Olympiads in Informatics, 2014, Vol. 8, 35–48

Learning Strategies of Informatics Contestants

Gilberto CUBA-RICARDO, P. Alberto LEYVA-FIGUEREDO,
Laura L. MENDOZA-TAULER
José de la Luz y Caballero University of Pedagogical Science of Holguín. Cuba
email: {gilberto.cuba, albertoleyva, laura}@ucp.ho.rimed.cu

Abstract. This paper shows a study about learning strategies that informatics contestants use. These are the result of the experience in the topic through systematized interviews and participant observations by coaching the contest group.

This paper also details the methodology used and describes the logic we followed in order to determine the strategies used by contestants. Among these: Competitiveness to be "First", tendencies to use "Trial and Error", "Traceability" of source code, study of existing source code along with algorithm description, and use of mnemonic systems.

Keywords: learning strategies, programming contests, informatics olympiads.

1. Introduction

Knowledge competition activities, as instituted in the Ministerial Resolution 91/2007 (Cuban Ministry of Education); are designed to eradicate reductionist conceptions; such as identifying these activities only as competitions. The goal, then, is to transform them into creative environments where students can extend and deepen their general proficiency. In particular in subjects of their interest throughout the school year.

Recent research (Campos-Maura, 2006) has shown that knowledge competition: incentivize mass participation, develops interest in the subjects of study, improves the quality of learning and morally encourages the effort and work of students and teachers.

In Cuba, this activity is carried out on several levels of Education, from primary school to senior high school. Particularly, regarding this latter level of education, the Cuban Minister of Education, (Velázquez-Cobiella, 2011), speaking at the inaugural conference of "Pedagogía 2011" Congress, highlighted that "…the work done at Vocational Senior High School Institutes of Science is strengthened by stimulating students to continue university studies in science, a necessary condition for the scientific and technical development of the country".

Without a doubt, according to this idea, senior high school is the cornerstone of student education and future professional training according to Fidel Castro's ideas when he says: "The future of our country must be necessarily a future of men of science; it must be a future of men of thought".

Taking into account the premise that this level of education (senior high school) should decide the student's future profession, delving into the actual contents of the curriculum should not be enough. An environment where all the student's potential and capabilities can be developed must exist. Knowledge competitions provide such an environment, sorted by subjects and grades.

High school seniors can enroll in the competition's groups once they are identified and selected by the teacher acting as trainer (from now on: coach). These requirements also include: basic skills and potential related to the subject of their preference. The students selected are often known as contestants or challengers, and in the scientific literature are categorized as talented. This is manifested by their high degree of motivation to perform tasks and by their high performance and potential. This potential is evidenced in any of these fields separately or in combination: general intellectual skills, particular academic aptitude, creative or productive thinking, leadership skills, visual or performing arts' skills, and psychomotor skills (Marland, 1971).

Given that, special attention must be paid to students selected for competition (from now on contestants). We have created co-curricular environments in which the knowledge is deepened, theirs skills are prepared, and their potential developed and improved with the objective of achieving better performance and precision in the results of the competitions they carry out. All this increases the quality of their learning and prepares them for future life, thus achieving one of the aspirations expressed by the Minister of Education, this time at the "Universidad 2010" conference (Velázquez-Cobiella, 2010), when she say that is necessary: "... that students acquire basic knowledge to achieve higher levels of learning, with strength and possibilities of application, and the development of skills aimed at solving learning and social life problems".

Contestants not only prepare for their future profession, they also proudly represent their country in the knowledge and skills competitions at Latin American and international levels. In the process, contestants acquire a sense of belonging and identity for their nation. To this day, they have achieved outstanding results, including medals and the moral personal recognition they bring.

However, the decorous results obtained today do not meet the expectations of such talented students who have the intellectual abilities to solve efficiently and in a short amount of time the programming problems presented at these competitions. Therefore it is necessary to strengthen the intellectual formation of programming contestants. The ultimate objective being to integrate students into society, encouraging the scientific and technological development of the country. The reality on the current competitions is characterized by inadequate strategies and techniques to solve programming problems, which impedes the search for efficient solutions in the shortest time.

The purpose of this paper is to show contestants' learning strategies diagnosed during the competition activity. In addition, we consider the importance of each learning strategies and their correct use.

