Facile Programming

Hanan Elazhary
Computers and Systems Department, Electronics Research Institute, Egypt

Abstract: *High-level languages and very high-level languages have been developed to simplify programming. But, programming is still hard for many people especially those from disciplines that are not computer-related. Facile programming implies the modification of programming languages to be easily learnt, remembered, and used by programmers from different disciplines. This is achieved by studying and trying to tackle the practical difficulties that face such programmers. The paper addresses the difficulty of learning, remembering, using, and understanding compilation error messages of common English-like programming languages by programmers who are not fluent in English. To tackle this problem, we developed Arabic versions of LISP and SQL in an attempt to figure out whether developing versions, of common programming languages, that are like natural languages of programmers would improve their programming capability. Syntax errors in the Arabic versions can be detected and the corresponding error messages are produced in Arabic. To encourage the use of these Arabic versions, we also developed translators that can translate programs between the corresponding Arabic and English versions of these programming languages for portability. This paper explains the Arabic version of SQL, but reports results from our experience with the Arabic versions of both LISP and SQL.*

Keywords: *Programming languages, Arabic programming, SQL, Arabic SQL.*

Received November 22, 2009; accepted March 9, 2010

1. Introduction

If two parties would like to communicate with each other, they have to speak the same language or they should use a translator. Thus, when computer systems were invented, programmers had to learn the machine language of a given computer system in order to be able to communicate with it. Unfortunately, this task was very hard since machine language includes two letters only (zero and one). Besides, machine language is a low-level language that requires knowledge of the underlying hardware and is machine dependent. In other words, it differs from one family of computer systems to another. This means that programmers had to learn more than one machine language. The alternate solution was to use translators, but this was not a practical solution due to ambiguity of natural languages [10]. To solve this problem, assembly language [9] was invented. In assembly language, an English-like statement corresponds to each machine language statement and an assembler translates assembly language programs to machine language programs using lookup tables. Unfortunately, assembly language is also a low-level language and is machine dependent like machine language. Striving to make programming much easier, high-level languages and very high-level languages [5, 6, 8] have been invented. These languages are more user friendly since they are not machine dependent and do not normally require knowledge of the underlying hardware. Unfortunately, in spite of this, programming is still a difficult task for many programmers especially those from disciplines that are not computer-related. Facile programming implies the modification of programming languages so as to be easily learnt, remembered, and used by programmers from different disciplines. The goal is to provide programmers from diverse backgrounds with programming capability. This is achieved by studying the practical difficulties that face such programmers and trying to tackle them.

From our experience with teaching programming to students from different disciplines (including computer-related disciplines) at several universities, we noticed that programmers who are not fluent in English had difficulties in learning, remembering, and using common programming languages. By examining programs written by such programmers, we figured out that it is hard for them to remember the keywords and syntax of these common programming languages. We also noticed that it is even harder for them to understand the English compilation error messages. The reason is that these English-like programming languages have been invented by English-speaking developers and have been intended to be used by English-speaking programmers [20]. Consequently, we conducted a research study in an attempt to answer the following question: Can developing versions of common programming languages, which are like natural languages of such programmers, improve their programming capability? We developed an Arabic version of LISP [15], namely Arabic LISP [7] as an example of a general-purpose programming language and an Arabic version of SQL, namely Arabic SQL as

