Research Journal of Health, Medicine and Nursing
Vol. 1 Issue 1 February 2024
ISSN: PENDING

PROGRAMMING EDUCATION IN MAURITIUS: A CRITICAL ANALYSIS OF LOWER SECONDARY CHALLENGES AND STRATEGIES

[1]Aisha Fadilah Rajput, [2]Ravi Harish Choudhury and [3]Leena Krishnan Nair
[1,2,3] Mauritius Examinations Syndicate, Mauritius.

Abstract: The realm of programming education encompasses a diverse range of components, including proficiency in programming languages, pseudocode, algorithms, flowcharts, and problem-solving skills. Traditionally, pedagogical approaches in programming have involved establishing foundational knowledge and progressively building complexity to foster effective skill development among students. Central to programming curricula is the cultivation of problem-solving acumen and computational thinking skills. Recognizing the significance of these foundations, the Code Craft project was initiated to bolster the acquisition of fundamental programming skills and understanding of data structures.

This report focuses on the unique context of teaching and learning programming at the lower secondary school level in Mauritius, encompassing grades 7 to 9. A notable gap exists in the literature, as no prior studies have investigated the nuances of programming education within this educational stratum. Notably, Mauritius, as revealed by the Survey of ICT and Education in Africa: Mauritius Country Report (2007), introduced Information and Communication Technology (ICT) into secondary schools in 1991. However, its integration faced challenges, particularly in securing a substantive position within the secondary education curriculum, where it was not considered a primary subject until the implementation of the national assessment of education.

One impediment to effective programming education in Mauritius was the scarcity of qualified Computer Science educators. Many educators in this domain primarily hailed from information systems or Information Technology backgrounds. Consequently, the lack of specialized educators hindered the seamless integration of programming education into the curriculum. Previous attempts by computer educators to mitigate these challenges yielded suboptimal results, with students failing to achieve satisfactory performance levels.

This report aims to fill the existing gap in research by investigating the dynamics of teaching and learning programming at the lower secondary school level in Mauritius. By delving into the historical context, challenges, and strategies employed thus far, the study seeks to contribute valuable insights that can inform the development of effective educational strategies in this domain. As Mauritius grapples with the evolution of ICT education, understanding the specific challenges and opportunities at the lower secondary level is crucial for refining educational approaches and ensuring the successful integration of programming skills into the broader curriculum.

Keywords: Programming Education, Lower Secondary School, Computational Thinking, Curriculum Development, ICT Integration

INTRODUCTION:

The theory of programming comprises of knowledge of any programming language, pseudocode, algorithm, flowchart and problem-solving skills. The regular method in programming is to initiate the basics and construct in programming so as to effectively channel the students towards developing programming skills. The curriculum related to programming usually stresses on developing problem solving and computational thinking skills. The

code craft project was initiated to promote the learning of basic programming skills and data structures. To have an effective concepts and educational strategies for the usage of programming it is vital to review the various studies carried out in this domain.

This report emphasizes on the teaching and learning of programming at lower secondary school
(Meaning grade 7 to 9) in Mauritius as no such studies have been carried out. According to a Survey of ICT and Education in Africa: Mauritius Country Report (2007) Information and Communication Technology (ICT) has been introduced since 1991 in secondary schools. It was not immune to problems and it has a weak position in the secondary education curriculum where it was not considered as a main subject until the national assessment of education was implemented. Another setback of teaching ICT was the lack of Computer Science educators as they were mainly from the information systems or Information Technology background. The computer educators have tried out several strategies to tackle the pitfalls of teaching and learning programming however the students did not perform satisfactorily.

1. Mauritius Education System

The Mauritian Education System has structured its pathway to easily monitor the performance of the students in an incremental manner. The progression of education is initiated from the age of 3 to 5 as pre-primary to tertiary after 18 years by going through primary and secondary schools.

The implementation of the NYCBE (Nine Year Continuous Basic Education) in 2018 has transformed the Mauritian education system. The educational reform aims at equipping the students with knowledge and skills in achieving the 2030 ministry of education vision. The primary level students are assessed after 6 years to obtain their level one certification and secure admission to secondary schools.

During the secondary level schooling the students will be examined after their grade 9 to obtain their level 2 certification. After the assessment the students will secure admission in academies, regional or vocational schools. Having completed their grade 13 students are entitled for tertiary education.

2.1. Computer Education in Mauritius

According to the Ministry of Education Arts and Culture (1991) an entire chapter was dedicated to computer education in schools. The plan stated that implementation of computer usage in schools occurred during 1982-83 and in 1986 the Ministry of Education initiated Computer Literacy which consisted of the computer operations, programming and some common application software like Microsoft Word, Excel, PowerPoint as a pilot study in 10 secondary schools around the island in Form 3. The implementation of computer education has not been implemented successfully because of lack of resources and practical sessions. Another pitfall was the lack of trained educators and a solution to this problem was to provide Bachelor of Education course in computer education by the Mauritius Institute of Education. The ICT subject was being assessed at school level at that time. According to the Ministry of Education plan 2012, the Mauritius Examination Syndicate (MES) is responsible to carry out the National Assessment at Form 3 (today known as Grade 9). The computer education (also known as Information Communication Technology (ICT)) in Mauritius exhibits some fragile points such that the curriculum does not elaborate the skills and competencies which students are supposed to acquire in order to guarantee its standard as it was an optional subject at School Certificate level. Report from the Quality Assurance Division (QAD) of the Ministry of Education clearly show that there has been a decrease in the percentage pass from the last decade. As regard to the item analysis for the ICT assessment, it is clear that the decline is mainly due to problem solving and programming concepts.