2. Informatics Contests Activity

Programming contests are defined in Cuba (Hernández-González, 2008) as curriculum activities where students develop their potential and talents, although the contents and skills to be developed in the activity are extracurricular. Algorithms and programming languages are not taught in the senior high school, only in a very introductory way during twelfth grade.

Contestant groups are made of students with high motivation and programming skills, the activity consists of two stages: training sessions and competitions. In both stages the fundamental activity is to solve programming problems that generally have an algorithmic nature (Verhoeff *et al.*, 2006).

According to Hernández-González (2008), the contestants need to develop some specific skills to set up their learning efficiently. These are:

- Write code to solve the proposed problem using a programming language.
- Develop a program using the advantages of an Integrated Development Environment (IDE) and debugger facilities.
- Create test cases covering all variants of the problem.
- Consider the elapsed time and memory available when choosing algorithmic solutions and data structures according to its limits.

Due to the nature of the programming competitions activity and specially training sessions, the contestants acquire an academic and scientific education; mainly because they have been researching new knowledge to solve programming problems. This improves the contestants learning strategies, and develops the skills that stimulate self-study and learning. These are important elements in the formation of a contestant who wants to grow the skills to solve programming problems.

Unlike academic training, the programming competition group also educates values of modesty, honesty, cooperation, solidarity, collectivism, and especially national identity. Moreover, it should create friendship among group members, and communication ought to flow spontaneously until the objectives are met. Therefore, one of the objectives of the Cuban Education is designed to improve the position, the quantity and quality of the Cuban winnings on the International Olympiad of Informatics (IOI); where to do so, we should improve the results of the National Informatics Competitions.

The development of contestants' skills depends on the learning process during the assimilation of the contents in the training sessions, and during the activity of solving programming problems that are performed in competitions. Given that, influencing the learning of each contestant is a key goal for coaches.

3. Intervention in the Contestant's Learning Process

Thus, in an effort to modify the contestant's learning process, the following actions have proposed. These are supported in the methodological work developed before the training sessions:

1. Diagnose the particular learning strategies of the contestants.
2. Determine the relationship between learning strategies used by contestants, and the activity of solving programming problems.
3. Address individual differences considering the diagnosed learning strategies.

4. Promote the conscious use of diagnosed learning strategies so contestants are able to control them.
5. Reorient the use of learning strategies that are not suitable for certain processes.
6. Teach new learning strategies that are not used by the contestants.

Learning strategies are described by several authors (Chamot and Kupper, 1989; Oxford, 2003; Weinstein and Underwood, 1985; Weinstein *et al.*, 1988; Carrasco, 2004), which emphasize the important role in the cognitive process in general. They are mostly recognized as a set of procedures, actions and activities used by individuals to acquire, store and/or use information in order to make the learning process more effective (Chamot and Kupper, 1989; Oxford, 2003; Carrasco, 2004; Ortiz *et al.*, 2007); and as a number of different skills that have postulated themselves as necessary, or helpful, for effective learning and retention of information for its later use (Weinstein and Underwood, 1985; Nisbet and Schucksmith, 1986; Weinstein *et al.*, 1988). Furthermore, learning strategies are kind of rules that make proper decisions in a certain time of the cognitive process (Ortiz *et al.*, 2007).

Considering the characteristics listed by each of these authors, and contrasting the contents of learning strategies with them, the assumption is that it recognizes learning strategies as a set of procedures, actions, activities used by individuals to acquire, store and/or use information in order to make more effective the learning process.

Knowledge of learning strategies used by programming contestants, as diagnosed by the coach, must be used to ensure that educational activities are prepared according to its advantages and disadvantages to improve the cognitive process during training sessions. To do this, it is necessary to know how strategies have classified, this will allow determining the "when" and "how" to carry out or reorient the learning strategies.

One classification that appears in the work of many researchers is the one that considers (Chamot and Kupper, 1989; Weinstein and Mayer, 1991; Oxford, 2003): cognitive strategies, metacognitive strategies and socio affective strategies, each one these as a kind of learning strategy.

4. Learning Strategies of Informatics Contestants

Based on theoretical considerations related to learning strategies and the previously stated actions on intervening in the learning process, it is necessary to improve the skills developed by contestants to solve programming problems of an algorithmic nature.

