Brain-based Programming

Barbara Sabitzer
Department of Informatics Didactics
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria
barbara.sabitzer@aau.at

Sandra Strutzmann
Department of Informatics Didactics
Alpen-Adria-Universität Klagenfurt
Klagenfurt, Austria
sstrutzm@edu.aau.at

Abstract**—***Learning languages can be hard. As the yearly results of the course "Introduction to structured and object-based programming" at our university show, learning the first programming language might be even harder. Many students complain about the difficulty of the course and fail in the exam. With the desire to support the students and enhance the learning outcomes we initiated the project "Brain-based Programming". The basic question is: "How can learning to program be made easier?" The answer may come from the interdisciplinary field of neurodidactics that offers many general suggestions for improving teaching and designing teaching material. But concrete examples for computer science education are scarce, and empirical research is still missing. This was the impetus for the project "Brain-based Programming" that aims at (1) creating and evaluating a brain-based script for beginners in Java programming and at (2) implementing and evaluating brain-based teaching methods in the programming course. In the pilot phase we conducted a didactic experiment in one of seven parallel groups and combined brain-based teaching methods and exercises. The results demonstrate the success of the experiment and support the hypothesis that learning is more effective when it considers how the brain learns and follows neurodidactical principles.*

Keywords—brain-based learning; neurodidactics; cooperative learning; programming;

I. Introduction

Learning languages can be hard. As the yearly results at the University show, learning the first programming language might be even harder. Many students taking the course "Introduction to structured and object-based programming", shortly "ESOP", complain about the difficulty of the course. But more important and alarming, many students fail at completing the course. With the desire to support the students and enhance the learning outcomes of this course, the idea for the project "Brain-based Programming" was born.

The basic question is: "How can learning a programming language be made easier?" Answers may come from two directions: Firstly, in the literature about introductory programming learning different approaches can be found, e.g. the use of a three dimensional animation software [1] or interactive learning objects [2]. Furthermore, all attempts to increase motivation may be helpful as described e.g. in [3].

Other answers can be found in the field of neurodidactics and educational neuroscience that offers mainly general suggestions for improving teaching and designing teaching material by considering the functioning of the brain [4], [5]. But concrete examples for the field of computer science education are scarce [6], [7] and empirical research is still missing. This was the impetus for the project "Brain-based Programming" that wants to build a first bridge between theory and practice - between neurodidactical research and its application in computer science education.

The aims of the project are:

(1) Developing, testing and evaluating a "brain-based" self-learning script for beginners in Java programming and

(2) Implementing and evaluating brain-based teaching methods in an university course for beginners in Java programming.

This paper first describes the basis of the project "Brain-based Programming", the neurodidactical principles behind the concept, and reports on the pilot test: a didactic experiment in the programming course, its evaluation by the students and the learning results in comparison with the other parallel courses. These results demonstrate the success of the experiment and moreover the principles behind it. The paper will close with the discussion of the results and an outlook on future work in this field.

II. Neurodidactics – The Basis of Brain-based Programming

A. The Basis: Neurodidactics – A New Way for Teaching?

Neurodidactics is an interdisciplinary research field that combines findings of brain and memory research, psychology, pedagogy and didactics that shall help to improve teaching and learning. The term neurodidactics, a combination of neuroscience and didactics, was proposed by a German mathematician in order to emphasize the interdisciplinarity of the field. In the English-speaking world two other terms for similar research have become widespread: Educational Neuroscience and Brain-based Learning. All three have the same goal: the improvement of teaching and learning by considering how the brain works. The contributions come from three directions:

1. *Neuroscientists* inform about structure and functions of the brain that might be useful for teachers. They also give suggestions for the improvement of pedagogy and didactics, but sometimes without considering that the everyday life in schools and university courses often deviate from the experimental settings of their research. [8]