an example of a special-purpose programming language. The reason for selecting the Arabic language is that it is the official language of hundreds of millions of people in the Middle East and North Africa. The reason for selecting Lisp is that it is one of the easiest general-purpose programming languages [15]. On the other hand, the reason for selecting SQL is that it is used by programmers from diverse disciplines to access database systems [3, 17]. Syntax errors in the Arabic versions can be detected and the corresponding error messages are produced in Arabic. To encourage the use of these Arabic versions, we also developed translators that can translate programs between the corresponding Arabic and English versions of these programming languages for portability. These Arabic versions were tested by Arabic-speaking programmers who are not fluent in English. Samples of these programmers were selected from different disciplines and different ages. This paper is concerned with explaining Arabic SQL, but it reports results from our experience with both Arabic Lisp and Arabic SQL. The paper is organized as follows: section 2 provides the background of the paper and discusses related research studies explaining why they are insufficient to tackle the problem studied in this paper. Section 3 briefly explains SQL for readers who are not familiar with it. Section 4 explains in details Arabic SQL and its operation. This section is well-explained such that even non-Arabic speakers can understand it. Section 5 provides results from our experience with both Arabic LISP and Arabic SQL. Finally, section 6 presents the conclusions and directions for future research.

2. Background

There have been some attempts in the literature to develop programming languages that are like natural languages other than English. For example, Lukaszewicz [13] developed a Polish programming language called SAKO. Yaohan [19] developed a Chinese programming language. Al-Sadoun *et al*. [1] developed an MS-DOS based Arabic programming language. Amin [4] developed an Arabic object-oriented programming language called Al-Risalh. Tetsuji *et al*. [16] developed a Japanese-based programming language called Mahoroba. Similar attempts have been made for programming environments. For example, Rafea [14] developed an Arabic interactive programming environment. Another Arabic programming environment has been developed by Al-Salman [2]. But, to the best of our knowledge, such attempts have not been extended to cover common high-level and very high-level programming languages. For example an Arabic programming language similar to C has been developed [21], but this language is not exactly like C. Thus, no direct translation is possible between this programming language and C and hence portability is not straightforward. For these reasons, such programming languages have not been widely accepted. Some other programming languages are partially English or English-like. For example, Microsoft Access [18] allows using the Arabic language in specifying names and data values in database systems. But, unfortunately, SQL (which is used to access database systems) keywords are English-like and error messages are English. For these reasons, these programming languages were not used in testing the feasibility of developing versions, of common programming languages, which are like the natural languages of programmers.

3. SQL

Structured Query Language (SQL) [5] is a language designed for the management and retrieval of data in Relational Database Management Systems (RDBMS). Unfortunately, there are many different versions of SQL, but to be in compliance with the ANSI standard, they must support the same major keywords (such as SELECT, UPDATE, DELETE, INSERT, and WHERE) in a similar manner. The most basic statement in SQL is the SELECT statement. SELECT retrieves information from a specified table or multiple related tables in a database with several optional keywords and clauses including:

- *FROM*: Indicates the source table or tables from which the information is to be retrieved.
- *WHERE*: Includes a comparison, which is used to restrict the number of rows returned by the query.
- *GROUP BY*: Groups rows with related values resulting in a smaller set of rows.
- *HAVING*: Includes a comparison used to eliminate rows after the GROUP BY keyword is applied.
- *ORDER BY*: Identifies columns used to sort the resulting rows.
- *AVG*: Computes the average of a given argument.
- *AS*: Specifies a name for a returned column.

The specification of the desired information is called a query. For example, consider a database system in which there is a table called الموظفون (“Employees” in English). This table includes information about employees in different cities as shown in Figure 1. The English translation of this table is shown in Figure 2. الكود means “Id”, الاسم means “Name”, المدينة means “City”, and العمر means “Age”. Three examples of SQL queries on this table and the results of these SQL queries are shown in Table 1:

الموظفون : Table

	الكود	الاسم	المدينة	العمر
+	1	احمد	القاهرة	30
+	2	عمر	الجيزة	29
+	3	منى	الجيزة	20
+	4	سحر	طنطا	31
+	5	نادر	الاسكندرية	40
+	6	نهى	القاهرة	20
+	7	سارة	القاهرة	18

Figure 1. Table الموظفون.

- In example 1, the SQL query requests all information about employees in القاهرة ("Cairo" in English). In this example, الرقم means "Number".