2.1.1. Programming Education

Programming is one of the nine content areas that are being assessed in ICT at lower secondary in Mauritius as the curriculum has been design to allow the learners developed their ICT skills and competencies to face the challenging technological world. It helps to demonstrate problem solving and logical reasoning skills through computational thinking.

Programming involves a number of stages in coding, debugging and maintaining the source code of a program (Wikipedia, 2020). Mawby, Kurland, Pea, and Clement (2009) defined programming as an expertise which is difficult to study, however, if effective teaching strategies are employed, it can help students to master the concepts easily. This report will emphasize on investigating the challenges associated with teaching programming. The aim is to support the educator as well as the students with an appropriate method that can support students to amusingly cram and sustain educators in the learning process by monitoring students' performance (Stephenson, Gal-Ezer, Haberman, & Verno, 2015).

2. Aim of the Study

Computer programming has been a compulsory content in the ICT curriculum since the implementation of the national assessment in Mauritius. This initiation of programming at grade 7 has brought about many pitfalls such as educators requiring appropriate textbooks, hardware with appropriate programs and programming skills to facilitate the initiation of programming. However, the introduction of programming such as scratch is an important technological development in problem solving and critical thinking in ICT.

This report will provide solutions to the different pitfalls and aims at:

- Identifying the drawbacks related to the teaching and learning of programming.
- Integration of computer game learning such as scratch thereby engaging students in learning programming

3. Pedagogical content knowledge of programming

Pedagogical Content Knowledge (PCK), is a theory defined by Koehler and Mishra (2009) whose techniques demonstrate and express the content area of a subject and thus making it understandable. The PCK incorporates an appreciation of the difficulty level of the different learning outcomes (Shulman, 2006). From various studies it can be concluded that there are various types of learning styles to satisfy the needs of the learners (Rayner, 2015). The answers of the main questions of the PCK for programming are shown in Figure 1.

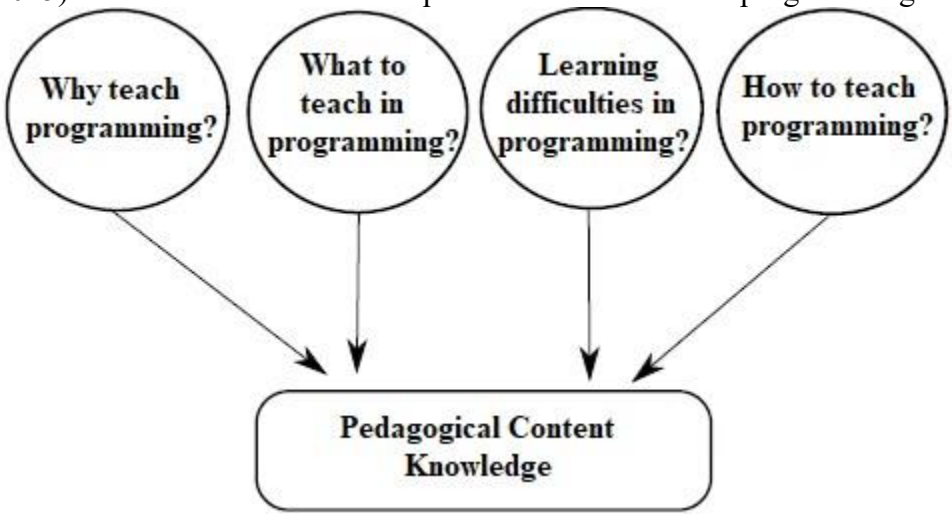

Figure-1.

An adaptation of Grossman's reformulation of PCK.

4.1.1. Why Teach Programming?

Why do we need to teach programming at lower secondary level? This question should have been better formulated as "why should the students learn programming?" and the outcome of this question will be of interest from an educator's point of view. This will therefore motivate educators in promoting ICT and help students to better engage in computer or Information Technology (IT) related courses at tertiary level. However, this is not the goal of this paper.

Papert (2006) highlighted the importance of studying programming as it helps students to better develop their problem-solving and critical thinking skills. Students have to undergo a number of stages in achieving the solution for a particular problem by adopting different programming syntax.

4.1.2. What Should Be Taught?

The answer to this question will help us formulate the key concepts of programming which is about finding solutions to problems. At lower secondary level programming key concepts involve flowcharts and algorithms. These are applied in problems where learners have to translate their programming skills into coding solutions to problems. Different programming tools have been adopted as learning strategies to facilitate the teaching and learning of programming (Resnick et al., 2009). These are shown in the Table 1:

Table-1.

Current Programming Teaching Tools used in the ICT curriculum.