Several of the solutions commercially available (Weinstein and Underwood, 1985), and major of those (Learning and Study Strategies Inventory, LASSI; Learning and Study Questionnaire, LSQ; Shortened Experiences of Teaching and Learning Questionnaire, SETLQ; Experiences of Teaching & Learning Questionnaire, ETLQ; Visual Aural Read/Write Kinesthetic, VARK; Approaches to Studying Inventory, ASI) are positivist with a predominant quantitative paradigm. Their purpose is predominantly the assessment of students' awareness about the use of learning and study strategies. Ferrera (2008) considers that, when students answer the tool's questions, they do not objectively assess themselves, which mean that students are not using true learning strategies. To make the appropriate assessment of student knowledge and learning strategies, one must use the analysis of concrete execution on learning tasks.

Context

The programming competitions described here are carried out in one senior high school in the Holguín province. There, the incoming contestants are selected from their willingness to join the contests group and also according to interviews and test results. In the latter, the reasoning and informatics skills that the future contestant has are assessed. Therefore, students of 10th, 11th and 12th grade comprise the competition group.

The training takes place in extracurricular school sessions since there isn't any other time scheduled for training, which requires self-preparation of the contestant. In addition, training is carried out separately on the three grades. When a competitive event is near, training camps are created by grouping all the contestants. Training is performed mostly using Charguéraud and Hiron (2008) proposed method, hence the diagnosis of learning strategies are mainly aimed at learning programming, and not the algorithmic theory in isolation.

The timeframe of the study coincides with the years of the author's experience in the field, specifically the last three years.

The instruments and procedures are applied by the coach without the intervention of others. This avoids the introduction of extraneous variables in the system. Furthermore, the conditions created in the group are of empathy and trust, through the interaction of each training session, year after year. It also emphasizes that at no time should become explicit to the contestants that they are being evaluated as part of an experiment.

Participants

Year after year, in the programming competition group, the amount of contestants in the three grades is between 10 and 20 students (age $M = 16.28$, $SD = 1.23$; all male). This is the approximate amount of contestants that participate in the current research. In particular cases there were other samples that will be described as soon as needed.

Materials

Due to the variety of the learning strategies, various resources were used. In general, we used interviews by contestants and teachers; the participant observation keeping a written record of the contestants' behaviour; programming tasks; a digital recorder; and some applications (the C/C++ IDE, Code::Blocks, GNU C++ Compiler, "gcc" with GNU Debugger, "gdb", and a distributed revision control system, "git") configured together to monitor the contestants' actions during the problem solving process.

The interview questions are open, this makes it possible to collect as much information as possible, as well as to relate it to the results of the observation and interviews with other people in the contestant's social circle.

The most prevalent learning strategies determined in a diagnostic first step are: competitiveness to be "first", behavioural tendency to "trial and error", the "trace" of source code, reading source codes and problem solutions, and using mnemonic systems.

The following subsections explain some of these strategies, but that does not mean they are listed in order of priority or importance. In this paper, only the first step description of the previous actions is presented.

4.1. *Competitiveness to be "First"*

Training sessions are organized into classes that mostly assess the content taught in previous classes. These assessments are setting a global ranking about the contestant behaviour. Taking into account the ranking hierarchy, "first" contestants are taken into consideration to make the team that represents the school, province or country in any competition: IOI, National Informatics Contests, National Cups, etc. Hence, the motivation is to be the "first" or at least be among the "first ones".

There is a high degree of motivation triggered by the competition and the desire to represent, including: recognition from the contest group, other schoolmates or from community and family members. This stimulates the student to achieve stronger and efficient acquisition of knowledge, particularly in the activity of solving programming problems.

Revilla *et al.* (2008) mention some related ideas of the self-competitive behaviour when users participate in online judges. They recognize too, the importance of several learning strategies involved in the training process, which can be very positive for the student's formation and maybe neutralize the negative effects that many people impute to any kind of competitive learning activity (Revilla *et al.*, 2008).

To identify this learning strategy in contestants, some practical methods are used. For instance, in interviews, contestants were asked some of these questions:

- What interested you the most about joining the competition group? (Commonly known as elite group).
- What are your future aspirations about programming competitions?
- Do you like solving programming problems?
- What kind of feeling do you experiment while solving problems?
- What interest do you have in solving more tasks than others contestant?

The majority of these interviews (that we call "conversation") were digitally recorded and then analyzed for the occurrence of strategy processes and behaviours.

