Journal of Elementary Education
Vol.25, No. 1 pp. 145-165

A Strategy for the Design of Introductory Computer Programming Course in High School

Adnan Abid[*], Muhammad Shoaib Farooq[**], Uzma Farooq[***],
Kamran Abid[****], Muhammad Shafiq[*****]

Abstract

The ever increasing involvement of electronic and programmable devices in life invites people to learn computer programming as an essential skill. Mathematics and computer programming are two inter-related and inter-dependent subjects. Several different concepts of mathematics are introduced at high school level, and usually the students do not feel comfortable with this subject. In this article, we propose an approach for introducing computer programming at high school level. We have argued that the programming skills should be enhanced with the help of mathematical concepts learned by the students. We present the main idea, and pave the way for its materialization with the help of mapping the constructs of computer programming onto the concepts of mathematics. We also discuss as to how we should customize the programming languages to make it easier to teach and learn computer programming and mathematics with the help of one another. In order to emphasize on our proposed methodology we present a mapping of concepts of mathematics onto the constructs of a widely used introductory computer programming language C++. We further discuss the customization of C++ for teaching and learning computer programming in a more seamless manner by introducing relevant abstractions in the language. Lastly, we also highlight the need of developing new tools, defining syllabus, writing new books, and preparing interesting exercises and exams are the next steps to achieve this goal.

***Keywords**:* Programming Languages, Pedagogical Skills, School Level, Core Mathematics, First Programming Language.

* Associate Professor, Department of Computer Science, University of Management and Technology, Lahore. Email: adnan.abid@umt.edu.pk
** Associate Professor, Department of Computer Science, University of Management and Technology, Lahore.
** *Assistant Professor, Department of Computer Science, University of Management and Technology, Lahore.
****Assistant Professor, College of Engineering and Emerging Technologies, University of the Punjab, Lahore
*****Assistant Professor, Institute of Quality and Technology Management, University of the Punjab, Lahore.

Introduction

Computer programming is an emerging discipline which has found its place in general sciences. As a matter of fact, in many countries computer programming has become a part of curricula at high school level. Whereas, in the UK computer programming has become mandatory for elementary and high school students since 2014 (Sophie, 2015) (Deamicis, 2015) (Curtis 2015). Similarly, the US (Wagstaff, 2015)and Estonia(Olson, 2015) have also introduced computer programming at school level.

Computer science is about learning the mapping of logic behind any computing problems on mathematical, scientific and engineering principles (Farooq et al., 2014). After understanding logic, the next step is to apply that logic on relevant problems. Furthermore, we can also solve logical problems using computer programming, and to this requires the knowledge of computer programming. The art of computer programming is becoming essential for all fields of life, and plays a major role in curriculum(Smith et al., 2010).Steve Jobs emphasizes on the importance of learning computer programming by the following statement: ñeverybody in this country should learn how to program a computer because it teaches you how to thinkò(Verkroost & Eliens, 2013). A proper course on programming is taught at university level under the degree of engineering or computer sciences. Students learn this new dimension of solving problems using computer programming at the university level. However, we believe that by introducing computer programming at high school level may help students developing analytical skills for problem solving in a much more efficient manner (Verkroost & Eliens, 2013) (Farooq et al., 2015). As a matter of fact, the subject of Computer Science has been introduced at school level since the beginning of this century (Julie et al., 2009) (Tobert et al., 2010). Similarly, computers are being used at school level for basic skill building and fun (Farooq et al., 2012), though problem solving is not the major focus of this activity.

In this article, we have capitalized on the global trend of introducing computer programming at high school level and have propose a methodology to introduce it in high schools in general. Our approach mainly relies on teaching computer programming with the help of already learnt mathematical concepts. To this end, we present a methodology in which mathematics is taught, and propose a methodology with which computer programming can be taught with the help of mathematics. Furthermore, we consider the curriculum of mathematics in the province of Punjab in Pakistan to define the mapping of mathematical concepts onto the computer programming constructs. We thus, sum it up as a case study for our proposed approach. We further propose some customizations in a widely used introductory computer programming language C++, so as to make it conformant to our proposed approach by defining certain abstractions which will make it easy for schools students and instructors to learn and teach, respectively. Lastly, we highlight the requirements for the implementation of this approach.

