# STUDYING OBJECT-ORIENTED PROGRAMMING IN SCHOOL

**Pasichnyk T.,**
*candidate of physical and mathematical sciences,*
*professor of the programming department*
*of Ivan Franko National University of Lviv*
*79000, Ukraine, Lviv, st. University, 1*
**Solyar T.**
*candidate of physical and mathematical sciences,*
*senior researcher associate,*
*professor of the programming department*
*of Ivan Franko National University of Lviv*
*79000, Ukraine, Lviv, str. University, 1*

# ВИВЧЕННЯ ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ В ШКОЛІ

**Пасічник Т.**
*кандидат фізико-математичних наук*
*доцент кафедри програмування*
*Львівського національного університету*
*імені Івана Франка*
*79000, Україна, Львів, вул. Університетська,1*
**Соляр Т.**
*кандидат фізико-математичних наук,*
*старший науковий співробітник*
*доцент кафедри програмування*
*Львівського національного університету*
*імені Івана Франка*
*79000, Україна, Львів, вул. Університетська,1*
https://doi.org/10.5281/zenodo.10591908

**Abstract**

The article discusses the problems of studying object-oriented programming (OOP) in educational institutions. The main aspects that require teachers' attention are identified, including the choice of programming language, the order in which OOP concepts are taught, the necessary learning environment, and the inclusion of modeling and design elements. The problems of choosing a programming language for learning OOP are pointed out, providing a comparison between C++, Java and Python. Caution is expressed against using C++ due to its complexity, and a recommendation is made to use Java as a purer object-oriented language. The article also raises the issue of methods of teaching OOP, indicating the need to emphasize abstract thinking and design already at the initial stage. A recommendation is given for the use of graphical tools to facilitate the understanding of OOP concepts. The issue of choosing a programming environment for an introductory course in OOP is considered. It is noted that integrated graphic systems can be too complex for initial training, which is why the use of special environments, such as the BLUE system, which simplifies the process of creating methods and objects, is proposed. The importance of building an effective approach to learning OOP, which takes into account the choice of language, the order of teaching, the environment and the use of graphic tools, is emphasized.

**Анотація**

У статті розглянуто проблеми вивчення об'єктно-орієнтованого програмування (ООП) в середніх навчальних закладах. Визначено основні аспекти, які потребують уваги вчителів, включаючи вибір мови програмування, порядок викладання концепцій ООП, необхідне середовище навчання та включення елементів моделювання та дизайну. Вказано на проблеми вибору мови програмування для навчання ООП, надаючи порівняння між C++, Java та Python. Висловлено застереження щодо використання C++ через його складність, а також надано рекомендацію щодо використання Java як більш чистої об'єктно-орієнтованої мови. Стаття також порушує питання методів викладання ООП, вказуючи на необхідність акцентування уваги на абстрактному мисленні та дизайні вже на початковому етапі. Надано рекомендацію щодо використання графічних засобів для полегшення розуміння концепцій ООП. Розглянуто питання вибору середовища програмування для вступного курсу з ООП. Зазначено, що інтегровані графічні системи можуть бути занадто складними для початкового навчання через що запропоновано використання спеціальних середовищ, таких як система BLUE, яка спрощує процес створення методів та об'єктів. Підкреслено важливість побудови ефективного підходу до вивчення ООП, який враховує вибір мови, порядок викладання, середовище та використання графічних інструментів.

**Keywords:** Object-oriented programming, school, teacher, C++, Java, Python, educational programs.
**Ключові слова:** Об'єктно-орієнтоване програмування, школа, учитель, C++, Java, Python, навчальні програми.

На початку 20-х рр. XXI ст. об'єктно-орієнтоване програмування (ООП) стало домінуючим стилем програмування. Методика вивчення ООП у навчальних закладах різного рівня висвітлена в багатьох працях вітчизняних науковців [1-3, 6, 7]. Проте вчителі, зазвичай, відчувають проблеми під час ознайомлення початківців з об'єктно-орієнтованою концепцією та програмуванням і тому розробка ефективних підходів до навчання основ ООП в школі та на початкових курсах у вищих навчальних закладах все ще залишається актуальною серед науковців [4, 5]. Педагоги в цій галузі продовжують експериментувати із різними стилями, методами та підходами до навчання.

Насамперед відзначимо, що ООП – це парадигма програмування, яка використовує концепцію «об'єктів», щоб організувати код. Основна ідеєя ООП – моделювання програмного коду як сукупності взаємодіючих об'єктів, які представляють екземпляри класів.

Основні поняття ООП включають: (а) к**ласи –** шаблон або визначення для створення об'єктів. Він визначає властивості та методи, які об'єкти цього класу можуть мати; (б) о**б'єкти –** конкретний екземпляр класу (представляє собою конкретний екземпляр даних та функціоналу, описаного в класі; (в) і**нкапсуляцію –** сприяє об'єднанню даних та методів, що працюють з цими даними, в один об'єкт, що дозволяє приховувати деталі реалізації від зовнішнього світу та сприяє зменшенню залежностей між різними частинами програми; (г) н**аслідування –** дозволяє створювати нові класи на основі вже існуючих (підклас може успадковувати властивості та методи батьківського класу, а також додавати або перевизначати свої власні); (д) п**оліморфізм –** дозволяє використовувати об'єкти різних класів з однаковим інтерфейсом без необхідності зазначення їх конкретного типу (сприяє виразній та гнучкій роботі з об'єктами) [10].