Language	Description	Date	Shortcoming
Logo	It is an educational tool that was used to teach learners programming concepts related to Lisp.	1967	The idea of assessment for learning is missing from those tools. There is no consideration of the learners' preknowledge of programming. Monitoring the progress of learners is also missing from these. Programming tools.
Scratch	It is a graphical tool that can be used by pupils to make animated stories, games and so on.	2006	

4.1.3. What are the Learning Difficulties*?*

Understanding and coding applications is usually thought to be tough and the chapters related to programming and algorithms have led to increase in student's dropout percentage. Soloway and Spohrer (2009) pointed that a student needs at least 10 years of programming experience to master it properly. Various researchers have conducted different studies on the differences between objectoriented and procedural languages where C++ and Java are extensively used as educational languages. The answer for this question will help us identify the different drawbacks for learning programming and Govender (2006) identifies these difficulties:

1. Discover programming is useful and how beneficial it will be when studying programming.
2. Understanding the syntax of programming.
3. Understanding the concept of programming construct in term of sequence, selection and iteration.
4. Grasping the key concepts of algorithm, flowchart and pseudocode.

4.1.4. How to Teach Programming?

The answer to this question will serve to recognize the different strategies used to initiate programming, deal with complexities involved in teaching and eventually motivating students to enjoy programming. Hromkovi˘c (2006) pointed out that programming can be seen as a communication skill in problem solving by setting instructions. It is a method to initiate students in writing simple syntax in any programming language. Thus, writing instructions helps in problem solving which is directly related to algorithms.

4. Taxonomy of Education

According to Forehand (2010) Bloom's taxonomy attributes help educators in arranging their educational goals and teaching strategies. In addition, Bloom has identified cognitive, affective and psychomotor as the educational goals (Krathwohl, 2012). They updated the six levels in the taxonomy based on feedback from teaching practitioners and their interactions with students, from lowest to highest, as Remembering, Understanding, Applying, Analysing, Evaluating and Creating as shown in Table 2.

Table-2.

Bloom's categories and programming directives.

Bloom's Categories	**Programming directives**
Remember	Can the student remember the syntax of example an iteration?
Understand	Can the student remember the operation of example an iteration?
Apply	Can the student implement example iteration?
Analyse	Can the student differentiate between iteration and sequencing?
Evaluate	Can the student decide whether it is better to use sequencing or iteration in the given question?
Create	Can the student design an application?

Thompson (2008) also developed Bloom's taxonomy, citing the difficulties in applying the levels of cognition to software engineering and programming; in their work, the categories were explained using examples specific to programming. Therefore, this taxonomy has been used in the teaching of programming to allow learners in recalling the information from memory that are needed for writing a code and understanding the syntax of a structure used in the program. For example, students at this level should be asked to name the different types of iteration. In short, learners at this level are expected to recall from memory what they have learnt in the classroom.

5. Learning Styles

Rutherfoord and Rutherfoord (2008) outlined that learning styles are the desired methods of studying new concepts that will help in understanding and retaining contents. According to Stickel (2009) the different learning styles of the students has an impact on the teaching strategies of the educators as they have limited time devoted for the preparation of their lessons. As learning styles are dynamic, we will concentrate on the following Figure 2.

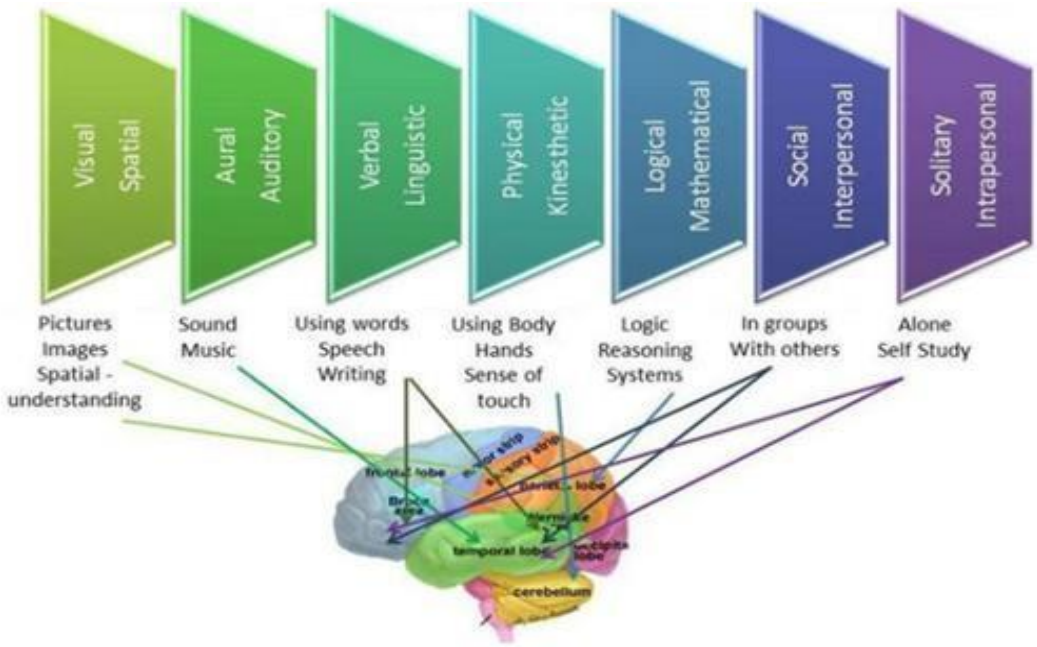

Figure-2.
Learning styles.