The rest of the article has been structured in the following manner. Section 2 presents the review of the related work. Section 3 presents the logic behind our proposed approach by providing a comparison of problem solving using mathematics, and while using computer programming. Section 4 presents a summary of mathematical concepts that are being taught in the syllabus books of the Punjab text book board of Pakistan. It further presents a mapping of the constructs of a famous computer programming language C++ onto the concepts of mathematics. Furthermore, we also present an idea of enhancing the capabilities of C++ with the help of defining new and relevant abstractions to support the students in implementing their problems in the same section. Further in section 5 we present some discussion with the help of few code listings where we show how the students can write simple codes to learn problem solving using mathematical principles and computer programming. We also present the major requirements for implementing the proposed approach. Lastly, we conclude and discuss the future directions in the Section 6.

Objectives of the Research

The main objectives of this study are to:

a) introduce computer programming at high school in order to increase skilled resources for software industry.

b) mapping the mathematics concepts on programming language constructs in order to reduce the learning overhead for the student, i.e. the student is only learning one new thing with the help of an already learnt material (mathematical concepts).

c) customize any computer programming language to make it more conformant for teaching and learning computer programming at high school level.

d) outline the requirements for implementing the proposed approach.

Related Work

Teaching computer programming at elementary or high school levels is becoming popular and desired. Many different countries are endeavoring to teach computer programming in schools (Sophie, 2015; Wagstaff, 2015; Olson, 2015). In Chang (2006) the authors propose a computer assisted system named Math CAL which are based on systematic problem solving techniques. This system invites students to watch and listen to know what is important for them. In (Denner,2006)and (Clarket al., 2006) the authors have proposed the idea of introducing computer programming at high school level using an already used concept of pair programming. The authors (Ferreira et al., 2009) conduct a study with teachers and students in Brazilian public schools to explore programming at early stage in order to enhance student learning skills. In an effort to help school students to learn programming several Mini languages have been developed (Olimpo, 1988). Here Mini language term is used for languages that use some actor (turtle or robot) which performs different operations in a micro-world using some predefined commands. The main purpose of such approaches is to help the novice programmers learn programming with ease (Ragonis & Ben, 2005). Logo (the turtle) (Pattis, 1981) was the first Mini Language. There are a number of mini-languages which were directly encouraged by Karel and use several of its features: Martino (Cooper et. al., 2003) and Marta (Calabrese, 1989) in Italy, Darel Kay & Tyler (1993) in Australia and Karel-3D Hvorecky (1992) in Slovakia.

In Felleisen & Krishnamurthi (2009) the authors have emphasized on the concept of ñimaginative programmingò considering to be the most crucial element of computing as it closely aligns mathematics with computing. This approach has the added benefit that it makes mathematics relevant for students, improves the grades of students in mathematics courses, and builds on a natural synergy between the two subjects. The teachers find it extremely difficult to create a learning environment that fosters creativity within existing school and curricular structures reported in many research studies (Johnson, 2000; Games 2006; Roschelle et al., 2010; Kurland et al., 1986; Gomes et al., 2006; Felleisenet al., 2004; Feurzeig, Papert & Lawler, 2011; Howe, Shea & Plane, 1979) shares their experiences regarding impact of mathematics and programming for problem solving skills. The Feurzeig (1969) and Feurzeig et al., (2011) created Logo to leverage the power of computing to create an environment for children to explore and discover computational mathematics concepts. In (Linda et al., 2006) author used Python as their first language in introductory programming course in high school. Many research reports prefer Python over other imperative programming languages to teach introductory programming course (Elkner, 2000)

(Julie et al., 2009) (Zelle, 2004). Whereas, C++ has also been widely used as an introductory programming language for decades (Farooqet al., 2014).