Emplyees : Table

	Id	Name	City	Age
	1	Ahmed	Cairo	30
	2	Omar	Giza	29
	3	Mona	Giza	20
	4	Sahar	Tanta	31
	5	Nader	Alex	40
	6	Noha	Cairo	20
	7	Sarah	Cairo	0

Figure 2. Translation of table الموظفون .

- In example 2, the SQL query requests متوسط العمر ("average age" in English) for employees in القاهرة ("Cairo" in English).
- Finally, in example 3, the SQL query requests متوسط العمر ("average age" in English) for employees in each مدينة ("city" in English) ordered by متوسط العمر ("average age" in English).

Table 1. Examples of SQL queries on الموظفون table and their results.

Example Number	SQL query	Result
1	SELECT * AS [الرقم] FROM الموظفون WHERE المدينة = "القاهرة"	Query1 : Select Query الرقم | الاسم | المدينة | العمر 1 | احمد | القاهرة | 30 6 | نهى | القاهرة | 20 7 | سارة | القاهرة | 18
2	SELECT AVG (العمر) AS [متوسط العمر], المدينة FROM الموظفون GROUP BY المدينة HAVING المدينة = "القاهرة"	Query2 : Select Query متوسط العمر | المدينة 22.6666666666667 | القاهرة
3	SELECT AVG(العمر) AS متوسط [العمر], المدينة FROM الموظفون GROUP BY المدينة ORDER BY AVG(العمر)	Query3 : Select Query متوسط العمر | المدينة 22.666666667 | القاهرة 24.5 | الجيزة 31 | طنطا 40 | الاسكندرية

It should be noted that similar SQL queries can be applied to more than one table. Also, SQL can be generally used in one of three ways:

- Typing commands at a terminal and receiving immediate responses.
- Writing SQL modules that can be called from other languages
- Embedding SQL queries in a program written in another language.

4. Arabic SQL

In this section, we present Arabic SQL, which is the Arabic version of SQL. The basic Arabic SQL keywords (corresponding to the basic SQL keywords discussed in section 2) are shown in Table 2. The Arabic SQL queries corresponding to the SQL queries of Table 1 are shown in Table 3. In these queries, each SQL keyword is replaced by the corresponding Arabic SQL keyword, but unlike English, Arabic is read from the right to the left. Figure 3 shows a block diagram of Arabic SQL system. In this system, the user can type Arabic SQL إستخرج (SELECT) statements at the user interface. Since the SELECT statement is the most basic SQL statement, it was selected for our prototype.

Table 2. Basic SQL keywords and the corresponding Arabic SQL keywords.

Basic SQL Keywords	Corresponding Arabic SQL Keywords
SELECT	إستخرج
FROM	من
WHERE	حيث
GROUP BY	جمع باستخدام
HAVING	حيث بعد التجميع
ORDER BY	رتب باستخدام
AVG	متوسط
AS	بعنوان

Table 3. Arabic SQL queries corresponding to the SQL queries of Table 1.

Example Number	Arabic SQL Query
1	إستخرج * من الموظفون حيث المدينة = "القاهرة"
2	إستخرج متوسط (العمر) بعنوان [متوسط العمر]، المدينة من الموظفون جمع باستخدام المدينة حيث بعد التجميع المدينة = "القاهرة"
3	إستخرج متوسط (العمر) بعنوان [متوسط العمر]، المدينة من الموظفون جمع باستخدام المدينة رتب باستخدام متوسط (العمر)

The Arabic SQL system is formed of two main modules: the Arabic SQL lexical analyzer and the Arabic SQL parser. The function of the Arabic SQL lexical analyzer is to convert the character stream into a token stream detecting any spelling errors. In this lexical analyzer, we consider all of the reserved words as separate tokens, because it is the easiest thing to do. Names start with a letter and are composed of letters, digits, and underscores. This pattern follows all of the

reserved words so the reserved word patterns take precedence. The function of the Arabic SQL parser, on the other hand, is to detect any syntax errors in the combination of the generated tokens. The statement defined in the Arabic SQL parser is the manipulative SELECT statement which manipulates a database.