In addition, contestants were asked to express considerations and experiences about some of the colleagues individually and about the contests group in general. Not all interview questions are asked at the same time. Questions were asked according to the contestants' behaviour.

Some examples of the contestants' answers are: "When I am trying to solve a problem, I feel a new challenge that invites me to compete, then I begin to look for a solution and when I get it, I think that I can help other friends and demonstrate them that I can"; "I aspire to obtain a medal and join the national pre selection team", and the one which gave us the name of the current learning strategy, "I compete to be the first in the group (…)".

Besides this, the problems that the contestants tried to solve were monitored; and so were the participant observations. Some indicators were taken into considerations. For instance: what are the main subjects of conversation with their partners; what interest do they have for the celebration of competitive activities; how are they motivated by the content; what kind of relationships they have with their partners; were they able to express their ideas openly in the group. More indicators are registered by the coach and then checked against the contestants' behaviour and their answers to interview questions.

The contestant's behaviour changes constantly and it is impossible to assure that they have "some kind of motivation" to any subject. When contestants begin competing, their young age brings on an exploration of wishes, motivations and interests, and the measuring of their knowledge.

Hence, it becomes difficult to ensure the presence of this learning strategy. An example of this is shown in the table below (Table 1), in which it has been represented the state of this strategy (LE) in the contestants (C). In all cases, checks were made in the middle of the school year. In the second year, most of the contestants who were in 10th and 11th grade from last year were repeated. In these grades, the contestants are unstable. As a consequence, they could stop joining the programming competition group with relative ease.

Table 1 also shows this learning strategy occurs more in the contestants of 11th and 12th grade. Such is the case, that sometimes they prioritize the learning of content related to programming skills to the extreme and neglect the subjects of their grade level curriculum.

In general, this marked interest manifested by contestants has been identified as a socio – affective learning strategy. Therefore, this strategy is contrasted with some cited study tools, and the result is that there are coincidences. For instance, in LASSI with the "attitude" and "motivation" scales, and in ASI with the "intrinsic motivation", "extrinsic motivation" and "achievement motivation" scales.

Although this strategy is of great importance and has considerable influence on the contestants, the coach should be careful, as it can cause the formation of habits not consistent with the objectives outlined in the competition group. The contestants should always remember that although they compete with peers to be "first" and that most tasks are performed individually, they must show solidarity, be honest and also act modestly. In the competition group must prevail the collectivism and exchange of ideas.

Generally, with this strategy comes also motivation to learn new content, its peculiarities and practical applications. This provides an efficient support for contestants when trying to solve programming problems.

Table 1

Contestants' coincidences of learning strategy "competitiveness to be first"

Grades	First Year			Second Year			Third Year			C		LE	
	C	LE	%	C	LE	%	C	LE	%	M	SD	M	SD
10th	8	4	50,00	9	3	33,33	9	5	55,56	8,67	0,58	4,00	1,00
11th	6	5	83,33	7	6	85,71	6	6	100	6,33	0,58	5,67	0,58
12th	3	3	100	4	4	100	4	4	100	3,67	0,58	3,67	0,58
Totals	**17**	**12**	**70,59**	**20**	**13**	**65,00**	**19**	**15**	**78,95**	**18,67**	**1,53**	**13,33**	**1,53**

4.2. *Behavioural Tendency to "Trial and Error"*

When the contestant solve exercises that represent a programming problem, he has to search the algorithm that coded into a programming language will correctly provide the results to all provided data sets. During this process of heuristic search in which he is creating an algorithm, the contestant, using an IDE, implements the source code solution. Here can we notice the use of the current learning strategy, when the source code is compiled into a runnable program and executed to evaluate if the solution is correct.

To diagnose this learning strategy, the C/C++ IDE, Code::Blocks, and a distributed revision control system, "git", was used. "git" was configured within the Code::Blocks, so that when contestant send a compile and run order, a "commit" is made taking the changes produced into the source code. This allows to register date and time of execution, and register also the changes made in the source code since the last execution. The "git" also tracks changes in modified lines, which gives a measure of the amount of changes made during each execution.