Problem solving and Programming with School Level Mathematics Syllabus

Mathematics is generally considered to be a difficult course by most of the school students. The teachers and students struggle in conducting this course successfully. Most of the students rely on scaffolding to increase their grasp on the concepts of mathematics. Thus, increasing the teaching hours for the course of mathematics. On the other hand, if we observe the stream of computer programming courses we can clearly see that it involves two to three different courses in computer programming. Among these courses, the introductory course in computer programming revolves around the basic concepts of mathematics which are generally learned during the high school. Hence, a first course in computer programming involves the topics covered in mathematics at class six onwards. For instance, some of the fundamental topics in core programming are data types which can be directly mapped on the Sets, there is an excessive use of variables and operators in implementing computer programs, where both of them are a core part of mathematics learned at school level. Most of the lectures are devoted to these topics in introductory course on programming. Similarly, many examples and exercise questions written in computer programming text books are based on these topics (James, 2015; Tobert et al., 2010). Therefore, we can establish a link between the basic concepts of mathematics and introductory constructs of computer programming. This link urges us to envisage, if we can support the learning of mathematics in schools by involving computer programming, and vice versa. The methodology of teaching computer programming and mathematics can result into a better learning of both subjects. Certainly, involvement of computer programming at schools will help increasing the interest of students to both courses. It will help to reduce the basic syllabus at university level computer programming courses, such as introduction to data types, operator precedence and associativity, Boolean logic, assignment and decision control instructions. At university level lot of lecture time and lab work required for these topics that have been already covered in elementary school level mathematics.

Method and Procedure

In order to materialize our approach we have used a case study which uses the curriculum addressed in text book on mathematics designed by Punjab Text Book Board. These book organized topics into chapters, each chapter introduces the concept of a topic, followed by solved examples, and finally detailed exercises. This approach has been presented in Figure 1 (a).

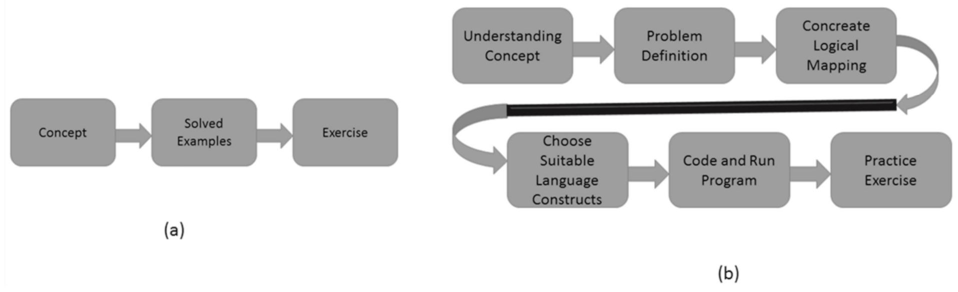

Figure 1:(a)Method Used in Mathematics Books
(b) Proposed Methodology for Introducing Computer Programming with Mathematics

Thus, the teachers follow the same approach as given in text books by, firstly introducing concept of topic, secondly student is trained to solve problem by using concept, and then finally, student practices these concepts by solving exercises of the topic. This approach creates attrition of students and dropout/failure rate increases in mathematics and student has not enough opportunity to practically apply these concepts on real world problems but focus on just solving exercises.

In this research we argue that introducing computer programming at school level should be linked with the concepts of Mathematics. This would help introducing computer programming with minimal overhead. Similarly, the student can learn and practice the concepts of mathematics with this approach. This proposed approach will change the earlier methodology as show in Figure 1(b). The proposed approach for learning mathematics with the help of computer programming firstly introduces the concept of a topic, secondly it focuses on problem definition, thirdly develops concrete logical mapping of the problem, fifthly chooses suitable language constructs, sixthly finalizes and runs the program; and lastly, the student practices given exercises in books.

Certainly, by using the proposed approach students will practice mathematics concepts in computer lab that are already introduced by his teacher in class. In this way, the students will only focus on learning computer programming, as the mathematical question which they are going to solve has already been learnt in the class room. Thus, the only new element for learning is that of the constructs and logic for writing computer program. Similarly, in order to minimize the overhead of writing code, we intend to offer student write their program in computer by simply drag and drop C++ instructions. This will help students write a computer program without any syntax error. It will enhance student logical thinking and improve student performance in course of mathematics.

Course Curriculum of Mathematics at School Level (Grade: 6 to 10)

In this section we define the syllabus of different topics covered at different levels of mathematics. We have focused on the matriculation system practiced in the province of Punjab in Pakistan. Major topics covered in first programming course at university level are variables, constants, data types, assignment, operators (precedence and associativity), control statements, arrays and functions. The root of these topics are traversed back to school level mathematics syllabus. For example concept of variables and constants are being taught in the topic of algebra, which is one of the core concepts of mathematics. The whole concept of integer, real, boolean has been introduced at six grade students and, we can map these concepts on C++ integer, double and boolean data types. In this research we map all core mathematics topics offered at 6th grade to 10th. Major focus is on simplifying the tasks of computer programming and learning mathematics with the help of one another. Table 1 defines the summary of the concepts of mathematics learned in different grades in high school.