Learning to code has many layers from learning the actual programming languages to learning how to think like a programmer. Complicating the learning process is the fact that everyone learns differently.

There is no one correct way for learners to learn. Some learners are evenly split between one, two, or maybe even three different learning styles as described below.

- Visual (spatial): Learner prefers the use of pictures, images, and spatial understanding.
- Aural (auditory-musical): Learner prefers to use sound and music.
- Verbal (linguistic): Learner prefers to use words, both in speech and writing.
- Physical (kinesthetic): Learner prefers using your body, hands, and sense of touch.
- Logical (mathematical): Learner prefers using logic, reasoning, and systems.
- Social (interpersonal): Learner prefers to learn in groups or with other people.
- Solitary (intrapersonal): Learner prefers to work alone and use self-study.

When teaching students programming, the use of visual, verbal and logical learning theory can be included by making the learning into a form of problem solving and letting them learn programming concepts through solving a problem. In the learning programming, students learnt concepts such as iteration by having to use their analytical skills to solve a problem. The concept of differentiation is critical and means that teachers provide instructional strategies that support the various learning styles of their students.

6. Learning Programming through Scratch

Teaching programming was initiated through gaming where learners had to be familiarized through instruction. According to Papert (2006) Logo programming was best suited for learning programming. Scratch programming through the code craft project in the Mauritian Educational System was implemented to develop the programming skills of students of Grade 7. The use of colour coded blocks of code was adopted to overcome the different pitfalls that students had in learning programming, which eventually had helped students from typing mistakes and enjoyed programming.

For example, students were given a small project to check the password. They have drawn a flowchart and then code it through Scratch and Python.

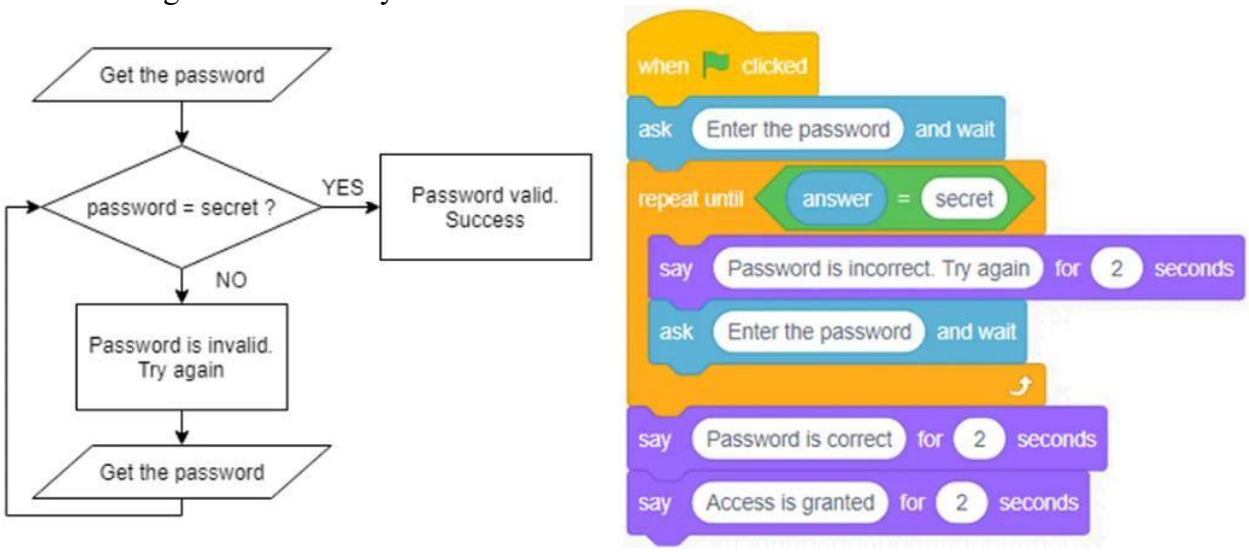

Figure-3.

Flowchart and Scratch block of code to check password.

Regarding the iteration-programming concept, students were introduced to the repeat block as a way to repetitively make scripts shorter. The block is going to check its Boolean condition and then accordingly; it will

act (if it is YES, the blocks held inside it is going to run whereas, if it is NO, the code inside the block is going to be repeated until the password is correct) Figure 3.

According to Armoni, Meerbaum-Salant, and Ben-Ari (2015) higher order thinking is essential for students in problem solving and students with this competency can easily master the programming concepts and eventually motivates others to learn. On the other side, students with lower problemsolving skills have to struggle a lot to learn programming and they are sometimes demotivated and dropped ICT at higher secondary level. The lower problem-solving ability students usually perceived programming language as an obstacle as they have to pay attention to syntax rather than developing algorithm (Grover & Pea, 2013). Thus, scratch programming will be beneficial for the lower secondary students in addressing the problem of syntax. Thus, selecting the correct programming strategy for these students is an essential key factor.

