Contribution to Teaching Programming Based on "Object-First" Style at College of Polytechnics Jihlava

Marek Musil(✉) and Karel Richta

Department of Technical Studies, The College of Polytechnics Jihlava, Jihlava, Czech Republic
{marek.musil,karel.richta}@vspj.cz

Abstract. There are several different approaches to teaching programming, based on programming styles. A concept "first the object-oriented style, then the other one" known as "object-first" is currently being promoted by a number of technical colleges. The reasons originate from the practical area. Also the teaching of programming at the College of Polytechnics Jihlava (COPJ) is being switched from the "structural-then-object" style of programming to the "object-first" style. After the second run, the results achieved by students do not seem (very) good. This can be confirmed from the courses' feedback. It seems that the students are puzzled and their skills are poorer. Therefore, we decided to examine the results achieved by students and their opinion on the "object-first" style especially. This survey was carried out after the completion of the course with "object-first" teaching and at the beginning of the course with "structural" teaching. We are interested in skills in object-oriented programming and also in structural programming, but especially skills at the beginning of study at COPJ and the type of completed high school. We addressed our students attending the course of "structural programming". The third run of this teaching approach started. In this paper we introduce the first survey results. Even though the number of respondents is not big, the statistic results are significant within the College.

1 Introduction

There are several approaches to teaching programming, based on programming styles. The first choice is the established approach "first the structural style, then the object-oriented style" known as "structure-first" approach. The second offered teaching programming concept is the style "first the object-oriented style, then the other one" known as "object-first". This is currently being promoted for the programming teaching by a number of technical colleges.

The reasons mentioned come from the practical area. Current programming systems are typically developed in the object-oriented style. For this purpose, knowledge and skills mentioned hereafter are required. Students should understand the principle of objects including meaning of their properties and methods. In addition, they should be able to use existing objects (components) without profound structural principles knowledge (without profound knowledge of structural principles and structural programming). Deep structural programming is not required for these work positions. On

J. Janech et al. (eds.), *Proceedings of the 2015 Federated Conference on Software Development and Object Technologies*, Advances in Intelligent Systems and Computing 511,
DOI 10.1007/978-3-319-46535-7_17

the contrary, the opinion on the object-first is not unified. However, even the form of the course “object-fist” including exercises is in discussion.

We are interested in the student’s opinion on “object-first” especially. Our next goal is understanding programming skills following in relation to the programming ability and skills gained at high school. A technical high school or a technical study programme at high-school as well as respondent’s self- evaluation of programming skills are important for our intended inquiring.

2 Related Works

Empirical studies are presented in [2, 5, 7]. These reveal results for object-first and against structural programming as first. Bennedsen and Schulte [2] described three categories – using objects, creating classes and concepts. This option is justified in their article. The presented discussion on object-first leads to the new and innovative ways of teaching introductory programming. Johnson and Moses [7] performed an empirical study for the purpose of comparison of two students groups after a semester including programming teaching. While the first group studies objects and classes very early in the semester, the second group studies the basic programming structures. The exam result is clear. It indicates that students who take the object-first approach outperform those who take a structure-first approach [7]. It seems that the object-first approach outperforms the structure-first.

Briand et al. [3] proclaim a distinct lack of empirical evidence. They claim that object-oriented techniques are more popular as a result of opinion and anecdotal evidence only. Result of empirical evidence is omitted. A few of these not too convincing results are presented in their paper and mentioned below.

Firstly, Jones [6] confirms insufficiency of evidence. He identified several areas where a distinct lack of empirical evidence exists to support the assertions of gains in productivity and quality, reduction in defect potential and improvement in defect removal, and reuse of software components. [3] However, no empirical research proves the object-oriented development techniques as the option always providing the many benefits. Several are presented hereafter.

Basili et al. [1] introduces a positive result of their study of object-oriented techniques. This result provides significant benefits from reuse in terms of reduced defect density and rework as well as increased productivity [3]. For example, it could be shown in a development environment after introducing an object-oriented design method.