Table 1

Topics of mathematics covered in grades 6-10

Topic	Grade	Areas Covered
Fractions	6th	• Types of Fractions (Proper, Improper, Compound) • Distributive property of addition and subtraction over common and decimal Fractions • Use of brackets in Fractions • Solution of Real life problems involving Fractions.
Ratio/Proportion	6th	• Concept of Ratio among two or more quantities • Relationship between ratio and fraction • Dividing quantity in a given ratio • Daily life problems related to inheritance and partnership • Concept of direct and inverse proportion and their use in

		daily life.
Integers	6th	• Concept of number line and directed numbers • Explanation of integers as directed numbers on the number line • Operations of addition and subtraction on directed numbers • Concept of smaller and bugger integers • Use of the symbols < and >
Average	6th	• Calculating average of given quantities • Concept of weighted average • Real life problems based on weighted average
Percentage	6th	• Concept of percentage • Relation among percentage, common fraction and decimal fraction • Finding percentage of given quantity • Finding quantity when percentage is given • Conversion of one quantity as the percentage of another quantity • Use of percentage for the solution of problems related to Zakat, Tax, Custom Duty, Commission, Discount, Profit & Loss
	7th	• Concept of Purchase and Sale Price, Profit and Loss • Finding profit or loss percent when purchase price and sale price is given • Finding sale price when purchase price and profit or loss percent is given • Finding the purchase price when sale price and profit or loss percent is given
Algebra	6th	• Basic concepts of Algebra • Concepts of variables, constants, exponents, power and co-efficient • Use of arithmetic operations (÷, x, - , +) in algebra • Concept of like and unlike terms • Concepts of monomial, binomial and trinomial and operations of addition and subtraction on them • Laws of exponents • Concept of Algebraic Sentences • Types of algebraic sentences (true, false and open sentences) • Concept of an equation

	7th-8th	• Solution of simple equation • Concepts of Polynomials • Arrangement of polynomials in descending and ascending order • Arithmetic Operations on polynomials • Concept of Symbolic expression • Linear Equations and their solutions • Solution of daily life problems using linear equations
	9th-10th	• Solution of simultaneous linear equations by substitution method, comparison method, equating the co-efficient method • Graphs of Linear Equations • Quadratic Equations and their solution • Quadratic Formula • Inline equations and their solution • Remainder theorem and its applications
Geometry and Trigonometry	9th-10th	• Lines • Circles • Arcs • Triangles

Mapping of mathematics concepts onto the programming constructs

In this section we define the mapping of mathematics constructs on general programming constructs. Later on, we present an idea of developing new abstractions in any language to support the school students in learning and writing computer programs easily. We choose C++ for this purpose because this language has been taught as an introductory first programming languages for many years (Farooq et al., 2014).

Although C++ is one of the best option to teach programming to the students but we cannot teach this directly at school level. There are different paradigms of teaching computer programming namely, object first approach, and imperative first approach, and C++ supports both approaches. Generally, imperative first approach is recommended for teaching an introductory course in computer programming, and we also recommend this approach for our purpose. To this end, we have also defined a subset of C++ constructs that can help learning mathematics at school level, and will also help in learning basic concepts in computer programming, thus enhancing the problem solving capabilities and skills of the students in many different ways. We choose the following C++ constructs:

1- Data Types (integer, double, character)
2- Arrays (integer and double) up to second dimension
3- Arithmetic operators (+, /, *)
4- Relational Operators (<, <=, >, >=, !=, ==)
5- Function call (by value)
6- Declaration statement
7- Control Statement (If statement, While loop)
8- I/O (cin and cout)

Here we discuss about how to teach a programming language to school students by dividing its constructs into grades and then into course related topics. Table 2 defines the mapping of different topics of mathematics onto the constructs of C++ programming language. These topics are given in mathematics syllabus produced by Punjab text book board for high school (Rafiq & khan, 2014; Rafiq & Ansari, 2014; Amin et al., 2014; Asghar& Habib, 2014; Dar, Irfan-ul-Haq & khan, 2014).