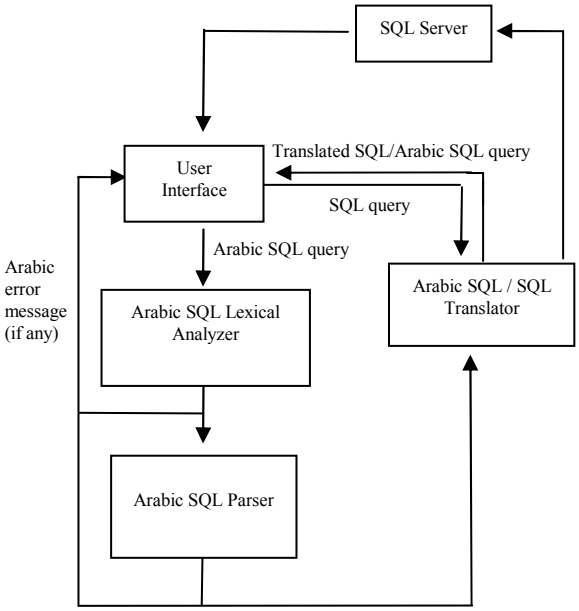

Figure 3. Block diagram of the Arabic SQL system.

More details on building a lexical analyzer and a parser for SQL can be found in [11, 12]. They explain how to use Lex and Yacc or Flex and Bison to generate lexical analyzers and parsers. A whole chapter in each book is dedicated to explain how to generate a lexical analyzer and a parser for SQL. Arabic SQL used a simplified version of these tools since it is only a prototype. Besides, the English characters and keywords have been replaced by the corresponding Arabic characters and keywords. The operation of the Arabic SQL system can be explained as follows:

- The Arabic SQL query passes through the Arabic SQL lexical analyzer to convert the character stream into a token stream detecting any spelling errors.
- The generated tokens are passed through the Arabic SQL parser for detecting syntax errors if any.
- Any errors detected by the Arabic SQL lexical analyzer or the parser are reported to the user through the user interface in Arabic to be easily understood by Arabic SQL programmers. Table 4 shows some example Arabic SQL errors, the corresponding error messages in Arabic, and their translation.
- To encourage the use of this Arabic version, Arabic SQL system is accompanied by a translator that can translate programs between SQL and Arabic SQL for portability. This is done in a manner similar to that of the assembler [9] using lookup tables. If no errors are detected, the Arabic SQL query is passed to the Arabic SQL translator to be translated to a SQL query. This SQL query is then returned to the user through the user interface to be used as required.
- Meanwhile, the SQL query is passed to the SQL server to be executed and the required information is returned to the user through the user interface.
- Alternatively, a SQL query submitted through the user interface can be passed to the SQL translator to be translated to Arabic SQL and returned to the user to understand it.

It should be noted that Arabic SQL system can process queries on more than one table. But, semantic errors such as an attempt to apply AVG keyword to a non–numeric field cannot be detected in the current version.

Table 4. Arabic SQL error messages and corresponding translation.

Example Number	Arabic SQL Query	Error Message	Translation
1	إستخرج * من الموظفون حيث المدينة = "القاهرة"	الرمز "من" لابد أن يأتي قبل الرمز "حيث"	The keyword FROM should precede the keyword WHERE
2	إستخرج * من جدول الموظفون حيث المدينة = "القاهرة"	الكلمة "جدول" غير مفهومة	The word جدول is unrecognized

5. Results of the Research Study

Arabic LISP and Arabic SQL were introduced to a sample of about 200 Arabic–speaking stakeholders from different disciplines and ages. All the selected stakeholders are not fluent in English. It should be noted that young novice programmers are normally expected to learn basic programming concepts, but older stakeholders from disciplines that are not computer-related are expected to learn how to use SQL to access database systems (at their schools or universities).