On the other hand, there are presented two negative results. Van Hillergersberg et al. [10] deliver the finding, that object-oriented concept is not easy to learn and use quickly. However, their evidence is not too convincing as regards the object-oriented design. Daly et al. [4] discovered the finding about inheritance depth and class hierarchies. Inheritance depth and conceptual entropy of class hierarchies can cause programmers difficulty maintaining object-oriented software.

Finally, Brian et al. [3] presents a controlled experiment performed with computer science students as subjects. They try to answer questions regarding the ease of understanding and modifying as two essential components of maintainability, including their impact on quality standards. Whereas object-oriented design documents win/triumph quality standards based on Coad and Yourdon principles, they are more sensitive to poor design practices.

Their results are as follows:

1. Maintainers with little experience don’t gain great benefit maintaining object-oriented designs over structured designs. (By using practical significance, statistical was not achieved.)
2. Adherence to ‘good’ object-oriented design principles will provide ease of understanding and modification for the resulting design when compared to an object-oriented design in which the principles have not been adhered to. (Used by statistical significance).
3. An object-oriented design which did not adhere to quality design principles is likely to cause more understanding and modification difficulties than an appropriate structured design. (The significant evidence).

Results mentioned above suggest the following. Switching developers proficient in structured techniques to object-oriented techniques may actually have negative effects on the design of the product until they become as proficient with the object-oriented techniques.

3 Materials and Methods

In conformity with re-accreditation of the study programs of Applied Computer Science and Computer Systems at the College of Polytechnics Jihlava (COPJ), which started from the academic year 2013/2014, the teaching of programming is being switched from the “structural, then object-oriented” style to the “object-first” style. The education concept, form of exercises and illustrated examples are presented in [8, 9]. The “object-first” course is marked as PRG1 - Programming 1 (the first semester) and the next course “structural programming” is marked as PRG2 - Programming 2 (the second semester). The second run has finished, and the moment for the evaluation of teaching using “object-first” style has come. The results achieved by students are not very convincing; their skills seem poorer. It seems that some students are puzzled in addressing the challenges during exercises. The reactions from further courses concerning programming may serve as feedback.

Our goal lies in investigation of programming ability and skills in students after finish PRG1 in different aspects. Especially, we would like to know the students’ opinion on the object-first approach. The responses should be categorized into groups by programming skills before starting at the College, by completing a technical high school, algorithmization and programming-teaching at high school, self-evaluation of programming skills at the begin of the study at COPJ, rating in the course PRG1, etc. Therefore, these questions should be in our intended questionnaire.

We prepared a questionnaire witch questions categorized into 3 sections. The first section contains questions oriented on the student and their programming knowledge and skills at the beginning of the study at the College including questions oriented on the completed high school and programming exercise there. The second section deals with object-oriented programming and the last section deals with structural programming. Based on that, we can carry out evaluation of some aspects mentioned above (such as completed technical high school, programming skills from the high school, self- evaluation of programming skills at the beginning of the study at COPJ, rating in the course of PRG1, etc.). The secondary goal is the level of programming knowledge before starting the study at COPJ and their changes after PRG1.

The questionnaire was anonymous and voluntary. We collected 37 questionnaire responses in total. This number is expected, because 37 students out of the 55 students registered in the course of PRG2 regularly visited exercises. All students filled in the questionnaire. We experienced a great success. The evaluation is presented in the next section.

4 Results and Discussion

4.1 Self-evaluation of Programming Skills at the Beginning of Study at COPJ

70.27% of students mention the completion of a technical high school. Roughly the same number of students (67.57%) acknowledge they have acquired programming ability and skills at the completed high school. Approximately half of the students confirm two aspects, algorithmization teaching (45.95%), and the programming teaching (56.76%) at high school. 62.16% of students mention a programming language that was used in the programming teaching at high school, namely the language C and C ++ at the most, then Pascal, Java, Visual Basic, but also JavaScript, PHP and CSS. Half of students admit, that they have gained programing skills by individual study.