Once they have configured these applications on each computer, they proceeded to perform tasks. Each contestant had to solve a daily exercise for three consecutive days. Each exercise lasted 80 minutes. At two and four months, the same process was repeated. Each three days were called a "season". In the second season, one exercise from the first season is repeated with a different description and the same input and output order and file structure. In the third season, one exercise from the first season was presented, making sure it did not match the one repeated in the second season.

Regarding the lines modified in the source code, this depends of the coding style used by the contestant, the programming language used, and of the algorithm. However, despite this, we reached some notable conclusions. For instance:

- The contestants who had a partial solution to the problem, made more executions than the ones who solved it correctly.
- There is an inverse proportionality between the amount of modified lines and executions of the program made by contestant.
- The process of observing the source code building in its various states, is of much greater value to study than to observe the final program.
- Overusing this strategy does not lead to good results in the ratings of the contestants and leaves them little time to reason out the solution to the problem.

Some features that were observed in this process are:

- In the first 10 minutes, executions are not manifested, which is pretty obvious, because the contestant is performing the exercise of reading; although, in the observations made, many contestants write during this time some basic instructions that are common to any program, and are recognized as a standard or template.
- As time progresses, the number of executions increases and the modified lines decreases considerably, demonstrating the use of this trial and error learning strategy.
- Most of the contestants who showed a large number of executions in the exercises, failed the same problem and did not reach the highest score in the exercises that were repeated in the second and third seasons.

- In the lasts 10 minutes, there was a slight decrease in executions.

During experimentation, the most significant question we asked ourselves was whether "the contestants actually learn using this approach". The conclusion was that they do, and this was demonstrated on the data collected each year. While we acknowledge that the sample is not as representative as to ensure precision; it gave us a trend of behaviour and that at least "something" was obtained. This made us wonder again about, what do they learn during this process? It was obvious that in the contestants' interviews we were not going to find the answer to the question. So, in reviewing the modified source code before each execution, we found the answer.

The review of the various states of the source code during building the solution algorithm revealed that contestants using this trial and error procedure adjust the implementation of a solution algorithm, so they improve their knowledge and the internal structures that compose it.

Besides, as a direct practical study linked to the behaviour of the contestant, it was determined from the regularities which data structures or algorithms are more difficult when being encoded by the contestant.

Although it is not considered an unsuitable strategy for study sessions that have a predominantly problem-centric approach, it is necessary to be careful with its use and especially the number of times a contestant uses the strategy. The internalization of instructions of a particular programming language is a slow process and it is improved with practice. Also with the help of other learning strategies that are more effective than this; an example of which is the "trace" of source codes. However, inside source codes that are quite large and which complexity increases because the complex data structures have much processing to do, it is necessary to use the current learning strategy to understand and to adjust in a quick way the logic of coded algorithm solution.

This learning strategy should not become a habit and common practice for the contestant, because it doesn't help the reflection and interpretation of source code, and therefore it is a warning point that coaches should notice for its proper use.

4.3. *The "Trace" of Source Code*

Good programming practices recommend, in many cases, the use of tools to trace the functionality of the program's source code. Although trace functionality appeared for other purposes (debugging or looking for errors in program source codes) today, it is an essential tool in learning and internalizing functionality for the user who begins in computer programming.

This sequential process of tracing or debugging is executed step by step, and it is memorized in the subconscious of the contestant to then use it in interpreting code that is subsequently analyzed without the debugging tool. Hence the importance of its use, it helps encoding better algorithms that are created to solve programming problems.

This is evidenced by the results obtained using the same procedure described in the previous strategy. The only difference is that the Code::Block was also configured to record, each execution of the debugger "gdb" that is integrated into the IDE. This made it possible to know the lines "traced" and the variables that were observed during this process.

Following analysis of the data collected, the contestants were interviewed to refine and clarify the inferences obtained from the traces. Based on regularities and the integration of these results it was determined that:

- Novice contestants use it more as a “tracer”, to understand how it was codified and what they aspire the implementation of the solution algorithm to be.
- More experienced contestants, use it as a “debugger”, to find and fix the semantic errors given in algorithms have been coded in the program.
- Total average time of debugging uses by contestants of each grades in the 9 exercises (the same exercises described in the previous subsection), showed a clear predominance of 10th grade over the rest grades.
- Most of the answer regarding the use of “trace” corresponds to its use adjusting the source code to the solution algorithm found.
- In general, all contestants, in the 9 exercises, use this learning strategy.

