one needs to really understand object oriented programming to make the most effective use of Java. For people used to programming in C/C++, Java might seem restrictive since it does not use pointers at all, and it does not provide the "raw" power of C/C++ C# categorically was significant as a language in terms of Impulsivity it was said to a predisposition toward rapid, unplanned reactions to internal or external

stimuli of the system software's with diminished regard to the negative consequences of these reactions to the impulsive on system uses.

Results on C# sorting is about data which is in array random order C# creates class hence generates random objective, results also show that C# easily stores numbers. C is suitable for systems-programming applications, hardware related applications, embedded device, chip designing, and industrial automation products. Java is able to remove all duplicates from an array, Counts number times it occurs, with the **Big-Oh** time estimate array are able to method in terms of n , the length of a ? Use the "light bulb pattern

Reference

- [1]. Hao chen (2010).Thesis a Vaasanammattikorkeakoulu University of Applied Sciences Degree Program of Information Technology
- [2]. Lu, I., R., R.; Kwan, E.; Thomas, D. R.; & Cedzynski, M. (2011). Two new methods for estimating structural equation models: An illustration and a comparison with two established methods. *International Journal of Research in Marketing*. 28(3): 258-268.
- [3]. Malthouse, E. C., A. C. Tamhane Chin, W., W., Marcolin, B., L., and Newsted, P. R. 2003. "A Partial Least Squares Latent Variable Modeling Approach for Measuring Interaction Effect: Results from a Monte
- [4]. Sanja Michael Mutong'wa, Anthony J Rodrigues, Samuel Liyala ,(2017) Analysis of Partial Least Squares on Split Path Models: Underpinning Delone & Maclean Theory, *Journal of Scientific and Engineering Research*, 2017, 4(9):453-463, ISSN: 2394-2630 CODEN(USA): JSER
- [5]. Sanja Michael Mutong'wa,, (2017), SMAUIT Computing Model: Driving Integrated Financial Management System, In Public Sector; School of Informatics and Innovative Systems, Jaramogi Odinga Oginga University of Science and Technology, Kenya
- [6]. Moeller, F., G., Barratt, E.S., Dougherty, D.M., Schmitz, J. M., & Swann, A.C. (2001).Psychiatric aspects of impulsivity. *American Journal of Psychiatry*, 158, 1783–1793.
- [7]. Collins, M., J., and A.,W., Frankle (2008). International cash management practices of large U.,S., firms. *Journal of Cash Management*. 5:4, 42-46
- [8]. H .,Chen, "Comparative Study of C , C ++ , C # and Java Programming Languages Degree Program of Information Technology", vol. 2, no. 5, pp. 11–39, 2010.
- [9]. J., E., Sammet, "Programming Languages: History and Future", *Communication of the ACM*, vol. 15, no. 7, July 1972.
- [10]. J., Paquet and S. A. Mokhov, "Comparative Studies of Programming Languages; Course Lecture Notes", p. 66, August 2010.
- [11]. Terrence W.Pratt, Marvin V. Zekowitz and Tadepalli V. Gopal, "Programming Languages, Design and Implementation", 4th Edition, Pearson, 2001.