The most students (72.9%) get rating worse than B in PRG1, rating C and D prevail. Every rating category is represented by 2 groups of students in ration 2:1 - students from the technical and non-technical high school, students with programming skills and without programming skills, etc. As for Q3a, Q4a, Q6a, Q7a and Q10a, the most frequent self- evaluating rating is 3, the average level of skills between the best and the worst. Only some students that have gained programming skills by individual study evaluate themselves by rating 1. However, some students get rating A in PRG1 (Fig. 1).

The questions for the chart 1 are as follows:

Q3 – Do you have programming ability, skills and knowledge from a technical high school?
Q4 – Did you study at a technical high school?
Q5 – Did you graduate from a course of ICT at high school?
Q6 – Did you learn algorithmization at high school?
Q7 – Did you learn programming at high school?
Q10 – Did you gain programming skills by yourself?

The questions for the chart 2 are as follows:

Q2 – What is your rating in PRG1?
Q9 – Self-evaluate your programming skills at the beginning of study at VSPJ.

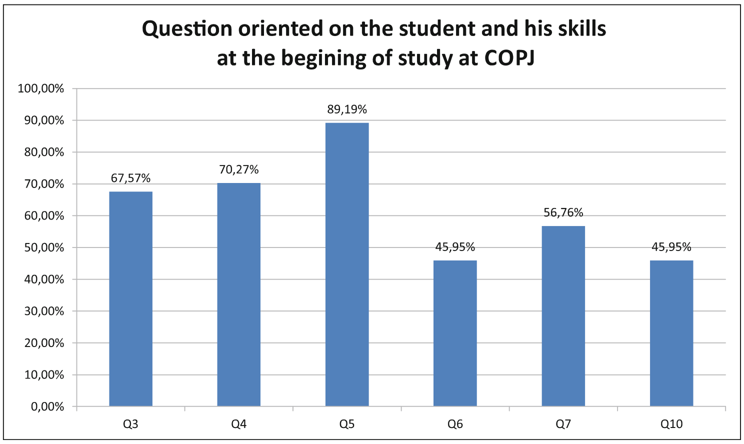

Fig. 1. Question oriented on the student and his skills in begin study at COPJ.

4.2 The Opinion on "Object-First"

As regards the opinion on "object-first", the most students (70.27%) prefer structural programming as first. Object-first is preferred only by some students having programming skills from the high school (13.51% of all students). 8% students have not decided, 8% students don't know. It includes the students having programming skills from the high school. The students that have no programming skills acquired at a high-school are definitely for "structure-first", several students of them are undecided.

While the most of students hold fast to the "structural-fist", only several students hold fast to the "object-first". While several students having programming skill from the high school don't know what to prefer, several students having no programming skills from the high school are undecided.

4.3 Ability and Skills of Object-Oriented Programming or Structural Programming

Object-oriented knowledge is not very clear in all students' categories. Nearly half of the responses (approximately 30%) are false. The responses to structural programming questions turn out a little better (Figs. 2, 3, 4, 5 and 6).

The questions for the chart 5 are as follows:

Q12 – What is a special method that is used for object creating?
Q13 – What is the name of a special method that is used when the object has been destroyed?
Q14 – Is their explicit definition necessary?
Q15 – Inheritance. What does it mean when object B is inherited from object A?
Q16 – Object B is inherited from object A. What are the names of these objects?
Q17 – Object B is inherited from object A. Who is the direct predecessor and the direct follower?