Table 2
Mapping of Mathematics Concepts onto Programming Language Constructs

Grade	Topic	C++ Constructs
6th	Integers	Data Type: int Relational Operators
	Algebra	Variables; Constants; Arithmetic Operators and Precedence; Boolean expressions; Conditional Structures
	Average	Arithmetic Operator
	Percentage	Arithmetic Operators
	Fraction	Function Calling
	Ratio & Proportion	Function Calling
	Geometry	Arithmetic Operators; Formulae Translation
7th& 8th	Algebra	Variables; Constants; Arithmetic Operators
	Percentage	Arithmetic Operators
	Rational Numbers	Data Type: double
	Ratio & Proportion	Function Calling
	Geometry	Function Calling; Formulae Translation
	Sets	Arrays; Loops
	Square Root	Function Calling
9th	Real Numbers	Data Type: double
	Logarithm	Function Calling
	Matrices & Determinants	2D Arrays; Nested Loops
	Sets	Arrays; Loops
	Algebra	Formulae Translation
10th	Trigonometry	Function Calling
	Algebra	Complex Expressions

Definition of Higher Level Abstractions

At this stage we have two options to teach programming constructs of C++. The first one is to teach pure C++ language constructs without any higher level abstraction and the second option is to design some higher level abstractions for school students so that they will easily implement their curriculum problems. We are in favor of second option because our purpose is to teach them basic programming constructs and not advanced concepts like Object Oriented Programming. For this purpose, we have proposed the design of Fraction as higher level abstractions for school students which are given below.

In addition to these data types as higher level abstractions, we should also introduce some new user defined abstract data types (ADT) for the ease of students like Fraction, Percentage, Set, Matrix, etc. As an example we present an abstraction for teaching, learning, and programming the concept of fractions.

Fraction

Fraction is purposed as user defined data type. User can declare the object of type Fraction and can assign a fraction value directly to it. The interface for Fraction class is shown in Figure 2.

Fraction
+ numerator: int + denominator: int
+ Fraction(char *) + simplify( ): void + checkType( ): void + convertToCommonFraction( ): void + convertToDecimalFraction( ): void + convertToPercentage( ): void + operator +(Fraction): Fraction + operator -(Fraction): Fraction + operator *(Fraction): Fraction + operator /(Fraction): Fraction + operator ==(Fraction): bool + operator !=(Fraction): bool + operator <=(Fraction): bool + operator >=(Fraction): bool + operator >(Fraction): bool + operator <(Fraction): bool + operator <<(ostream&, Fraction &): ostream& + operator >>(istream&, Fraction &): istream&

Figure 2: UML diagram that defines ADT for Fraction

Discussion

In this section we present some example scenarios which will help us understanding the strategy and mapping of concepts of mathematics onto the constructs of computer programming language. These examples lay down the foundation of the materialization of our concept of enforcing the learning of mathematics and computer programming with one another. We have chosen C++ language for providing the code snippets used in our examples. We have presented the examples based on the mathematics concepts including integers, fractions, algebra etc. Each concept of mathematics is implemented with the help of C++ code.

Examples

a. Integers

As discussed earlier, the topic of integers covers the concept of integers and the relational operators in detail. Hence we can easily map these concepts on C++ data type *int* and relational operators in conditional statement. It will make our job much easier to teach these basic concepts of programming language.

Consider the following example from the course curriculum. The question statement is *Add (+7) and (+2)*. We can easily solve this using C++. The Code Listing 1 presents a simple snippet of C++ code that helps implementing a simple mathematical expression.

Code Listing 1: Expression evaluation

Add (+7) and (+2).

```
#include <iostream.h>
void main( )
{
int sum;
  sum=(+7)+(+2);
cout<<sum;
}
```

```
Output:
9
```

Now, it is easy to grab the concept or purpose of *int* data type by the student of even 6th grade because it has the same concept as that of the course curriculum topic. Similarly consider the example shown in Code Listing 2 that involves relational operators.