7.1. Learning Programing through Python

Python is a programming language that sustains object-oriented approach which is the current trend at tertiary level. It is a language that is compatible with all operating systems. It makes use of punctuation characters as compared to block of codes in Scratch. The use of whitespace allows learners to write codes and it allow them to structure the program in various ways. It also consists of a number of freely available libraries which might help learners in the coding process.

Python programming has been initiated to grade 9 students as from 2020 and the question that is being raised by educators “Does the initiation of python as a programming language suitable for lower secondary level students?” Most educators pointed that programming syntax is a major pitfall in the teaching and learning programing due to the long statements. Abstraction, clarity of language and irrational structures of python are the other drawbacks. From the learner’s point of view, debugging is a problem when codes do not compile and learner’s have to remember codes. The following example shows the codes in python for checking password.

```
# password checking program

password = input('Enter the password:')
while password != 'secret':
    print('Password is incorrect. Try again')
    password = input('Enter the password:')
print('Password is correct')
print('Access granted')
```

Figure-4.

Python code for checking password.

The student will enter the password. If the password is “secret” then access is granted. If the student has wrongly entered the password, a message “Password is incorrect. Try again” will be displayed to allow the student to re-enter the correct password Figure 4.

7. Research Design

The purpose of this study is to assess the difficulties in teaching and learning program through scratch and any programming language (example, Python). Random sampling was considered for the target students of grade 7 to 9. In Mauritius ICT is a core component that is to be assessed at the end of the NCE program and programming forms part of the 9 contents areas that are to be assessed. Grade 9 students were the target sample for this study. As programming is interrelated problem solving and students had experience scratch during grade 7 and 8, a pre-test was considered in assessing the students. They were exposed to python programming as from grade 9. The Table 3 below demonstrates how the students had to undergo the programming experiments.

Table-3.
Research Design.

	Experiment		
	Topic	**New terms and instructions**	**New concepts**
Pre-test			
Python	Algorithms: sequencing, conditional and iteration	Algorithm, sequencing, conditional and iteration	Introducing algorithm term, basic algorithms: sequencing, conditional and iteration with examples from real life. Introducing to Python programming language
	Variable, input and output	Variable, input, print, int	Basic Python instructions, variable term and integers with examples in Python
	Input processing, output process phases of the computer program	Arithmetic operations (+, -, *,/)	Solving simple problems in Python program using input, processing including basic arithmetic operations and output
	Conditional	If else	Solving simple problems including branching algorithm in Python using if else.
Python test, questionnaire about programming and python			
Scratch	Aquarium simulation program	forward, left, right, repeat	Sprites, concurrency, loops
	Chasing ghosts game	If, variables	Conditionals
	Simple ricochet game	communication by messaging, conditional loops, Coordination and Synchronisation	Loops with conditionals
Scratch test, questionnaire about programming and programming languages			

The students were first exposed to Python programming for four weeks. The lectures included selected programming concepts: variables, input, print, sequencing and conditionals. Student skills in Python programming were tested afterwards. Three weeks later, we introduced students to programming in Scratch. We have selected a game-based approach and students were required to program simple games. They were introduced to basic programming concepts like sequencing, conditional and iteration.

The sample population for this study consisted of fifty students (thirty-four boys and sixteen girls) of grade 9 from two schools (one state and one private) in Zone 2.

8. Assessment Instruments

Data for this study was collected in 3 stages. First, students were evaluated on their problemsolving skills before initiating programming, then after undergoing through scratch programming and finally after introducing the key

concepts in terms of flowchart, python commands such as input and print, variable, and programming construct. At the end, the students had to evaluate their views towards learning programming.

9. Analysis of Discussion

Both qualitative and quantitative methods were chosen for analysis and triangulation approach was adopted to increase the validity of the study (Cohen, Manion, & Morrison, 2013). T-test was used for comparing results among the different groups while signed rank test was adopted for comparing different students results.

10. Discussion and Findings

A problem-solving test was carried out with the 50 students before any programming lectures were held and the maximum marks was 15. Based on the achieved score, students were placed in one of three group; stronger, intermediate and weaker as shown below. Table 4 shows distribution of participants by strength groups. A mean of 11.93 demonstrate that students having problem solving skills were able to recall the information from memory that were needed for writing codes and understanding the syntax of the different structures used in the program as compared to a mean 4.05 for the weaker groups who had some problem solving skills.

Table-4.

Group distribution by students' strength.

Group	Number of students	Marks	Mean	Standard Deviation
Stronger	15	>=11	11.93	1.223
Intermediate	16	Between 7 and 11	9.69	0.704
Weaker	19	< 7	4.05	2.970

The Figure 5 below shows, students' appreciation towards learning programming through Scratch. 37 of them loved scratch as a visual for learning programming while 13 were not of the same views as they thought that learning programming should be code based in terms of variable declaration as python. 40 students believed that learning Scratch has helped them learn programming due to its block of codes and visual layout.