2. Contributions from authors of different disciplines (not neuroscience), criticize the current pedagogy and offer more or less useful guidebooks for brain-based learning. Unfortunately, in the booming business of brain-based learning some neuro-

myths like the left brain/right brain oversimplification or the use of only 10 % of our brains persist. [8], [9]

3. Educators and didactics experts who work out neuropedagogical or neurodidactical concepts based on brain and memory research offer the third approach. Their point of view can be important when they consider the real conditions in the classroom. But often the suggestions are too vague and general like “Learning should be fun”. [8]

Neuroscience can do much for education but only when neuroscientists work together with didactic experts and teachers and when neurodidactical concepts are investigated empirically [8]. The project “Brain-based Programming” and the planned follow-up project “Teaching informatics with the brain in mind” shall be a step in this direction.

B. Neurodidactical Principles for Brain-based Programming

The most important principles that were considered in the project “Brain-based Programming” are the following:

1. Knowledge cannot be transferred. It has to be generated in each student’s brain [10].

Learning contents can be provided to students in different ways; but storage in the long-term memory is only possible when the input is actively processed in the brain of each student. Students should be active instead of only listening to a lecture and they should be supported to discover structures and rules themselves. This works best in open lessons where they can follow their own learning rhythm. [11]

2. Learning through imitating

Mirror neurons enable us to understand, interpret and imitate observed actions and to predict their results. They are responsible for learning through mirroring respectively imitating [4]. Students need models that they can imitate (e.g. role models, step-by-step instructions, solution-based learning). This activates also the brain mechanism of pattern recognition.

3. The brain recognizes and produces patterns, categories and rules itself [12].

Pattern recognition or patterning is a basic function of the brain. It helps us to extract rules and structures from available examples. So students do not need declarations and rules, but good and meaningful examples (e.g. correct program code, role models, step-by-step instructions, etc.) to understand the structure and extract the essential rules [12]. Using the function of patterning e.g. in discovery learning leads to a more active and therefore deeper processing and retention of information whereas lecture usually results in the lowest degree of retention (see fig. 1). [4]

4. New content is always built on existing knowledge and learning occurs through associating.

In this context knowledge is not only subject-specific but means also experience as well as knowledge from all living areas and from the world of the students. The physiological basis for this principle is that learning occurs through creating new or strengthening existing synapses (connections between neurons). This previous knowledge is individually different, which means that students should have the possibility and time to ask their individual questions to the teacher and/or peers respectively peer-tutors. [4, 12]

5. Learning is more effective when it makes sense and has meaning [4].

This criterion should be considered where possible in the selection of topics, tasks and products to be developed by the learners. The students should have the possibility of choice between different competence-oriented exercises and different topics. This increases motivation and, hence, may increase the learning outcomes, too. [4]

6. The brain needs time for consolidation.

The brain needs time (short breaks) to consolidate new information. Only then it can be permanently stored in the long-term memory. This should be considered in the structure and organization of the lessons. If the brain does not get time for consolidation (e.g. in usual lectures of 90 minutes) new information can overlay earlier content and/or information heard before can inhibit the processing and memorizing of new input. [4, 11]

7. The instruction method has impact on the retention of new information [4].

Besides practice by doing cooperative learning settings like group discussion and peer tutoring (teaching others) seem to be the most effective methods concerning retention as shown in fig. 1.

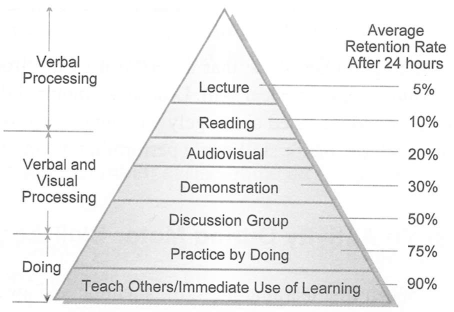

Fig. 1. Average percentage of retention of material after 24 hours [4]

