Asesorías y Tutorías para la Investigación Científica en la Educación Puig-Salabarría S.C.
José María Pino Suárez 400-2 esq a Lerdo de Tejada. Toluca, Estado de México. 7223898475
RFC: ATI120618V12

Revista Dilemas Contemporáneos: Educación, Política y Valores.
http://www.dilemascontemporaneoseducacionpoliticayvalores.com/
Año: VI Número: Edición Especial Artículo no.:12 Período: Marzo, 2019.

TÍTULO: Modelado de estructuras matemáticas y programación orientada a objetos.

AUTORES:

1. T.Yu Gainutdinova.
2. M.Yu Denisova.
3. L.V. Riazanova.
4. Z.F. Shakirova.
5. O.A. Shirokova.

RESUMEN: El artículo contiene los problemas de la enseñanza de una programación orientada a objetos para los estudiantes que estudian en la Facultad de Matemáticas con la especialidad en pedagogía. La programación orientada a objetos permite aprovechar el enfoque orientado a objetos no solo en las fases de diseño y desarrollo de los sistemas de software, sino también en las etapas de su implementación, prueba y mantenimiento. El diseño de proyectos orientados a objetos de modelado de sistemas y estructuras contribuye a la formación de las habilidades de los estudiantes para formalizar la tarea, destacando las abstracciones y objetos del dominio del tema, estructurándolos e implementándolos.

PALABRAS CLAVES: programación orientada a objetos, diseño orientado a objetos, componentes visuales de desarrollo de software, clases, objetos.

TITLE: Modelling mathematical structures and object-oriented programming.

AUTHORS:

1. T. Yu Gainutdinova.

2. M. Yu Denisova.

3. L.V. Riazanova.

4. Z.F. Shakirova.

5. O.A. Shirokova.

ABSTRACT: The article contains the problems of teaching an object-oriented programming for students studying at the Mathematics Faculty with the pedagogy major. Object-oriented programming allows to take an advantage of the object-oriented approach not only at design and developing phases of software systems, but also at the stages of their implementation, testing and maintenance. The object-oriented projects design of modeling systems and structures contributes to the formation of students' skills in formalizing the task, highlighting the abstractions and objects of the subject domain, structuring and implementing them.

KEY WORDS: object-oriented programming, object-oriented design, software development visual components, classes, objects.

INTRODUCTION.

Object-oriented programming (OOP) is the basic part of the curricula for students studying at the Mathematics Faculty with Pedagogy major (Barkov, 2009; Gainutdinova and Shirokova, 2016; Shirokova, 2015).

The use of the object-oriented approach creates students' object thinking. However, here are arisen a number of problems (Pavlovskaya, 2003). These problems are associated with the complexity of the studied subject area, inability to identify the necessary classes and objects in it, their

connections and structures (Shirokova, 2015). There are also the following methodological problems:

– Training of software design in school and in the first year of the university is based on the principles of structural programming, which uses functional decomposition of tasks and does not include an object-oriented approach. These types of training develop a stereotype of procedural thinking. A complete reorganization of system thinking is necessary while using object decomposition of tasks. The best methodical solution of this problem is to study object-oriented programming in parallel with studying the structural at the very beginning of the learning process: both in school and in the first year of the university.

At the present stage, object-oriented programming (Booch, 2008; Booch, 2007; Yuvarajan, et. al. 2018) is focused on the designing of complex programs; therefore, the developers of software tools, as a rule, publish the implementation of their software in the form of a «black box». Detailed description of the stages of object-oriented analysis, design, modeling and developing of this project is hidden, user have only the external logic of the software. Therefore, there are no good examples of implementing object-oriented software with explanations on the choice of the solutions.

In relation to the above mentioned, methodical development of examples of the implementation of object-oriented projects (Barkov, 2009; Gainutdinova and Shirokova, 2016; Shirokova, 2015; Vikentieva and Polyakova, 2012) in various programming systems by teachers is necessary and important, as this is the basis for the teacher to transfer his experience to students.

DEVELOPMENT.

Materials and methods of the research.

Problem Statement.

The main building blocks of object-oriented methodology of program analysis and design are classes and objects (3,10). Classes are abstractions of reality.