Stakeholders were all provided with descriptions of the basic commands in the English and Arabic versions of LISP and SQL. Afterwards, they were asked to write very simple programs using the different versions of LISP such as:

- Computing the average of three numbers.
- Forming a list out of two lists.
- Adding an element to the end of a given list.

They were also asked to write simple queries using the different versions of SQL such as:

- Retrieving one or two columns from a single table.
- Retrieving one or two columns from a single table under simple conditions.

- Retrieving one or two columns from a single table and sorting the result.

Those who failed to write the required programs and/or queries were allowed to see them at the end. Two different surveys were conducted, one for experienced programmers and one for novice programmers. The survey questions for experienced programmers were:

- Are you an experienced programmer?
- How old are you?
- Which programming languages do you use?
- Do you prefer using the Arabic version of Lisp or improving your English language?
- Why?
- Do you prefer using the Arabic version of SQL or improving your English language?
- Why?

The survey questions for young and old novice programmers were:

- Are you a novice programmer?
- How old are you?
- Which programming languages have you ever attempted to learn or use?
- Are you interested in learning basic programming concepts?
- If yes, do you prefer using the Arabic version of Lisp or improving your English language?
- Are you familiar with database systems?
- Are you interested in learning how to access database systems using SQL?
- If yes, do you prefer using the Arabic version of SQL or improving your English language?

The results of the conducted survey can be summarized as follows:

- About 98% of the novice stakeholders who are 14 years old or younger preferred the Arabic version of LISP to help them learn basic programming concepts. This is because Arabic LISP is similar to their natural language, but they did not show any interest in SQL.
- Older novice stakeholders from disciplines that are not computer-related did not show much interest in LISP, but about 82% of them preferred the Arabic version of SQL.
- About 85% of the stakeholders with programming experience preferred the English versions of both LISP and SQL to simplify communication with other programmers worldwide. They were ready to improve their English language rather than use the Arabic versions.

The above results can be rephrased and summarized as follows:

- Developing Arabic versions of general purpose programming languages is needed to help students at early ages learn programming concepts. Such students need to use a programming language that is like a language in which they are fluent.
- On the other hand, developing Arabic versions of special purpose languages is needed for older novice programmers, who are from disciplines that are not computer-related. Developing such versions implies a faster learning phase, better remembrance of the programming language syntax, easier programming, less errors, and more programmers.

6. Conclusions and Future Work

Facile programming implies the modification of programming languages so as to be easily learnt, remembered, and used by programmers from different disciplines. This is achieved by studying the practical difficulties that face such programmers and trying to tackle them. The paper addressed the difficulty of learning, remembering, and using common English-like programming languages by programmers who are not fluent in English. It is even harder for them to remember the English compilation error messages.

The paper presents an Arabic version of SQL, namely Arabic SQL in an attempt to figure out whether developing versions of common programming languages that are like natural languages of programmers would improve their programming capability. Syntax errors in Arabic SQL can be detected and the corresponding error messages are produced in Arabic. To encourage the use of Arabic SQL, it is accompanied by a translator that can translate programs between the English and Arabic versions of SQL for portability. Although the paper discusses Arabic SQL, it reports results from our experience with Arabic SQL and Arabic LISP [9].

In summary, developing Arabic versions of general purpose programming languages is needed to help students at early ages learn programming concepts. Such students need to use a programming language that is like a language in which they are fluent. On the other hand, developing Arabic versions of special purpose languages is needed for older novice programmers, who are from disciplines that are not computer-related. Developing such versions implies a faster learning phase, better remembrance of the programming language syntax, easier programming, less errors, and more programmers. As a future work, we intend to extend Arabic SQL to cover all possible SQL queries. Arabic SQL system should be also extended to be able to detect semantic errors. Finally, we intend to continue our work in facile programming by trying to study other practical programming difficulties and problems and trying to tackle them.

Our aim is to make programming easy and facile for all programmers regardless of their background.