An evaluation of more than 800 meta-analyses considering millions of students confirms the effectiveness of cooperative learning environment and shows an overall effect size of d= 0.41 [13]. The success of cooperative learning can be explained by the fact that it supports the memory process: cooperation and communication always require a recall from the memory and each recall or retrieval always causes a restart of the whole memory process. It leads to new processing, new storing and also new and stronger anchoring of the information in the long-term memory. Cooperative methods can therefore support the learning and memory process and are an essential part of the new brain-based programming concept. [4, 11]

III. The Project “Brain-based Programming”

A. About the Project

“Brain-based Programming” is the pilot project for a larger empirical study about “Teaching Informatics with the brain in mind”. It is based on the hypothesis that teaching and learning can be more effective when neurodidactical principles are considered in the design of tasks and exercises as well as in the lesson structure and the teaching methods. The phases of the project are the following:

1. The development of tasks and worksheets under consideration of neurodidactical principles as well as their first evaluation in the university course "Introduction to structured and object-based programming" (2012).

2. A didactic experiment: Application and evaluation of "brain-based" teaching methods of the developed worksheets in one of seven parallel practical courses "Introduction to structured and object-based Programming" (winter term 2012/13).

3. Evaluation and adaptation of the worksheets, aiming to develop a self-learning script for beginners in Java programming based on neurodidactical principles (2013).

The project is currently in phase three, where the results of the brain-based course are taken into account and the student feedback is used to redesign the worksheets for the following self-learning script.

Before describing the didactical experiment the following section gives a detailed description of the developed worksheets and the different types of "brain-based" tasks, always referring to the related neurodidactical principles. After that we present the course organization in the experimental group, the applied teaching methods and the form of assessment.

B. Brain-based Tasks

Usually, all students of the parallel courses get the same worksheets. The experimental group got them, too, but only for voluntary work and in order to provide a broader variety of exercises. Additionally, they got the new "brain-based" worksheets because one reason for the difficulties in the course may lie in complex contents (e.g. mathematical problems) or too complicated instructions of the existing worksheets that are often incomprehensible for beginners.

Every worksheet is divided into three parts and contains the following types of exercises:

1. Reading exercises

The first part of the worksheet shall foster discovery learning and take advantage of the automatic brain function of patterning. It contains the following types of tasks:

- Reading corners with a simple and small, but complete piece of Java program code, which is accompanied by some questions that should lead the learners and help them to discover the structures and rules behind.
- Puzzles of jumbled program code or cloze-tests where small parts of the code are missing.
- Step-by-step instructions including also the whole sample solution, which help to comprehend and reproduce a task by taking one step after another.
- Mini exercises or short tasks including a sample solution that the students can use immediately in the sense of discovery learning or after having resolved the task for verifying their own solutions.

2. Competence-oriented tasks

This part provides short and rather easy competence-oriented tasks covering different contents and topics that may have sense and meaning for the students, e.g. programming a course schedule or a vocabulary trainer. Following the principle "practice makes perfect", the students get a big variety of tasks that they can choose according to their individual interests, competences and needs. This may increase motivation, occupy all students with useful tasks and may therefore support the memory process and enhance the learning outcomes, too.

3. Programming tasks

The last part of the worksheet provides different tasks for small, complete programs or some subprograms as parts of a complex semester topic that have to be assembled to complex Java project at the end of the course. This considers a neurodidactical principle already postulated by Aristotle "The whole is more than the sum of its parts" or, as defined in neurodidactical literature, "Learning is more effective when it considers the whole AND the details" [14]

C. Brain-based Lessons: A Didactic Experiment

Usually in the practical programming courses only two settings are used in turns – *laboratory* in one week, where the students had to do a part of the exercises that they continue at home, and *presentation* in the following week, where one student presents his/her solution and the others are passive. In the experimental group we tried to offer more possibilities and to keep all students active according to their individual capabilities. This satisfies the neurodidactical principle that learning has to be active.