During the object-oriented programming course at the Faculty of Mathematics, the main methodological way of practical classes is to inculcate the basic skills of developing object-oriented projects. It should be noted that the basis of these projects should be the classes and objects, the prototypes of which are mathematical structures. The course provides students with creating projects related, for example, to the implementation of classes of mathematical abstractions and structures.

The article proposes the development of projects for implementing the class of a mathematical object «complex number» in various systems of object-oriented programming: C ++ (Booch, 2007; Pavlovskaya, 2003; Pobegailo, 2006; Yuvarajan, et al. 2018), Delphi (Darakhvelidze and Markov, 2005; Gainutdinova and Shirokova, 2016; Shirokova, 2015), Python (Beazley, 2016; Cox and Raspberry, 2014; Gurikov, 2017; Suzie, 2015; Tosi, 2009; Vikentieva and Polyakova, 2012).

The developing of the projects considers the following issues of object-oriented analysis:

- indication of the complexity of the problem
- object-oriented decomposition of the subject area;
- object-oriented analysis and object-oriented design;
- use of the universal modeling language UML for object-oriented modeling;
- implementation of object-oriented technology in a specific programming system;
- implementation of the structure of the object superstructure of the programming system, using libraries of visual components;
- implementation of the most important classes and their interaction with the operating system.

Methodology.

The developing of the projects contains the concept of the life cycle of a software product, inclosing the processes, actions and tasks that must be performed when the projects are created (Ivanova, 2011). In the process of object-oriented analysis an enormous amount of attention is given to the

defining and describing objects in terms of the subject domain. It is proposed to use the UML system modeling language to develop an object-oriented domain model (Quatrani, 2002). For this purpose, analysis and design models are used using the basic types of diagrams. Throughout the process of object-oriented design logical program objects are defined as a future-implemented in a specific programming system. The resulting program objects are classes and these objects include attributes, methods, and properties. An important point in the design is a separate description of the structure and implementation of classes.

The main task in the development of an object-oriented project is to code using a library of visual components. Moreover, the student must know the hierarchy of components that describes their interaction; the environment of the program in which the components work; interaction of the program with the operating system.

The development of projects for implementing a class of complex numbers in various object-oriented environments C ++, Delphi, Python give a teacher an opportunity to solve a number of methodological tasks:

- conducting object-oriented decomposition of the subject area;
- learning the UML language to build models of analysis and design using the main types of diagrams;
- identifying logical program objects that will be implemented in various programming systems;
- analyzing the programming systems, implementing the complex number class and program structure, using the features of the specific environment;
- creating object-oriented projects using the features of libraries of visual components of specific environments.

An important methodological aspect is that each next project in the new software system is based on the knowledge gained in the creation of previous projects. Development of object-oriented models using the UML language, project module codes, using of right rules of naming of identifiers and other methodical approaches significantly improves continuity during moving to the creation of a project in a new software environment. The learner during moving to a new project is often forced to implement changes in existing programs, that's why the awareness of the need to take care of the modifiability of the program is quickly instilled.

Practical lessons of the course of object-oriented programming are aimed at implementing the stages of analysis, design, modeling and programming of a given project. Now we should move to the following project: create a class of complex numbers whose fields are the real and imaginary parts of the number; class methods perform actions on complex numbers. Implement the capabilities of the class to work with the object, calculating with a given accuracy, using the Taylor series. Develop a project in object-oriented environments: C ++, Delphi, Python.

Parts of the main modules describing the structure of the class and its methods in the C ++ system are showing here.

The structure of the class is described in the header file Complex.h

```
ref class Complex,
private:
        Double x, y;
public:
        Complex ();
        Complex (Double x, Double y);
        void Sum (Complex ^A, Complex ^B);
        void Sub (Complex ^A, Complex ^B);
```

```
void Multi (Complex ^A, Complex ^B);
void Div (Complex ^A, Complex ^B);
void TgA (Double eps);
void TgB (Double eps);
event Result^ ChangeValueInTable;
```

Descriptions of class methods are contained in the implementation file Complex.cpp (here's a part of the Sum method description):

```
void Complex: Sum (Complex ^A, Complex ^B)
    this->x = A->x + B->x;
    this->y = A->y + B->y;
    ChangeValueInTable(x, y);
```

The window of the visual project in C ++:

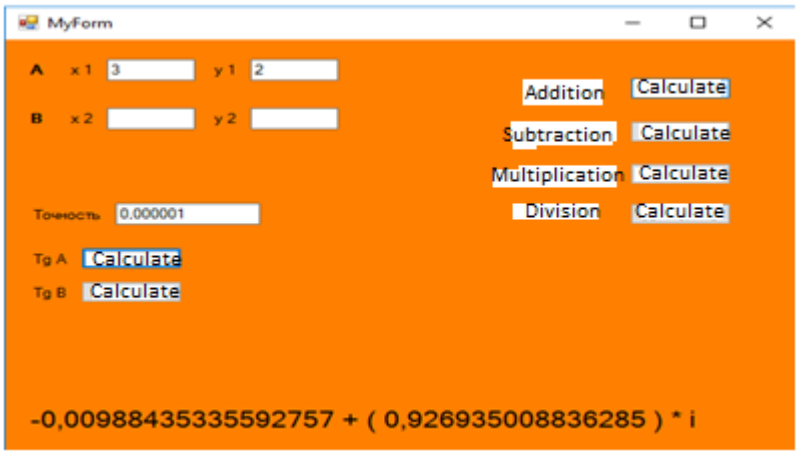

Figure 1. Visual project window.

Parts of the module describing the structure of the class and its methods in Delphi are showing here.

```
Type Complex=class
re,im:real;
procedure Impre (x:real);
```

```
procedure Impim (y:real);
function Outre:real;
function Outim:real;
procedure Sum (z:Complex);
procedure Sub (z:Complex);
procedure Multi (z:Complex);
procedure Div (z:Complex);
procedure TgA;
procedure TgB;
end;
```

Description of the Sum method in Delphi:

```
procedure Complex.Sum;
begin
  re:=re+z.re;
  im:=im+z.im;
end;
```

Parts of the module describing the structure of the class and the Sum method in Python are showing here:

```
class TComplex=
def SetReal(self,x):
  self.Real=x
def GetReal(self):
  return self.Real
def SetIm(self,y):
```

```
  self.Im=x
def GetIm(self):
  return self.Im
……..
def Sum(self,z):
  self.Real +=z.Real
  self.Im +=z.Im
……
```

The use of rules for naming identifiers and similar methodical methods significantly increases continuity. Easy understanding of the program is an important practical principle. Using the naming rules of identifiers incorporated in the programming system leads to the fact that as a result of using standard names (for example, buttons "Button", labels "Label", etc.), the assignment of these components in this program can be determined only after its careful investigation. Therefore, identifiers that denote the names of components, methods, fields, etc., must carry a meaning.

High-quality program should not only solve an applied problem, but be easily read by any user, because in practice, often a program developed by one programmer is passed on to another specialist to maintenance and develop it. Thus, the object-oriented style of programming makes it possible to take advantage of the object-oriented approach not only at the stages of designing and constructing software systems, but also at the stages of their implementation, testing and maintenance.

Results.

The task of teaching students to think objectively is complex. Object-oriented methodology is based on the concepts of classes and objects. Classes form the structure of data and actions and use this information for the construction of objects. From one class can be built more than one object at the

same time, each of them will be independent of the others. The creation of projects related to the implementation of classes of mathematical abstractions forms object thinking at students of the mathematical faculty.

The development of object-oriented projects includes the environment and the design object (the problems of creating software systems, the features of a complex system, the project model, the life cycle of the software product), the design of software products (the meaning of design, the importance of model building, elements of software design, object-oriented analysis, design and programming), the principles and tools of object-oriented analysis (classification and object-oriented programming, the difficulties of classification, object-oriented analysis), the principles and tools of object-oriented designing (abstraction, encapsulation, modularity, hierarchy, typification), the advantages of the object model.

The presented methodical approach of the implementation of classes of mathematical abstractions and structures allows: to conduct object-oriented decomposition of the subject area (mathematical object - complex number); to learn the UML language for building a project model; to define logical program objects that will be implemented in various programming systems (C ++, Delphi, Python); to analyze the systems of programming, to realize the class of complex numbers and the structure of the program, using the features of the concrete environment; create visual projects.

CONCLUSIONS.

The peculiarities of studying the course of object-oriented programming technology are considered in the article. Object-oriented programming is one of the most effective approaches to modern programming. The procedural programming principle, which was used by programmers earlier, has become obsolete over time, causing many problems for developers. These problems are solved by object-oriented programming, which allows you to combine data and methods related to one entity,

and work with them as one. Therefore, object-oriented programming and object-oriented programming technology have already become an integral part of university program.