Code Listing 2: Relational Operators

Statement -3>3 is true or false?

```
#include <iostream.h>
void main( )
{ if(-3>3)
cout<<ñtrueò;
  else
cout<<ñfalseò;
}
```

```
Output:
False
```

Simple Algebra

Through simple algebra topic, we can easily cover the concept of variables, constants, arithmetic operators and their precedence, Boolean expressions and conditional statements as shown in Code Listing 3.

Code Listing 3: Simple algebra with operator precedence

Take three integer values and solve an expression based on these integers.

```
#include <iostream.h>
void main( )
{
int no1=5;
int no2=9;
int no3=2;
intans;
ans=no1+no2*no3;
cout<<ans;
}
```

```
Output:
23
```

Similarly to re-emphasize the concept of Boolean expressions, consider the example presented in Code Listing 4.

Code Listing 4: Boolean Expression using Relational Operator

Take two numbers from user and print the maximum of them.

```
#include <iostream.h>
void main( )
{int no1;
int no2;
cout<< ñNo1: ò;
cin>>no1;
cout<< ñNo2: ò;
cin>>no2;
  if(no1>no2)
cout<< ñMax: ò<<no1;
 else
cout<< ñMax: ò<<no2;
}
```

```
Output:
No1: 13
No2: 5
Max: 13
```

Fraction

How to call a function is one of the important concepts of C++ language. With the help of this topic, we can easily teach this concept to the students. Certainly, we shall have to define an abstraction for the topic of fraction in the language before writing the code presented in Code Listing 5.

Code Listing 5: Fraction with the help of ADT

Find that the given fraction is proper or improper?

```
#include <iostream.h>
void main( )
{
  Fraction f= ñ6/5ò;
f.checkType( );
 }
```

```
Output:
Improper Fraction
```

Here, we have used a higher level abstraction i.e. Fraction which makes the code easier to understand at school level. It hides the complex syntax of C++ and logic behind the function to make it a simpler program. Similarly, another example for the topic of fraction is presented in Code Listing 6.

Code Listing 6: Fractions with the help of ADT

Find that the given pair of fractions are equivalent or not?

```
#include <iostream.h>
void main( )
{
  Fraction f1=ò3/2ò;
  Fraction f2=ò12/8ò;
  if(x==y)
cout<<òEquivalentò;
  else
cout<<òNon-Equivalentò;
}
```

```
Output:Equivalent
```

In the above example, we have used == operator for variables of Fraction class. Here, the == operator is already overloaded in the Fraction class which makes the code a lot easier to understand.

Geometry

Through geometry topic, we can easily teach the concept of operator precedence and formula translation in C++ language. Consider the example shown in Code Listing 7 that calculates the area of a right angle triangle.

Code Listing 7: Trigonometry

Calculate the area of right angle triangle.

```
#include <iostream.h>
void main( )
{
int base=3;
int altitude=4;
int area;
  area=(base * altitude)/2;
cout<<ñArea of Triangle: ò <<area;
}
```

```
Output:
Area of Traingle: 6
```

The above examples help us establishing our idea that we can introduce computer programming at school level to help learning mathematics along with this new skill of writing simple software. Initially, it will enhance the student confidence and problem solving skills at their early stage. Further it will help improving the curriculum at university level by modifying the course outlines of the courses related to computer programming. Thus, the students can be taught more detailed and advanced concepts at the university level. Therefore, introducing programming at school level especially synchronized with core mathematics helps to enhance student problem solving skills. It will produce intellectual students for universities that already have programming experience at school level. These students will become good programmers at university level and it will help to increase ratio of programmers, which is the most important requirement of developing countries like Pakistan.

Requirements

In order to implement the proposed approach we need the following resources:

a. Student should have basic skills in computer such as keyboard, mouse and basic file handling and operating system.
b. Teachers should know mathematics plus basic computer skills such as keyboard handling, mouse, basic file system and operating system.
c. Three day workshop (24 Hours) on computer skills and implementing computer programs required for teachers/Lab Staff who have no or little knowledge on computer science.
d. Computer laboratories with supporting software installed that help the student to write code in easy way.
e. Pedagogical programming environments or some customized environments should be used to implement programs in computer. These environments should support drag and drop instructions, basic debugging, simple output console and pretty printer. The best options are net beans with some pedagogical plugin support, CppCheck, cpplint, lint, BlocC and PC-Lint.
f. Text books that includes computer programming concepts and practice exercises.
g. Proper student manual for the usage of tools all basic guidelines are published in the form of a book.

h. Teacher manual for the book and programming tool which helps understanding the concepts, methodology of teaching, and solutions of selective exercises.
i. At least, one computer science teacher should be there for a school with one section.