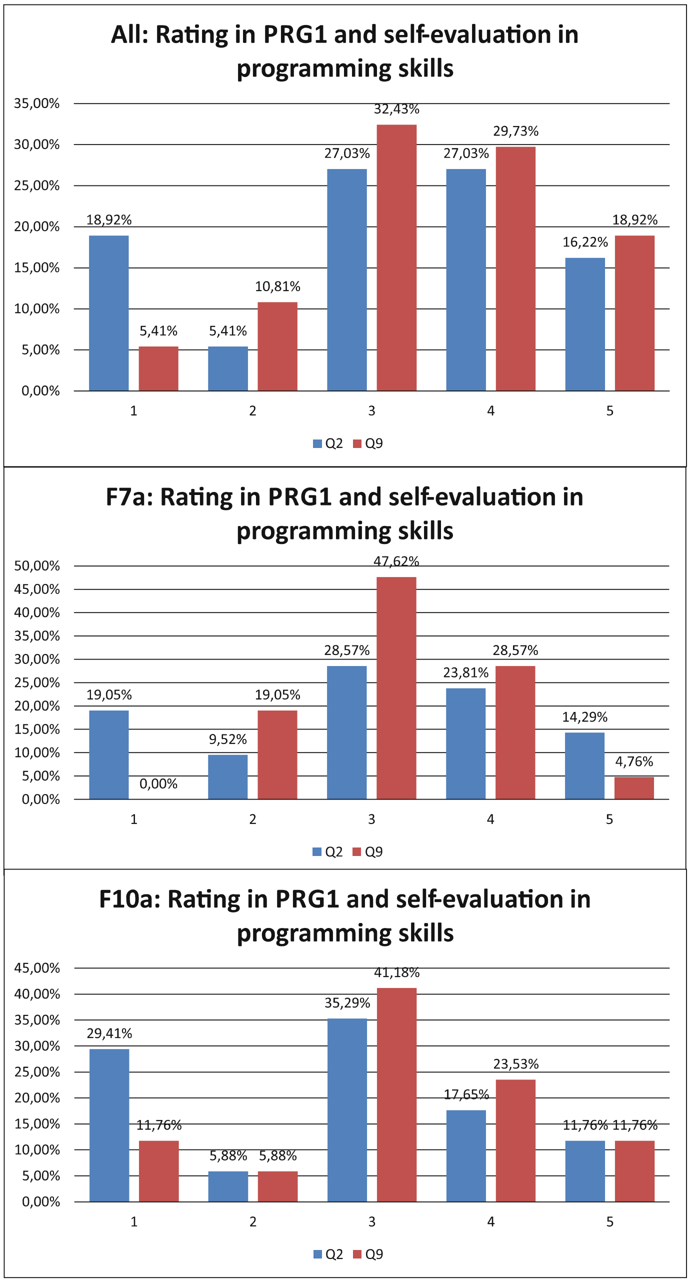

Fig. 2. Rating gained in PRG1 and self-evaluation filtered by some criterion. All - no filter (all respondents); F7a – title of programming in a course in the high-school; F10a – self-study of programming.

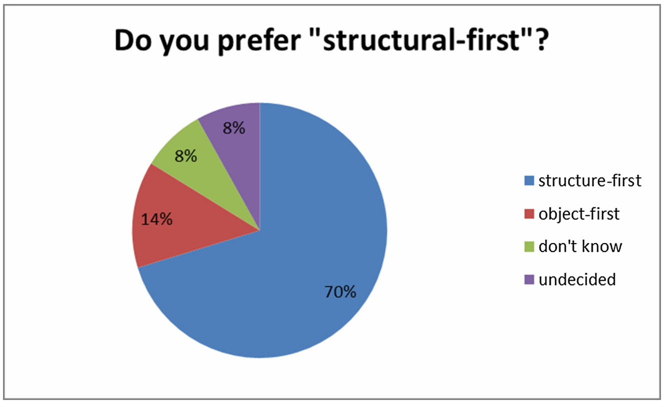

Fig. 3. The opinion on “object first”.