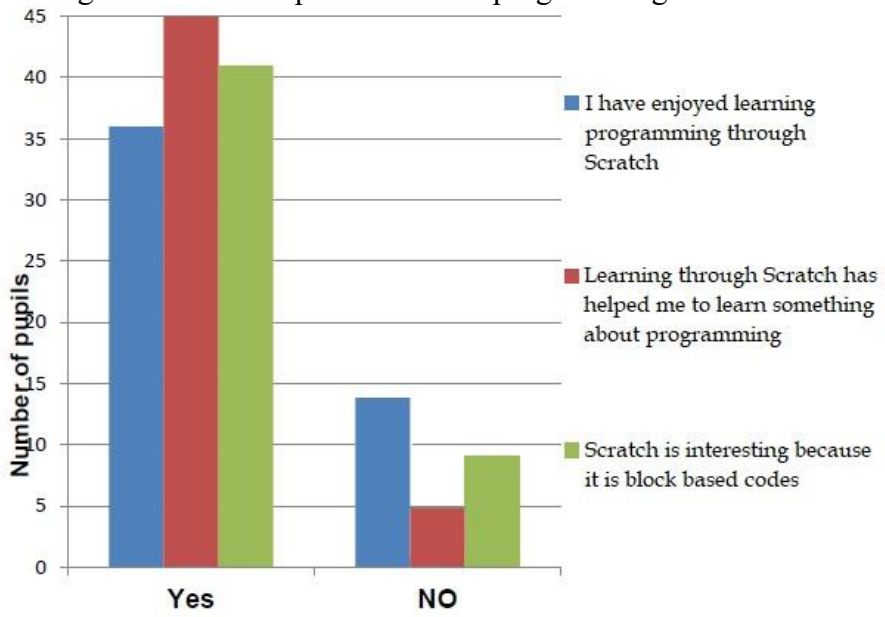

Figure-5.

Trail result on students.

Two assessments were carried out to evaluate the students' performance in programming. The first assessment was administered following the delivery of Python lectures, while the other one was based on Scratch Programming. Since the number of points in each test was different, we decided to use the percentage as a

measure of success. The Kolmogorov-Smirnov (S-test) test showed that there is a normal distribution of data in both Python (p=0.198) and Scratch (p=0.068) tests. The Table 5 below shows descriptive statistics results.

Table-5.

Descriptive results of student's performance.

	Python		**Scratch**	
	Mean	**Standard Deviation**	**Mean**	**Standard Deviation**
Stronger	81.667	20.5116	76.953	19.7003
Intermediate	61.831	22.5395	71.575	19.0275
Weaker	38.910	21.9488	54.537	17.1179

As the Python results did not meet the hypothesis of the normal distribution, a Kruskal-Wallis test was carried out which demonstrated substantial difference among the different groups ($\chi^2(2) = 19.343$, p=0.000). The above table shows that the brighter students performed better as compared to the two other groups. So, ANOVA test was carried to compare the group results of the Scratch test and it was concluded that there exist substantial differences among the groups. This can be established by a single ANOVA ($F(2,47) = 6.945$, $p = 0.003$). As a substantial difference exists among the groups, we had to perform a further analysis and a Mann-Whitney U test was employed.

For this analysis we had to consider only the intermediate and stronger students and the statistic was computed to define whether any difference exists between them and the result were as follows (U =63,5, z=-2,277, p=0,023). It can be concluded the stronger group performance was better than the intermediate one. But no significant difference was noticed in Scratch programming between the two groups (U =99,6, z=-0,838, p=0,419). So, we could conclude that Scratch programming was better to motivate intermediate and lower problem-solving skills students in learning programming as the result correlate with other researches where students programming skills were boost up through learning animations.

Next, we had to use a t-test in order to make a comparison between intermediate and weaker one. The results demonstrated that the brighter group performed better than the weaker ones in Python (t (33) =3,141, p=0,006), and Scratch (t (33) =2.789, p=0.010). it can be concluded that the weaker group had difficulties in understanding programming regardless of the software being used. Eventually, based on the result, we had to accept H1 as problem solving skills are directly interrelated to the success in Python. But this in not the case in Scratch as both the brighter and the intermediate students were successful in accomplishing their tasks and so we could reject the H2. We can conclude that students with higher problem-solving skills can master programming regardless the programming software. Another finding is that visual programming through scratch helped to boost up students' motivational level in learning problem solving. Thus, it is worth for Grade 9 students to be exposed to new programming languages in order to discourage students from quitting programming.

The H3 is expected to give positive outlook towards learning programming after mastering Scratch as compared to Python. After conducting some lessons about programming in Python, students answered a Likert scale question of five items about their attitude towards programming. This question was repeated in the small questionnaire students answered after the Scratch lessons. The questionnaire was composed of four Likert scale questions regarding their attitude towards the programming languages used.

The Table 6 illustrates the questions.

Table-6.

Questions related to survey.

	Questions	
After Python	Q1	How much do you like programming?

After Scratch	Q2	How much do you like programming?
	Q3	How much do you like Python?
	Q4	How much do you like Scratch?
	Q5	Which programming language do you prefer?