Conclusion and Future Directions

In this article, we have envisaged that computer programming should be included in the high school learning process. We strongly believe that mapping the concepts of mathematics learned at the high school level to the constructs of computer programming with the help of useful abstractions will help increasing the studentsô learning, understanding, and problem solving skills in many ways. Furthermore, in order to materialize our vision we present the mapping of mathematical constructs onto C++ language constructs. Lastly, we provide some examples to help visualizing our concept and strengthen the argument of learning mathematics with the help of programming and vice versa.

We also foresee that by imparting this necessary skill of computer programming to our students at high school level we shall be equipping our man power with an essential tool for future. This will help inviting people towards the software industry which can play a vital and decisive role in the progress of a developing country with scarcity of resources, like Pakistan. This may certainly help increasing the human capita for Information Technology sector, particularly in the domain of software development. Thus, we can expect a definite increase in the revenue and exports in the domain of software sector.

This vision involves many different activities for the future including, experiments, language customization, developing tools and integrated development environments for the students and teachers, writing good quality text books, and designing thought provoking tests and exams. All of the aforementioned aspects are very important and need effort and demand attention of the government.

References

Amin M., Baig, I. M., & Saeed, M.A. (2014). *Mathematics, General Science, Grade 8*, Punjab Text Book Board Lahore.

Asghar, C. A., & Habib, M. (2014). *Mathematics, General Science, Grade 10,* Punjab Text Book Board Lahore.

Calabrese, E. (1989). Marta - the "Intelligent Turtle". *Proceedings of Second European Logo Conference, EUROLOGO'89*. pp. 111-127.

Clark B., Courte J., & Howard, E. V. (2006). Programming in pairs with Alice to improve confidence, enjoyment, and achievement, *Journal of Educational Computing Research, Vol.34*(2), pp.213-228.

Cooper S., Dann, W. & Pausch, R. (2003). Using animated 3d graphics to prepare novices for CS1. *Computer Science Education, 13*(1):3ï 30.

Chang, K., Sung,Y., & Lin, S. (2006). Computer-assisted learning for mathematical problem solving. *Computers & Education, vol. 46*(2), 40-151.

Denner, J., & Werner, L. (2007). Computer programming in middle school: How pairs respond to challenges. *Journal of Educational Computing Research, 37*(2), 131-150.

Elkner, J. (2000). *Using Python in a high school computer science program*, Proceedings of the 8th International Python Conference.

Farooq, M. S., Khan, S. A., Abid, K., Ahmad, F., Naeem, M. A., Shafiq, M., & Abid A. (2015). Taxonomy and design considerations for comments in programming languages: a quality perspective. *Journal of Quality and Technology Management, 10*(2).

Farooq, M. S., Abid, A., Khan, S. A., Naeem, M. A., Farooq, A., & Abid, K. (2012). A Qualitative Framework for Introducing Programming Language at High School, *Journal of Quality and Technology Management, 8*(2).

Farooq, M.S., Khan, S. A.,Ahmad, F., Islam, S. & Abid, A. (2014). An Evaluation Framework and Comparative Analysis of the Widely Used First Programming Languages. *PloS one 9*(2): e88941.

Felleisen, M., &Krishnamurthi, S. (2009). Viewpoint Why computer science doesn't matter. *Communications of the ACM, 52*(7), 37-40.

Felleisen, M. et al. (2004). The Teach Scheme! Project: Computing and programming for every student. *Computer Science Education 14*(1): 55-77.

Feurzeig, W. (1969). *Programming-Languages as a Conceptual Framework for Teaching Mathematics.* Final Report on the First Fifteen Months of the LOGO Project.

Feurzeig, W., Papert, S. A., & Lawler, B. (2011). Programming-languages as a conceptual framework for teaching mathematics. *Interactive Learning Environments, 19*(5), 487-501.

Ferreira, A. M., Pereira, E. N., Silva, J. C. A., Carelli, I. M., Silva, M. A. R., & Dias, A. L. (2009). A Culturally Contextualized Web based Game Environment to Support Meaningful Learning. *In CSEDU* (2), pp. 205-210.