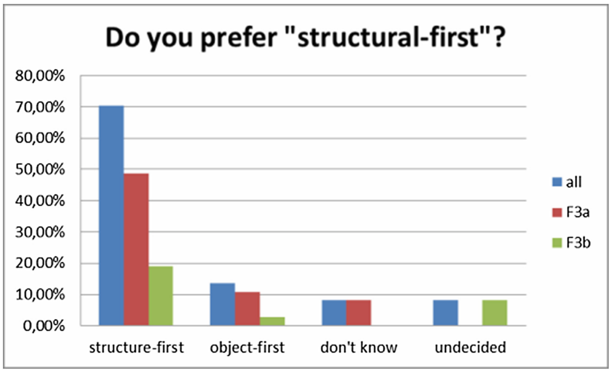

Fig. 4. The opinion on “object first” filtered by some criterion: all – no filter (all respondents); F3a – programming ability and –skills from the high-school; F3b – no programming ability or –skills from the high-school.

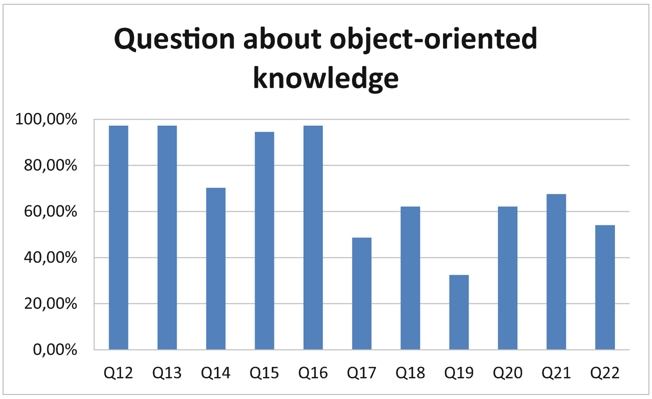

Fig. 5. Question about object-oriented knowledge.

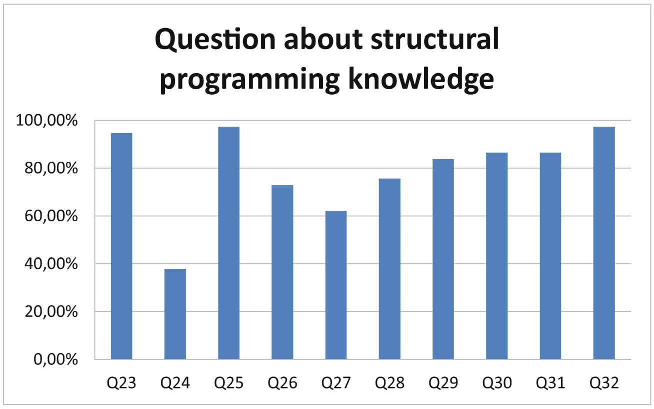

Fig. 6. Question about structural programming knowledge.

Q18 – What is the difference between "class" and "object"? Q19 – What does "state of an object" mean?
Q20 – An object owns a static variable/property. There are more objects of the same type. How many variables exist?
Q21 – What does "public interface of a class" mean?
Q22 – What does "a private item of an object" mean?

The questions for the chart 6 are as follows:

Q23 – There are commands. What is the value of x?

```
int x, y;
x = 5;
y = 4;
x = 2;
```

Q24 – There are commands. What is the value of x?

```
int x, *y;
x = 5;
y = &x;
*y = 2;
```

Q25 – There are commands. What is the value of x?

```
int x = 5, y;
if (x > 4)
        y = 4;
else
        y = -4;
```

Q26 – Is the meaning of commands sub-parts equivalent?

```
int x, y, z;

if (x > 5)
        if (y < 4)
            z = 1;
        else
            z = 2;
```

```
int x, y, z;

if (x > 5)
{
        if (y < 4)
                z = 1;
}
else
        z = 2;
```

Q27 – Thera are commands. What is the value of z?

```
int x, y, z;

if (x > y)
        z = x;
else
        z = y;
```

Q28 – Write commands for two values changing.