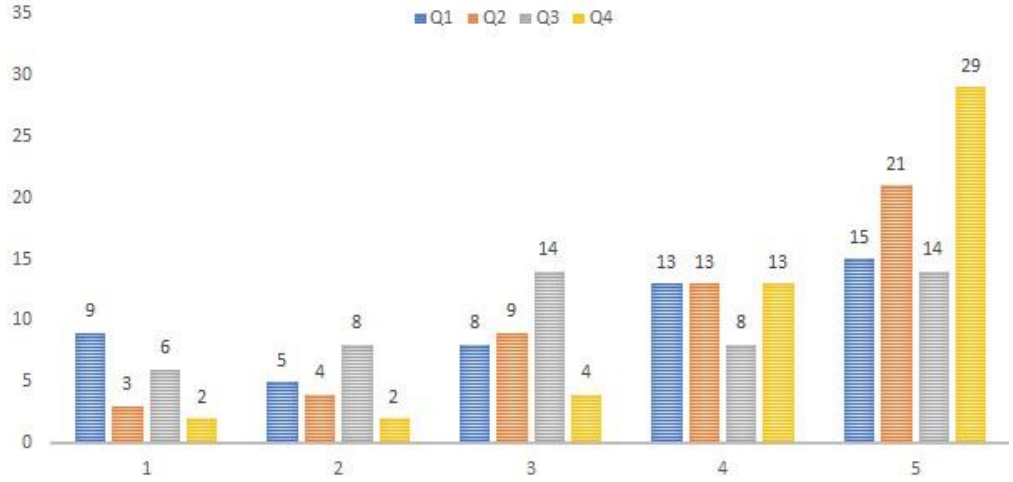

Figure-6.
Frequencies of results.

The above Table 6 demonstrates the frequency distribution (numbers at the top of the bar) of the different questions regarding learners 'attitude towards the programming languages used. It is clear that 29 students prefer Scratch as compared as a programming language and that they showed a positive attitude towards programming. From the findings it is clear that 76% of the respondents prefer to use Scratch as a programming language as compare to 24% who opted for Python. The Figure 7 shows the results of Q5.

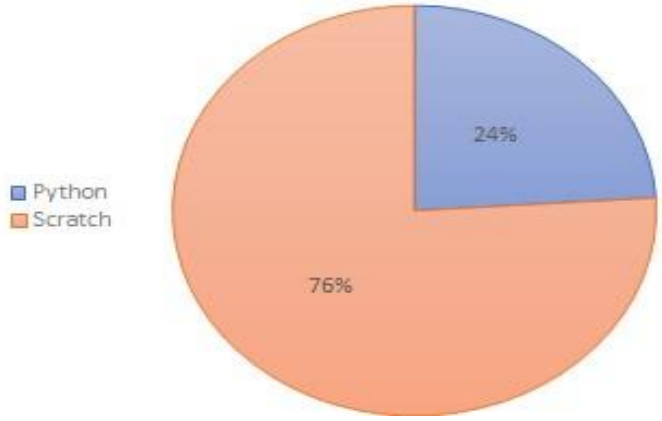

Figure-7.
Result for Q5.

We compared the attitude of students towards programming and the difference between Q1 and Q3 are tabulated below.

Table-7.
Student preference of programming languages.

Dependent variable	Learning method (Independent Variable)	Percentage	Mean	Standard Deviation
Student attitude towards programming	Sractch	76	3.35	1.057
	Python	24	0.57	0.59

The Table 7 shows the mean of the variable "Student attitude towards programming" for the two different programming languages and this statistical data is compared in Table 8. From the table 7, it is clear that learners had a greater affiliation for programming after Scratch (76%) compared to Python (24%). This confirms that Scratch had a positive effect, with a mean of 3.35 on student attitude towards programming as compared to 0.57 to Python. Thus, we can accept H3 and conclude that attitude towards programming is more positive after Scratch than after Python. However, it came as a surprise that even after being introduced to Scratch, a handful of students still preferred Python due to their problem-solving skills. We assumed that these are the students are the intelligent ones and are devoted to problem solving skills.

An independent sample was used here as a method to statistically compare the means of "Which programming language to learner prefer?". By looking at the statistical data in Table 8, it can be noticed there is a statistically significant difference in the attitude towards programming language; as shown in the Sig. (2-tailed) column, the significance result is 0.00, which is less than 0.05 (the result of the level of significance or P value).

Table-8.

Scratch programming preferences.

Dependent Variable	Levene's test for equality of variances		T-test for Equality of Means					95% confidence interval of the differences	
Which programming language do you prefer?	F	Sig	T	df	Sig (2tailed)	Mean difference	Std Error Difference	Lower	Upper
Equal variance assumed	0.17	0.69	2.21	43	0.03	-0.742	0.334	-1.421	-0.063
Equal variance not assumed			2.22	42.80	0.03	-0.742	0.334	-1.422	-0.062

The Table 9 below shows the mean of the variable "I have enjoyed learning programming" and its statistical result. It is clear that more learners preferred Scratch as it has a better (0.982) standard deviation as compared to Python.

Table-9.

Enjoy learning programming.

Dependent variable	Learning method (Independent Variable)	Mean	Standard Deviation
I enjoyed learning programming	Scratch	1.65	0.982
	Python	2.56	0.324

An independent sample was used here as a method to statistically compare the means of enjoyment for both programming languages. By looking at the statistical data in Table 6.10, it can be noticed there is a statistically

significant difference in the enjoyment; as shown in the Sig. (2-tailed) column, the significance result is .000, which is less than 0.05 (the result of the level of significance or P value).

Table-10.

T-test about I have enjoyed learning programming.