In addition, those allowed knowing that data structures and algorithms cause major problems in the implementation. It also allowed to attend the individual differences of the contestants and to make a more detailed work regarding their use.

Similar than the previous learning strategy, when a contestant traces or debugs a source code program, he adjusts the implementation of a solution algorithm, so he improves his knowledge and the internal structures that compose it, and also, learns about his errors.

Tracing source code is peculiar in the sense that it is a learning strategy for those novices in computer programming, and then it is transformed into a tool as contestants develop knowledge based on the programming logic. It ceases to be a contestant’s learning strategy to become a tool that is only used in some cases while trying to find the logic expressed in a source code of a program, or finding certain semantic errors that have been introduced during the encoding process.

Improving the use of this learning strategy is one of the pillars in programming teaching when the contestant begins his or her work on the competition group. For that, in many cases, the source code of program examples is given and contestants should be able to interpret and to deduce the solution.

4.4. *Reading Source Codes and Problem Solutions*

During the solving of programming problems, the contestants execute a program which solves the exercise and validate it against provided data sets, searching the validity and effectiveness of the response to each data set. Sometimes, the contestant does not reach the true solution, and needs impulse and stimulus oriented to enhance the knowledge already obtained. This is most likely the basis of the correct solution. However, even insisting on the coach presence to mediate in the problem solving process, the contestant does not always achieve right the solution.

In response, the contestant looks for strategies to acquire the knowledge that helps solving the problem correctly. Consequently, they rely on solutions that are part of the source code previously developed by another contestant or coach, or the description of the algorithm (or combination algorithms) that solve the problem. Contestants do not always have the description or source code, so it is necessary that communication between group members flow freely; it is common some contestants have solved the problem before or know a possible solution.

Shown in Table 2, below, is a comparative study regarding the behaviour of the contestants (C) after solving each exercise, consultation to the description of the solution algorithm (SA) and the source code (SC) program. This was sampled over the first three years of the first season described in 4.1. For each, the correct solutions (CS), the partial solutions (PS) and incorrect (IS) were quantified.

From the analysis of these data and after reviewing the responses from the interviews with the contestants, the following regularities were found:

- When the solution is correct, the contestant rarely looks at the algorithm description, and even less at the source code.
- When the solution is partial, the contestant first reads the algorithm's description, then checks that the algorithm matches the one he used. After that, he checks the source code, trying to find necessary adjustments that this source code needs to be correct or fulfil the time and memory restrictions.

Table 2

Contestant's consultation of the algorithm's description and source code

		First Task				Second Task				Third Task			
		CS	PS	IS	T	CS	PS	IS	T	CS	PS	IS	T
10th Grade	**C**	2	3	3	8	0	5	3	8	3	3	2	8
	SA	0	2	1	3	0	2	3	5	0	2	1	3
	%	0	66,67	33,33	37,50	0	40	100	62,50	0	66,67	50	37,5
	SC	0	1	1	2	0	1	3	4	0	0	0	0
	%	0	33,33	33,33	25,00	0	20	100	50	0	0	0	0
11th Grade	**C**	1	3	2	6	0	3	3	6	1	4	1	6
	SA	0	3	2	5	0	3	3	6	0	4	1	5
	%	0	100	100	83,33	0	100	100	100	0	100	100	83,33
	SC	0	2	2	4	0	2	1	3	0	3	1	4
	%	0	66,67	100	66,67	0	66,67	33,33	50	0	75	100	66,67
12th Grade	**C**	1	2	0	3	0	1	2	3	0	2	1	3
	SA	1	2	0	3	0	1	2	3	0	2	1	3
	%	100	100	0	100	0	100	100	100	0	100	100	100
	SC	0	2	0	2	0	0	2	2	0	2	1	3
	%	0	100	0	66,67	0	0	100	66,67	0	100	100	100
Totals	**T**	4	8	5	17	0	9	8	17	4	9	4	17
	SA	1	7	3	11	0	6	8	14	0	8	3	11
	%	25	87,5	60	64,71	0	66,67	100	82,35	0	88,89	75	64,71
	SC	0	5	3	8	0	3	6	9	0	5	2	7
	%	0	62,5	60	47,06	0	33,33	75	52,94	0	55,56	50	41,18

- When solutions are partial and the description doesn't match the algorithm, the contestant experiences disappointment. In most cases, he or she does not check the source code.
- The contestants with more experience mostly consult the proposed algorithm, not the program's source code.
- Some contestants that prefer that a partner reads it and comments on the algorithm to use.