Gomes, A., Carmo, L., Bigotte, E., & Mendes, A. (2006). *Mathematics and programming problem solving.* In 3rd E-Learning Conferenceï Computer Science Education.

Howe, J. A., O'Shea, T., & Plane, F. (1979). *Teaching mathematics through LOGO programming: an evaluation study.* Department of Artificial Intelligence, University of Edinburgh

Hvorecky, J. (1992). *Karel the Robot for PC*. Proceedings of East-West Conference on Emerging Computer Technologies in Education, 6(9), pp. 157-160.

James, J. (2015). *How to introduce high school students to programming.* Retrieved from http://www.techrepublic.com/blog/programming- and-development/ how-to-introduce-high-school-students-to-programming/1511

Johnson, D. C. (2000). Algorithmics and programming in the school mathematics curriculum: support is waning-is there still a case to be made? *Education and Information Technologies, 5*(3), 201-214.

Julie, A., Rursch & Jacobson D. (2009). *IT-Adventures: Turning High School Students "ON" to Information Technology 39th ASEE/IEEE Frontiers in Education Conference.*

Dar, K. H., Irfan-ul-Haq & Khan, A.R. (2014). *Mathematics, General Science, Grade 9,* Punjab Text Book Board Lahore.

Kay, J., & Tyler, P. (1993). A micro world for developing learning design strategies. *Computer Science Education 3* (1), 111-122.

Kurland, D. M., Pea, R. D., Clement, C., & Mawby, R. (1986). A study of the development of programming ability and thinking skills in high school students. *Journal of Educational Computing Research, 2*(4), 429-458.

Linda G., et al. (2006). *Why complicate things?: introducing programming in high school using Python. Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*. Australian Computer Society, Inc.

Olimpo, G. (1988). The Robot Brothers: An environment for learning parallel programming oriented to computer education. *Computers and Education 12* (1), 113-118.

Olson, P. (2015). *Why Estonia has started teaching its first-graders to code*? Retrieved from http://www.forbes.com/sites/parmyolson/2012/09/06/whyestonia-has-started-teaching-its-first-graders-to-code.

Pattis, R. E. (1981). *Karel - the robot, a gentle introduction to the art of programming.* London: Wiley

Rafiq, M. T., & Khan, T. R. (2014). *Mathematics, General Science, Grade 6,* Punjab Text Book Board Lahore.

Rafiq, M. T., & Ansari, Z. (2014). *Mathematics, General Science, Grade 7,* Punjab Text Book Board Lahore.

Ragonis, N., & Ben-Ari, M. (2005). A long-term investigation of the comprehension of OOP concepts by novices. *Computer Science Education, 15*(3), 203-221.

Roschelle, J., Shechtman, N., Tatar, D., Hegedus, S., Hopkins, B., Empson, S., & Gallagher, L. P. (2010). Integration of Technology, Curriculum, and Professional Development for Advancing Middle School Mathematics Three Large-Scale Studies. *American Educational Research Journal, 47*(4), 833-878.e Education.

Smith D., Lameras, P., & Moumoutzis, N.(2010). *Using educational programming language to enhance teaching in computer science.* Edge conference.

Sophie C. (2015). *Computing curriculum: Digital skills versus computer science.* Retrieved from http://www.telegraph.co.uk/technology/news/10584617/ Computing- curriculum-Digital-skills-versus-computer-science.html.

Sophie C. (2015). *Teaching our children to code*. Retrieved from http://www. telegraph.co.uk/technology/news/10410036/Teaching-our-children-to-code-a-quiet-revolution.html,

Tobert, S.,Vishkin,U., Tzur, R., & Ellison, D. J. (2010). Is teaching parallel algorithmic thinking to high school students possible?: One teacher's experience. *Technical Symposium on Computer Science Education-SIGCSE,* pp. 290-294.

Verkroost, Y., & Eliens, A. (2013). *Seriousify and prettify our educational system!*.

Wagstaff, K.. (2015). *Can we fix computer science education in America*? Retrieved from http://techland.time.com/2012/07/16/can-we-fix-computer-science-education-in-america.

Zelle, J. M. (2004). *Python programming: An introduction to computer science.* Franklin, Beedle & Associates, Inc.