Q29 – There are commands. What is written in the standard output?

```
void tisk(int x)
{
        printf("%d", x);
}

int _tmain(int argc, _TCHAR* argv[])
{
        for (int i = 0; i < 3; i++)
                tisk(i);

        return 0;
}
```

Q30 – How many times is the cycle performed?

```
for (int i = 0; i < 3; i++)
        ;
```

Q31 – How many times is the cycle performed?

```
for (int i = 0; i <= 3; i++)
        ;
```

Q32 – How many times is the cycle performed?

```
for (int i = 3; i < 3; i++)
        ;
```

5 Conclusion

The paper describes our experience with the teaching of "object first" style. The result of the questionnaire survey was presented. We discussed students' opinion on "object first". The responses showed, that the students take a clearly negative stand to "object-first". They think that structural programming first is better. The students' categorization plays no significant role in terms of programming skills or knowledge and in terms of rating of PRG1.

Only individual study of programming added to a better or good self-evaluation. Probably, these students are sure of the programming.

Although the response number is small, the questionnaire brought up interesting questions. We would like to carry out the questionnaire survey at the begin of study at the College, then after the course of PRG1, and after the course of PRG2. This will provide the opportunity for a larger scale comparison and evaluation.

Acknowledgment. This paper was partially supported by the grant “Inovace předmětů Programování 1 a 2” of COPJ and also by the Avast Foundation.

References

1. Basili, V.R., Briand, L.C., Melo, W.L.: How reuse influences productivity in object-oriented systems. Commun. ACM **39**(10), 104–116 (1996). doi:10.1145/236156.236184. Cited 3 Oct 2015
2. Bennedsen, J., Schulte, C.: What does “objects-first” mean?: an international study of teachers’ perceptions of objects-first. In: Proceedings of the Seventh Baltic Sea Conference on Computing Education Research. Koli Calling, Finland (2007). http://crpit.com/abstracts/CRPITV88Bennedsen.html
3. Briand, L.C., Bunse, C., Daly, J.W., Differding, C.: Emp. Softw. Eng. **2**(3), 291–312 (1997). doi:10.1023/a:1009720117601. Cited 3 Oct 2015

. Daly, J., Brooks, A., Miller, J., Roper, M., Wood, M.: Evaluating inheritance depth on the maintainability of object-oriented software. Emp. Softw. Eng. **1**(2), 109–132 (1996). doi:10.1007/bf00368701. Cited 3 Oct 2015

5. Ehlert, A., Schulte, C.: Empirical comparison of objects-first and objects-later. In: Proceedings of the Fifth International Workshop on Computing Education Research Workshop, ICER 2009 (2009). doi:10.1145/1584322.1584326. Cited 28 Aug 2014
6. Jones, C.: Gaps in the object-oriented paradigm. IEEE Comput. **27**(6), 90–91 (1994). doi:10.1109/MC.1994.10064
7. Johnson, R.A., Moses, D.R.: Objects-first vs. structures-first approaches to OO programming education: an empirical study. In: Proceedings of the Allied Academies 2008, vyd. Reno, USA, pp. 244–248 (2008). http://www.researchgate.net/publication/242549890_objects-first_vs._structuresfirst_approaches_to_oo_programming_education_an_empirical_study. Cited 28 Aug 2014
8. Musil, M., Richta, K.: Approaches to teaching programming in the “Objects-first” style. Logos Polytechnikos **5**(4), 114–121 (2014). http://www.vspj.cz/soubory/download/id/3829. ISSN 1804-3682
9. Musil, M.: Přístupy k výuce programování “object-first”. In: Informatika XXVII/2014: Sborník abstraktů z mezinárodní odborné pedagogicky zaměřené konference. MUSIL, Marek. 1 vydání, pp. 1–2. Ústav informatiky, Mendelova univerzita v Brně, Brno (2014). ISBN 978-80-7509-126-0
10. Van Hillegersberg, J., Kumar, K., Welke, R.: An empirical analysis of the performance and strategies of programmers new to object-oriented techniques. In: Psychology of Programming Interest Group: 7th Workshop (1995)