Dependent Variable	**Levene's test for equality of variances**		**T-test for Equality of Means**					**95% confidence interval of the differences**	
I have enjoyed learning programming.	F	Sig	T	df	Sig (2tailed)	Mean difference	Std Error Difference	Lower	Upper
Equal variance assumed	10.48	0.003	-4.57	49	0.000	-1.32	0.289	-1.89	-0.73
Equal variance not assumed			-4.49	43.9	0.000	-1.32	0.293	-1.92	-0.72

This statistical technique was used for comparing the mean scores on student's enjoyment in learning programming. For the *t*-test, the mean is -1.32 with a standard deviation error greater than 0.2. With 95% confidence, the difference in mean between lower and upper interval is less than 0 and the results shown that pupils who learnt programming traditionally found programming a boring and difficult process, and this had severely affected their motivation and acceptance of programming in the school.

11. Conclusion

A moderate introduction to programming is required at secondary level for Grade 7 students, where students should focus on problem solving and algorithms through the introduction of visual programming. So scratch programming provides syntax free blocks of codes that motivate students to learn programming. This helps educators to shift their teaching strategies from solving mathematical problems to game programming which boost up students' attitudes towards programming. Scratch programming is an effective tool to transfer visual into real programming language like Python. Students with high problem-solving skills can master programming irrespective of the programming language and those with lower problem-solving skills encounter different problems in mastering the programming concepts.

The lack of research in this area was a disadvantage as results could not be compared. As the study was carried out with 50 students only, the result did not show the reality and only one zone was considered. The result also pointed that there is no difference between intermediate students and brighter students in learning programming through scratch. The majority of the students preferred scratch and learning python first seems to be a problem for initiating programming. We believed that students' motivation to programming would be lower if syntax was to be introduced first.

References

Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From scratch to "real" programming. *ACM Transactions on Computing Education, 14*(4), 1–25. Available at: https://doi.org/10.1145/2677087.

Cohen, L., Manion, L., & Morrison, K. (2013). *Research methods in education. Forehand, M. "Bloom's Taxonomy," Emerging perspectives on learning, teaching, and technology*: Routledge.

Forehand, M. (2010). Bloom's taxonomy. *Emerging Perspectives on Learning, Teaching, and Technology, 41*(4), 47-56.

Govender, I. (2006). *Learning to program, learning to teach programming: Pre- and In-service teachers' experiences of an object-oriented language.* Unpublished Doctoral Dissertation, University of South Africa.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational Researcher, 42*(1), 38–43. Available at: https://doi.org/10.3102/0013189X12463051.

Hromkoviˇc, J. (2006). *Contributing to general education by teaching informatics. In R. Mittermeir (Ed.), Informatics education {the bridge between using and understanding computers* (Vol. 4226). Berlin / Heidelberg: Springer.

Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education, 9*(1), 60-70.

Krathwohl, D. R. (2012). A revision of Bloom's taxonomy: An overview. *Theory into Practice, 41*(4), 212-218. Available at: https://doi.org/10.1207/s15430421tip4104_2.

Mawby, R., Kurland, D. M., Pea, R. D., & Clement, C. (2009). A study of the development of programming ability and thinking skills in high school students.

Ministry of Education Arts and Culture. (1991). Master plan for the year 2000. 125-180.

Papert, S. (2006). *Programming versus application. In R.T. Mittermeir (Ed.), Issep 2006, Incs 4226*. Berlin/ Heidelberg: Springer.

Rayner, S. G. (2015). Cognitive styles and learning styles. In, J.D Wright, (Ed.), International Encyclopedia of Social and Behavioral Sciences (2nd ed., Vol. 4, pp. 110-117). Oxford: Elsevier.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., & Silverman, B. (2009). Scratch: Programming for all. *Communications of the ACM, 52*(11), 60-67.

Rutherfoord, H., & Rutherfoord, J. (2008). *Exploring teaching methods for on-line course delivery-using universal instructional design.*

Paper presented at the Proceedings of the 9th ACM SIGITE Conference on Information Technology Education. ACM, 16th October 2008.

Shulman, L. (2006). Those who understand: Knowledge growth in teaching. *Educational Researcher, 15*, 4-14.

Soloway, E., & Spohrer, J. (2009). Studying the novice programmer (pp. 497). Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Stephenson, C., Gal-Ezer, J., Haberman, B., & Verno, A. (2015). The new educational imperative: Improving high school computer science education (Tech. Rep.). Final Report of the CSTA Curriculum Improvement Task Force - February 2015.

Stickel, M. (2009). *Impact of lecturing with the tablet PC on students of different learning styles*. Paper presented at the Frontiers in Education Conference, FIE'09.39th, IEEE 18th October 2009.

Survey of ICT and Education in Africa: Mauritius Country Report. (2007). *InfoDev ICT and education series*. Washington, DC: World Bank.

Thompson, E. (2008). *Bloom's taxonomy for CS assessment.* Paper presented at the Proceedings of the Tenth Conference on Australian Computing Education, Australian Computer Society, Inc.

Wikipedia. (2020). Computer Programming.Wikipedia, Wikimedia Foundation, 29 July 2020, en.wikipedia.org/wiki/Computer Programming.