It is important to point out that the contestant is not forced to consult on the description and the source code when trying to solve the task. On the contrary, the necessary mechanisms are created for them to feel the necessity to study them as a part of self-learning. The first learning strategy described in this article influences this behaviour, along with the degree of motivation the contestant exhibits when applying previously acquired knowledge. However, it is necessary that he always consult at least the description of the proposed algorithm. This knowledge offers the tools to solve the current task as well as future problems.

The act of reading source code and algorithm description develops the independence contestants need in order to: acquire new knowledge, perfect implementation techniques, increase understanding of source code and improve preparation to solve problems of similar nature.

This learning strategy, when used properly, is remarkable for strong and accurate knowledge during cognitive process.

5. Other Learning Strategies

The technological influence in the training sessions of contests group allows other kinds of learning strategies that improve the use of autonomous learning. These strategies are within the cognitive type when reinforcing and applying the content previously received. An example of this is when the contestant makes summaries in digital documents or animations that visually show certain geometric and mathematical algorithms, among others. Also considered as metacognitive strategies are those that require planning. A set of actions and steps for reading and interpreting new algorithms, or the order in which contestants try to solve exercises from easy to more difficult.

Similarly, coaches can encourage the use of learning strategies common in other areas of knowledge, but are peculiarly absent in informatics contestants. Examples of these strategies are the graphic representation of certain data structures, and the creation of conceptual maps that summarize the content previously learned.

It may seem to the reader as if only the listed learning strategies are manifested in the informatics contestant, but this is not so. The challenge is to diagnose, discover and study them when they occur during training sessions or competitions, in order to improve learning.

Confirming Chamot and Kupper (1989) ideas, in interviews with contestants, they do not reflect and recognize the use of learning strategies. More of them express the use of strategies to solve problems and not for learning, confusing one with another.

It is necessary to note, that although there is little relation between them, learning strategies should not be confused with strategies for solving programming problems. The first have a very marked objective and addressed to what it is explained in this paper. The second are aimed to achieve in the shortest possible time, the solution to the problem, through a source code that must fulfil the requirements as expressed in the exercise.

6. Conclusions

All of the learning strategies explained in this paper are used unconsciously by contestants and are not recognized as such by them (self-learning strategies). There are other learning strategies that contestants use during competition, and with the practical methods to collect data as Code::Blocks C/C++ IDE + "git", the participant observations and interviews, more of them are identified.

Science, especially Pedagogy plays an important role in the study of this issue that is manifested in the didactic of computer science, and specifically of computer programming, since promoting these activities causes an individual impact, collective and social, that is triggered by the results obtained during national and international competitions.

Learning is not only manifested when a new programming language or a new algorithm is learned, but also when the contestant solves a task that constitutes a problem for him.

The method proposed by Charguéraud and Hiron (2008) has been applied in our informatics contests activity, it has created some learning strategies in contestants.

References

Campos-Maura, E. (2006). *Estrategia Metodológica para la Preparación de Alumnos que Participan en Concurso de Español-Literatura en Preuniversitario.* PhD thesis in Pedagogical Sciences, Villa Clara, Cuba.

Carrasco, J.B. (2004). *Estrategias de Aprendizaje para Aprender Más y Mejor.* RIALP (Eds.), Alcalá, Madrid.

Chamot, A.U., Kupper, L. (1989). Learning strategies in foreign language instruction. *Foreign Language Annals*, 22(1), 13–22.

Charguéraud, A., Hiron, M. (2008). Teaching algorithmics for informatics olympiads: the french method. *Olympiads in Informatics*: *International Journal.* 2, 48–63.
`http://www.mii.lt/olympiads_in_informatics/htm/INFOL018.htm`

CUBA. (2007). *Ministerial Resolution 91/2007: Indications Related with Application of Knowledge and Skill Contests.* Ministry of Education, La Habana, Cuba. (In Spanish)

Ferrera, R.A. (2008). *Estrategias de Aprendizaje. Construcción y Validación de un Cuestionario-Escala.* PhD thesis. Universitat de Valencia, Servei de Publicacions.