Object-oriented programming, being a largely developed discipline, includes both theoretical justifications and a rich set of professional practices and recommendations (Barkov, 2009). To improve the quality of teaching the "Programming" course, great attention must be paid to the creation of object-oriented projects of mathematical structures in various environments.

The creation of object-oriented projects of modeling systems and structures contributes to the formation at students of skills of formalization of the task, highlighting the abstractions and objects of the subject area, structuring and implementing them. Students develop the skills to apply software development models when creating software products, and also apply software modeling tools.

Acknowledgements.

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

BIBLIOGRAPHIC REFERENCES.

1. Barkov I.A. (2009). Teaching the discipline "Object-Oriented Programming" //Educational Technologies and Society, pp. 494-516.
2. Beazley, D. (2016). Python Programming Language Publisher: Addison-Wesley Professional. – ISBN: 9780134217314
3. Booch, G. (2007). Object-Oriented Analysis and Design with Applications in C++. Addison-Wesley. – ISBN 0-201-89551-X
4. Booch, G. (2008). Object-Oriented Analysis and Design with Applications (3rd Edition). Trans. with English – M.: «Williams». – 720 p. – ISBN 978-5-8459-1401-9

5. Cox, T. Raspberry Pi. (2014). Cookbook for Python Programmers. – Birmingham: Packt Publishing. – 402 p.
6. Darakhvelidze, P.G., Markov E.P. (2005). Programming in Delphi 7.– St. Petersburg: BHV – Petersburg, – 784 p.
7. Gainutdinova, T.Yu., Shirokova, O.A. (2016). Features of professional training in programming the future teacher of computer science. Program and theses of the II International Forum on Teacher Education (MFSP-2016). – Kazan: Kazan University. – pp. 231-232.
8. Gurikov, S.R. (2017). Fundamentals of Algorithmization and Programming in Python: Textbook. FORUM: INFRA-M. – 343 p.
9. Ivanova G.S. (2011). Programming technology: a textbook. /G.S. Ivanova – M., Knorus. – 336 p.
10. Pavlovskaya, T.A. (2003). C/C++. Structural programming: Workshop /T.A. Pavlovskaya, Yu.A. Shchupak – SPb: Piter– 240 p.
11. Pobegailo, A.P. (2006). C / C ++ for the student: Textbook /А.П. Pobegailo – St. Petersburg: BHV–Petersburg. – 528 p. – ISBN 5-94157-647-1
12. Quatrani, T. (2002). Visual Modeling with Rational Rose 2002 and UML (3rd Edition). – Boston– 134 p. – ISBN: 0201729326
13. Shirokova, O.A. (2015). Object-oriented programming with creation of classes for objects of type «massive» and «matrix» // Object systems – 2015 (Winter session): Proceedings of XI International Theoretical and Practical Conference (Rostov-on-Don, 10-12 December, 2015) / Edited by Pavel P. Oleynik. – Russia, Rostov-on-Don: SI (b) SRSPU (NPI). – pp. 15-23.
14. Suzie, R.A. (2015). Python: Manual, R.A. Susie. – St. Petersburg: BHV-Petersburg, 2015. – 759 p. – ISBN 978-5-9775-1417-0

15. Tosi, S. (2009). Matplotlib for Python Developers. – Birmingham: Packt Publishing.

16. Vikentieva, O.L., Polyakova O.A. (2012). Project-programming and C++ programming: object-oriented programming. O.L. Vikentieva, O.A. Polyakova. International Journal of Applied and Fundamental Research, No.6. – pp. 56-57.

17. Yuvarajan, D., Babu, M. D., BeemKumar, N., & Kishore, P. A. (2018). Experimental investigation on the influence of titanium dioxide nanofluid on emission pattern of biodiesel in a diesel engine. Atmospheric Pollution Research, 9(1), 47-52.

DATA OF THE AUTHORS.

1. T.Yu. Gainutdinova. Kazan Federal University.

2. M.Yu. Denisova. Kazan Federal University.

3. L.V. Riazanova. Kazan Federal University.

4. Z.F. Shakirova. Kazan Federal University.

5. O.A. Shirokova. Kazan Federal University.

RECIBIDO: 2 de febrero del 2019. **APROBADO**: 13 de febrero del 2019.