References

[1] Al-Sadoun H., Yaseen M., El-Jallad A., and El-Jallad M., "ARbic BasIc (ARBI): A New Arabic MS-DOS Based Programming Language," *in Proceedings of 12th National Computer Conference and Exhibition*, Saudi Arabia, pp. 449-464, 1990.

[2] Al-Salman A., "An Arabic Programming Environment," *in Proceedings of ACM Symposium on Applied Computing*, USA, pp. 19-25, 1996.

[3] Aljanaby A., Abuelrub E., and Odeh M., "A Survey of Distributed Query Optimization," *The International Arab Journal of Information Technology*, vol. 2, no. 1, pp. 48-57, 2005.

[4] Amin M., "The Arabic Object-oriented Programming Language Al-Risalh," *in Proceedings of ACS/IEEE International Conference on Computer Systems and Applications*, Lebanon, pp. 424-427, 2001.

[5] Beaulieu A., *Learning SQL*, O'Reilly Media, USA, 2009.

[6] Bloch J., *Effective Java*, Prentice Hall, Pearson Education, USA, 2008.

[7] Elazhary H., "Arabic Lisp," *in Proceedings of SEKE*, USA, pp. 382-385, 2009.

[8] Gaddis T., *Starting Out with C++: From Control Structures Through Objects*, Addison Wesley, Pearson Education, USA, 2008.

[9] Hyde R., *The Art of Assembly Language*, No Starch Press, USA, 2003.

[10] Jurafsky D. and Martin J., *Speech and Language Processing*, Prentice Hall, Pearson Education, USA, 2008.

[11] Levine J., Mason T., and Brown D., *Lex and Yacc*, O'Reilly and Associates Inc., CA, 1992.

[12] Levine J., *Flex and Bison*, O'Reilly and Associates Inc., USA, 2009.

[13] Lukaszewicz L., "SAKO - An Automatic Coding System," *Journal of Annual Review in Automatic Programming*, vol. 2, no. 8, pp. 161-176, 1961.

[14] Rafea A., Soliman D., Samy E., and Felfela G., "Al-Daleel: An Arabic Interactive Programming Environment," *in Proceedings of 3rd International Conference and Exhibition on Multi-Lingual Computing*, UK, pp. 1-30, 1992.

[15] Seibel P., *Practical Common Lisp*, Apress Berkeley, USA, 2005.

[16] Tetsuji I., Hiroshi S., Osamu O., and Shunsuke U., "Grammar of a Japanese-Based Programming language "Mahoroba"," *Joho Shori Gakkai Kenkyu*, vol. 2001, no. 31, pp. 143-152, 2001.

[17] Touzi A. and Hassine M., "New Architecture of Fuzzy Database Management Systems," *The International Arab Journal of Information Technology*, vol. 6, no. 3, pp. 213-220, 2009.

[18] Viescas J. and Conrad J., *Microsoft Office Access Inside Out*, Microsoft Press, USA, 2007.

[19] Yaohan C., "Structure of a Direct-Execution High-Level Chinese Programming Language Processor," *in Proceedings of ACM Annual Conference*, CA, pp. 19-27, 1974.

[20] Wikipedia, available at: http://en.wikipedia.org/wiki/ Non- English-based_programming_ languages, last visited 2008.

[21] Jeemlang, available at: http://www.jeemlang.com/documentation/webframe.html, last visited 2007.

Hanan Elazhary received her BSc degree in Electronics and Communications Engineering and her MSc degree in Computer Engineering from the faculty of Engineering, Cairo University, Egypt. She received her PhD degree in Computer Science and Engineering from the University of Connecticut, USA. Currently, she is working as a researcher at the Electronics Research Institute, Cairo, Egypt. She is also working as a part-time assistant professor at several reputable universities in Cairo, Egypt. Her research interests include high performance computing (HPC), software engineering, artificial intelligence, and computer networks. Elazhary has supervised several graduation projects and is currently supervising one PhD student in the field of high-performance computing.