In the very first lesson of the experimental group the students had to do a self-evaluation of their programming skills by means of a competence grid that should help them categorize their capabilities and competencies. Based on this self-evaluation, three groups were formed:

1. Professionals: students with solid knowledge about the topics in the course and the ability to help their colleagues as peer-tutors or peer-teachers.

2. Amateurs: students with some experience in programming, but who did not feel able to assist their colleagues.

3. Beginners: students without any programming experiences.

These groups were formed because we tried to take into account, as much as possible, the individual experiences, talents and needs of each student. The neurodidactical principles "New content is always built on previous knowledge" and "Learning is more effective when it makes sense and has meaning" were considered, as the students could choose between different types of tasks, topics and collaboration in the course. Whereas the advanced students solved more complex projects and/or acted as peer-tutors and peer-teachers the beginners got more short competence-oriented exercises and could benefit from the peer-tutors.

The weekly lessons of 90 minutes each were characterized by an open learning setting considering the individual learning rhythm that allowed time and room for individual interests and needs. In general the lessons were divided into the following three phases and, when required, supplemented by a short lecture of the teacher or a peer-teacher: Asking questions, discovering and laboratory with pair programming.

1. Asking Questions

In the first phase (ten to fifteen minutes) the students worked together in small groups with one peer-teacher or peer-tutor (one "professional" or "amateur" student). In this time, the students got the chance to ask any question they had in mind. The idea behind was to encourage them to ask also questions that they would have never asked in the whole group, perhaps because they considered them too trivial or too stupid. But sometimes these questions and certainly the corresponding answers can help to understand a concept because they build up on the previous knowledge of the students…

2. Discovering

In the second phase (ten to twenty minutes) the students worked in small groups, too, each of them guided by a peer-tutor. On the base of the different reading exercises, already described in section A *Worksheets and Tasks*, as well as short video clips they tried to discover new topics or re-discover topics they had already learned before. This active way of processing input takes advantage of the brain's automatic pattern recognition and rule extraction, which may lead to a better and stronger storage in the long-term memory [12]. The video clips of some minutes offer a further advantage: the multimedia or modality effect, investigated by [15]. When information is double coded, e.g. text combined with corresponding pictures or animations, it can be remembered better.

3. Laboratory: Pair Programming

The third and last phase in the course was the laboratory. The students were ought to solve the tasks of the given worksheets by collaborating in pairs according to a well known and effective software engineering method: pair programming. This setting allows cooperation and communication, which may enhance learning because, as mentioned above, each recall from the memory (which is necessary when discussing the problem of the task and the way to solve it) restarts the whole memory process.

Depending on the topic and the needs of the students, these three phases could differ from time to time. Worth to mention is that all students were active in all phases. Due to the open learning setting the lecturer had enough time to visit the small groups, one after another, to ask questions for checking participation and comprehension as well as to help them with upcoming difficulties. At the second half of the lesson, a student tutor of a master course assisted the teacher in answering questions and helping the students solving the exercises.

The open learning setting allowed the students to follow their own learning rhythm, and hence consider the individual consolidation phase of the brain. Sometimes a supplementary phase of lecture was necessary where the lecturer or a peer-teacher explained a topic or concept for all students in a maximum of twenty minutes. More is not useful because we know from neurodidactics that the attention decreases approximately after this period. The following "down-time" with a low attention level shall be used for consolidation and revision. Fig. 2 visualizes a possible segmentation of a learning episode. After presenting new information in the first prime-time and some practice in the following down-time a phase of closure or summary of the new information is offered.