Hernández-González, F.J. (2008). *Metodología para el Entrenamiento de los Estudiantes de Preuniversitario que Participan en Concurso de Informática.* PhD thesis in Pedagogical Sciences. Villa Clara, Cuba.

Marland, S.P., Jr. (1971). *Career Education: A Report.*
`http://www.eric.ed.gov/PDFS/ED080743.pdf`

Nisbet, J., Schucksmith, J. (1986). *Estrategias de Aprendizaje.* Madrid: Santillana.

Ortiz, L., Salmerón, H., Rodríguez, S. (2007). La enseñanza de estrategias de aprendizaje en educación infantil. *Profesorado. Revista de Currículum y Formación del Profesorado*, 11, 2.

Oxford, R.L. (2003). Language learning styles and strategies: an overview. *Learning Styles & Strategies/ Oxford, GALA*, 1–25.

Revilla, M.A., Manzoor, S., Liu, R. (2008). Competitive learning in informatics: the uva online judge experience. *Olympiads in Informatics. International Journal.* 2, 131–148.
`http://www.mii.lt/olympiads_in_informatics/htm/INFOL035.htm`

Velázquez-Cobiella, E.E. (2011). El reto a la educación: el futuro de las naciones latinoamericanas y caribeñas. In: *Conferencia Inaugural, Pedagogía 2011*. La Habana. Cuba.

Velázquez-Cobiella, E.E. (2010). La Educación en Cuba y los retos del personal docente. In: *7mo Congreso Internacional de Educación Superior. Universidad 2010*. La Habana. Cuba.

Verhoeff, T., Horváth, G., Diks, K., Cormack, G. (2006). A proposal for an IOI syllabus.*Teaching Mathematics and Computer Science*, 4(1), 193–216.

Weinstein, C.E., Wicker, F.W., Cubberly, W.E., Roney, L.K., Underwood, V.L. (1980). D*esign and Development of the Learning Activities Questionnaire*. (Tech. Rep. No 459.). Alexandria, Va: U.S. Army Research Institute for the Behavioural and Social Sciences.

Weinstein, C.E., Goetz, E.T., Alexander, P.A. (1988). *Learning and Study Strategies: Issues in Assessment, Instruction and Evaluation*. Nueva Cork: Academic Press.

Weinstein, C.E., Mayer, D.K. (1991). Implications of cognitive psychology for testing: contributions from work in learning strategies. In: Wittrock, M.C., Baker E.L. (Eds.), *Testing and Cognition*. Englewood Cliffs. N.J.: Prentice-Hall.

Weinstein, C.E., Underwood, V.L. (1985). Learning strategies: the how of learning. In: Segal, J.W., Chipman, S.F., Glasser, R. (Eds), *Relating Instruction to Research* (*Volume 1 of Thinking and Learning Skills*). London, Lawrence Erlbaum Associates.

G. Cuba-Ricardo is a doctoral student at the Curricular Doctorate of the University of Pedagogical Sciences of Holguín, Cuba. He is a researcher at the Department of Resource Development for Learning. His main research interest is the role of computer programming in educational processes and contestant training. He is a consultant and a coach of the informatics contests team in Holguín city.

P.A. Leyva-Figueredo holds a Ph.D. in Pedagogical Sciences. He is the Director of the Centre of Studies for Labor Education and Coordinator of the Doctorate Collaborative Curricular Program at the University of Pedagogical Sciences of Holguín. He has an extensive experience in labor education, doctoral training, research methodology, professional skills and professional guidance. Also, he is a member of the Provincial Scientific Committee of Science, Technology and Environment (CITMA) and a Permanent Member of the Evaluation Board to get the Scientific Degree of Doctor in Pedagogical Sciences.

L.L. Mendoza-Tauler received her Ph.D. in Pedagogical Sciences in 2001. She is the Director of the Centre of Studies in Educational Research at the University of Pedagogical Sciences of Holguín and Coordinator of the Ph.D. Program of UBV-2 Caracas, Venezuela. She teaches at the Ph.D., Masters and Qualified Program Studies in Venezuela and Perú. She is a member of the Academy of Sciences of Cuba in the Commission of Social Sciences. She is also a member of the Provincial Scientific Committee of Science, Technology and Environment (CITMA) and of the Evaluation Board to get the Scientific Degree of Doctor in Pedagogical Sciences.