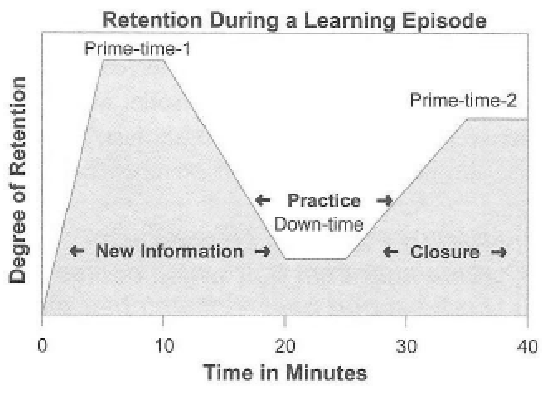

Fig. 2. Retention during a learning episode [4]

D. *The assessment*

The standard assessment criteria of all parallel courses of the "Introduction to structured and object-based programming" were valid for the experimental group, too, but were slightly adapted as follows:

- The course grade consists of 50% written exams a mid term and a final exam, each graded with max. 25 points and 50% participation in the course (number of correctly solved exercises, presentations, active participation in the lessons).

To get a positive grade, the students had to fulfill the following requirements:

- Compulsory attendance is required in at least 85% of the lessons.
- At least 50% of the tasks of each worksheet must be completed and the program code must be executable. As the students in the experimental group got the double number of worksheets and tasks, they had more choice. To offer a fair grading method, each task of the worksheets (the regular ones for all students as well as the "brain-based" tasks for the experimental group) was evaluated with points. Whereas usually only completely executable solutions are counted, the experimental group had a graduated assessment. The number of points depended on the percentage of the available correct program code. To give fair points, four categories were introduced: (1) complete and executable, (2) complete, but not executable (with slight errors), (3) program code mostly complete and (4) less than 50%. This concept of evaluation was introduced to motivate the students to do their own work, to avoid copying and to value mistakes, as we learn from them.
- A minimum of 50% of the written exams the same for all parallel groups must be achieved, that means 12.5 points for each, in total at least 25 points. Students who

don´ t reach half of the points in the mid term exam usually have to terminate the course. This was slightly adapted in the experimental group. The students had to achieve the total minimum of 25 points, too. But if one reached less than 12.5 points in the mid term exam s/he could compensate that in the final exam. This opportunity should motivate the students to remain in the course, to work more and to catch up on the missed topics. Whereas many students of the parallel courses had to terminate after their negative mid term exam two of three students in the experimental group actually could benefit from this criterion and terminate the course with a positive grade.

IV. Evaluation and Results

Brain-based programming was implemented as a didactical experiment in one of seven parallel groups with 21 of 126 students. Due to this small group and a certain teacher bias the positive results of this pilot project are not statistically relevant, but seem to support the hypothesis that learning can be more effective when brain and memory functions are considered in lesson and task design. This will be verified in a larger follow-up study at university and secondary school level.

To measure the acceptance and benefit of the brain-based methods and tasks, the students of the experimental group were given surveys at the beginning, the middle and the end of the semester. Furthermore, the university's official student feedback, completed by 15 of 21 students, has been considered.

Taking a closer look at this official feedback with grades from on a scale from 1 (best) to 5 (worst), it can be said that the experiment was successful. The "brain-based" course was graded with 1.1, whereas the average grading of all equivalent courses in the last three years lies at 2.1. Moreover, the students were very satisfied with the concepts used in this course. According to the students' verbal feedback, most notably and helpful are the concepts of peer-tutoring and peer-teaching, reading exercises discovery learning, mini exercises including solutions, step-by-step instructions, and the revision worksheet at the end of the course.

This confirmed the mid term feedback where 19 of 21 students graded the methods and tasks from 1 = very useful to 4 = not so useful (fig. 3).

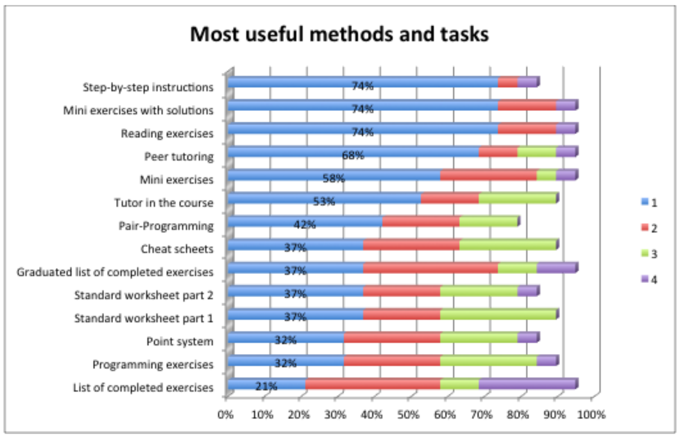

Fig. 3 Mid term evaluation of methods and tasks

Step-by-step instructions, mini exercises with solutions and reading exercises were considered as most useful tasks by 74% of the students. Concerning the teaching methods mainly peer tutoring (68%), the tutor from a master course (53%) and pair programming (42%) were considered very useful. Compared to the standard worksheets part 1 and 2 distributed in all parallel courses all new "brain-based" tasks (except programming exercises) and methods scored better.

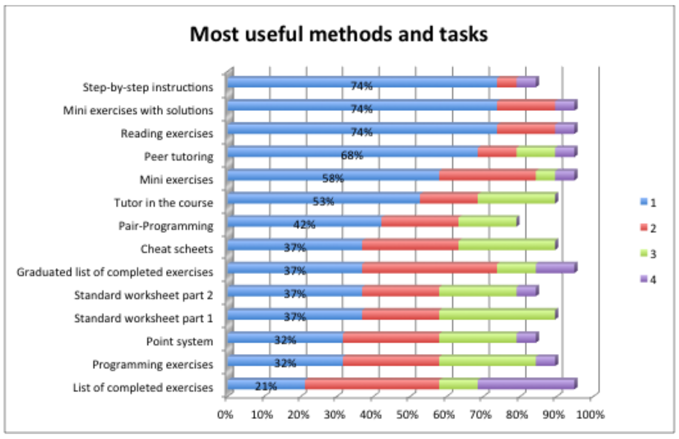

Fig. 4 Mid term evaluation of methods and tasks

The evaluation of the last survey (fig. 4), completed after the course only by 11 of 21 students, shows similar results.

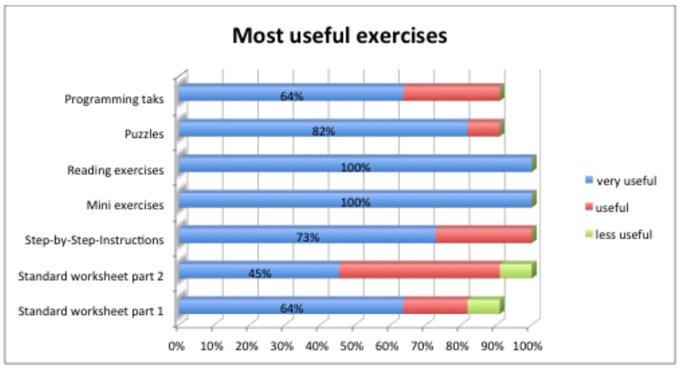

Fig. 5: Evaluation of the different exercise types

The students appreciated all forms of material for discovery learning, mainly the mini exercises including solutions and reading exercises (100% considered them very useful). They are followed by puzzles (82%) and step-by-step instructions (73%), which are other types of reading exercises. This shows that learning from worked-out examples, and benefiting from the automatic brain function of pattern recognition can be very useful.

The final evaluation of the methods shows the following results: All students found that the sample exams offered in the unit before the mid term and the final exam were very useful. Peer tutoring and pair programming where the most helpful methods (both indicated by 91% of the students) followed by the tutor (82%), discovery learning (73%) and asking questions (64%). the methods used in the parallel courses – laboratory programming and presentation of one student's solution – were considered very useful only by 45% of the students.

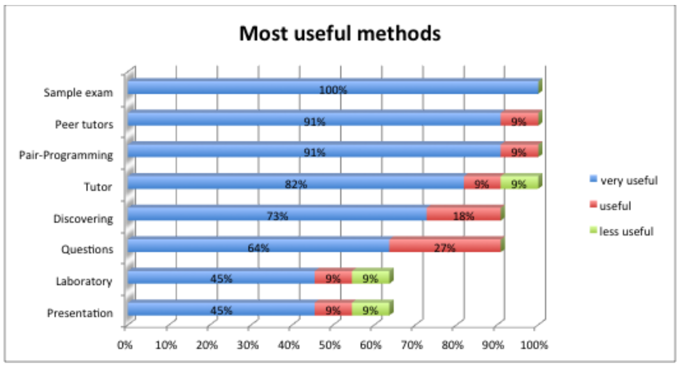

Fig. 6 Most useful methods, results of the final survey

In order to verify if brain-based methods and tasks can enhance the learning outcomes the results of two courses shall be compared: the practical courses as well as the corresponding lecture on "Introduction to structured and object-oriented programming". Unfortunately, the results of the lecture exams are still not available, hence we can only describe the outcomes of the practical courses.

As all students of the parallel practical courses got the same exams, a mid term and a final exam, that are evaluated using the Austrian grading system from 1 = excellent to 5 = failed. It has to be mentioned that there is still a teacher bias in this evaluation although all lecturers tried to assess the exams in the same way. The results of the lecture will be more relevant. The average of the grades in the experimental group was 2.19 whereas the average of the seven parallel courses was 2.94. Fig. 5 shows the average grades for all groups. The numbers in brackets indicate the final number of students in the respective courses.

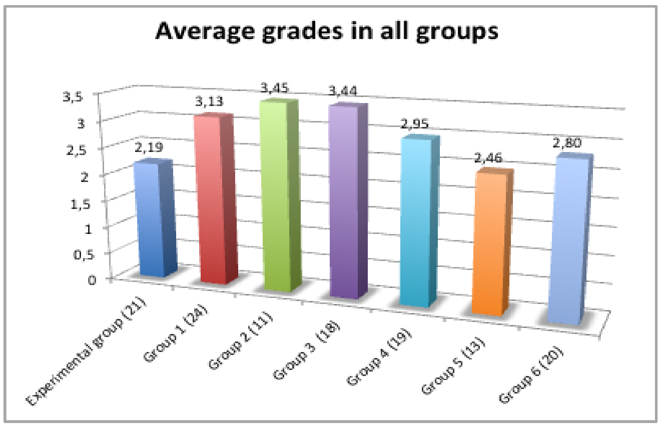

Fig. 7 Average grades in all parallel courses

This clearly shows a tendency of enhancement, although comparing the grading is, of course, only a week indicator. There are many factors involved in learning that can influence the outcomes. So the better score of the students in the experimental group could have different reasons; perhaps it is due to single methods, but the combination of different methods and exercise types. Maybe it is simply a consequence of higher motivation and more practice. This is to be verified in the larger follow-up study next year that will focus on the effects of cooperative learning (peer tutoring and pair programming) as well as on discovery learning (reading exercises and step-by-step instructions), which benefits from the functions of pattern recognition and rule extraction.

V. Conclusion and Outlook

The evaluation of the didactical experiment shows, that tasks and concepts based on neurodidactical principles are positively mentioned and preferred by the students. All in all the results of the pilot test "Brain-based Programming" are satisfying and show an overall improvement for the students as well as an above-average evaluation of the course. This supports the hypothesis that learning is more effective when the functioning of brain and memory is considered. As learning is a very complex process it cannot be said that the success of the pilot project is due to certain methods or the appropriate material. We suppose that the combination of both "brain-based" teaching material and "brain-based" methods, above all cooperative and discovery learning was the key to success. Should there really be single neurodidactical factors that lead to better learning, this has to be verified in further studies.

Therefore the concept of "Brain-based Programming" will be continued in a larger follow-up project "Teaching Informatics with the brain in mind". It is now extended to other courses of our department (Introduction to Computer Science and Software Engineering) and will be then adapted for testing and evaluation at schools, too.

References

[1] Moskal, B.; Lurie, D.; Cooper, S. 2004. Evaluating the effectiveness of a new instructional approach. In: *Proceedings of the 35th SIGCSE technical symposium on Computer science education* (SIGCSE '04). New York, NY: ACM, pp. 75-79.

[2] Villalobos, J. A.; Calderon, N. A.; Jiménez, C. H. 2009. Developing programming skills by using interactive learning objects. In: *ACM SIGCSE Bulletin*. Vol. 41 (3), pp. 151-155.

[3] Gomes, A., Paquete, L., Cardoso, A., Mendes, A. J. 2012. Increasing student commitment in introductory programming learning. In: *Proceedings of the 42th ASEE/IEEE Frontiers in Education Conference – FIE'12*, Seattle, USA, October 2012, pp. 82-87.

[4] Sousa, D. A. 2006. *How The Brain Learns*. Third edition. Thousand Oaks, California: Corwin Press.

[5] Jensen, E. 2008. *Brain-based learning: The new paradigm of teaching*. Thousand Oaks: Corwin Press.

[6] Sabitzer, B. 2011. Neurodidactics – A New Stimulus in ICT and Computer Science Education. In: L. Gómez Chova, I. Candel Torres, A. López Martinez (Hrsg.): *INTED 2011 Proceedings*. Barcelona: International Association of Technology, Education and Development (IATED), pp. 5881-5889.

[7] Sabitzer, B. 2011. Neurodidactics: Brain-based Ideas for ICT and Computer Science Education. *The Inter¬national Journal of Learning*, Champaign (IL): Common Ground Publishing vol. 18, pp. 167-177.

[8] Roth, G. 2011. *Bildung braucht Persönlichkeit: Wie Lernen gelingt*. Stuttgart: Klett-Kotta (German).

[9] Willis, J. 2008. Building a bridge from neuroscience to the classroom. *Phi Delta Kappan*, 896, pp. 424-427.

[10] Roth, G. Warum sind Lehren und Lernen so schwierig? In U. Herrmann (ed). 2009. *Neurodidaktik: Grundlagen und Vorschläge fu□r gehirngerechtes Lehren und Lernen*. Weinheim, Basel: Beltz (German).

[11] Brand, M., & Markowitsch, H. J. Lernen und Gedächtnis aus neurowissenschaftlicher Perspektive - Konsequenzen für die Gestaltung des Schulunterrichts. In U. Herrmann (ed). 2009, *Neurodidaktik: Grundlagen und Vorschläge für gehirngerechtes Lehren und Lernen* pp. 69-85. Weinheim, Basel: Beltz (German).

[12] Spitzer, M. 2003. *Lernen. Gehirn-Forschung und die Schule des Lebens*. Heidelberg, Berlin: Spektrum-Verlag (German).

[13] Hattie, J. 2008. Visible Learning. *A synthesis of over 800 meta-analyses relating to achievement*. London, New York: Routledge, Taylor & Francis Group.

[14] Caine, R.; Caine, G. *Overview of the System Principles of Natural Learning*. Available at: http://www.cainelearning.com/files/Summary.pdf. Accessed: 16 01, 2013.

[15] Mayer, R.E.; Moreno, R. 2003. Nine ways to reduce cognitive load in multimedia learning. Educational Psychologist, 38/1, pp. 43-52. Available at: http://cmapspublic2.ihmc.us/rid=1KXP7KR7M-8N27KG-1FNL/mayer_moreno_203.pdf . Accessed: 15 04, 2013