Таким чином, ООП робить код більш читабельним, облегшує його розуміння та підтримку. Багато сучасних програмних мов, таких як Java, C++, Python, використовують принципи ООП для створення програм.

Вчителі, які вперше ознайомлюють початківців з ООП, можуть стикатися з низкою проблем, найпоширенішими із яких стали:

1. **Складність абстракції:** ООП використовує концепції, такі як класи, об'єкти, наслідування, інкапсуляція та поліморфізм, які можуть бути для новачків важкими для розуміння. Вчителі можуть зіткнутися з труднощами у поясненні цих абстракцій.

**2. Зміна мислення:** ООП вимагає від програмістів зміни підходу до розв'язання завдань. Деякі вчителі можуть стикається з труднощами в навчанні учнів при переході від процедурного мислення до об'єктно-орієнтованого.

3. **Відсутність практичного досвіду:** для повного розуміння ООП важливо мати практичний досвід. Вчителі можуть стикнутися із проблемою недостатньої кількості практичних завдань та вправ для закріплення отриманого матеріалу.

4. **Недостатність ресурсів:** У навчанні ООП може бути обмежена кількість доступних навчальних ресурсів та прикладів, що може ускладнити розуміння концепцій.

5. **Велика кількість коду:** в осягненні розуміння ООП важливу роль відіграє практика, але велика кількість коду при вивченні може стати викликом для новачків, і вчителі повинні знайти баланс між теоретичними поясненнями та практичним використанням [9].

Вирішення вищезазначених проблем може включати в себе збалансований підхід, використання конкретних прикладів, практичних завдань та можливо використання додаткових навчальних ресурсів для підтримки процесу вивчення ООП [10].

Як видається, оптимальною є наступна схема навчання програмуванню, яка до того ж без особливих проблем вписується у навчальну програму з інформатики, затверджену МОН:

– Scratch-програмування (2-4 класи);

– основи структурного (алгоритмічного) програмування (5 клас);

– ознайомлення з об'єктним програмуванням (6 клас);

– ознайомлення із програмуванням на Android (7 клас);

– основи об'єктного програмування (8 клас);

– основи функціонального програмування (9 клас) [9].

Важливо також розглянути декілька важливих питань, які вимагають вирішення у контексті досліджуваних питань. Перше із них – *яку об'єктно-орієнтовану мову слід вивчати?*

Більшість курсів програмування у середніх школах та вищих навчальних закладах фокусуються на популярних об'єктно-орієнтованих мовах програмування, таких як C++, Java та Python. Остання, зокрема, стала популярним вибором для навчання програмуванню в школах через простоту та читабельність коду. Java та C++ використовуються для введення учнів у більш сучасні концепції програмування та комп'ютерних наук. Важливо зауважити, що програми навчання можуть змінюватися в залежності від регіону, школи або навчального закладу. Наприклад, у деяких програмах можуть також використовуватися інші мови програмування, такі як JavaScript, C#, або навіть більш високорівневі мови, такі як Scratch, для навчання основам програмування [10].

Все ж, значна кількість курсів програмування у середніх школах спрямовані на такі об'єктно-орієнтовані мови як C++ та Java. Хоча засновники мови Python, передусім, орієнтувались на спрощення вивчення програмування, однак такі аспекти

мови як динамічна типізація, відкритість об'єктів та множинне успадкування не сприяють використанню цієї мови як початкової при вивченні ООП [10].

У навчальних програмах з інформатики в переважній більшості середніх шкіл перевагу віддають мові С++ через її потужність та відносну простоту реалізації класичних алгоритмів і структур даних. Однак С++ немає багатьох характеристик, якими повинна володіти мова для навчання ООП. Зокрема, це «гібридний» характер мови, використання небезпечних типів, дуже складна об'єктна модель і недостатня читабельність. Тому її класифікують як не найкращого кандидата для навчання ООП [10].

На противагу Java вважається «чистою» об'єктно-орієнтованою мовою і немає багатьох важливих недоліків С++. Ринковий успіх Java також є ще одним позитивним фактором на її користь і тому, з-поміж різноманітних альтернатив, Java видається кращим вибором для початку освоєння об'єктно-орієнтованого програмування [8, с. 3].

Корисною у вивченні ООП є мова Visual Basic, яка виступає повноцінним засобом програмування для Windows і вбудована у Windows Office. Це дає змогу непрофесійним програмістам розширювати можливості Word, Excel тощо. Однак ця мова не є популярною на теренах України [8, с. 3].

Важливо також дати відповідь на запитання: «*Яким повинен бути порядок викладання об'єктно-орієнтованих концепцій?*».

Викладання основ ООП на початковому рівні залишається серйозною педагогічною проблемою, оскільки потребує певних зусиль, перш ніж розпочнеться фактичне кодування [10]. Робота на цьому етапі повинна бути зосереджена на пошуку відповідей на такі питання: яким буде результат завершення виконання (роботи) програми?; які класи та об'єкти будуть потрібні та які їх функції?; яким буде інформаційне наповнення та функціональність кожного об'єкта? Як об'єкти спілкуватимуться? [7].

ООП та проєктування можна полегшити за допомогою простих графічних інтегрованих графічних допоміжних засобів. Навіть проста нотація, яка використовує прямокутники, овали та стрілки, може бути дуже корисною. Використання домашньої діаграмної нотації значно полегшує розуміння зв'язків об'єкт → клас → екземпляр [7].

Зауважимо, що викладання об'єктно-орієнтованих концепцій у школі має відбуватися системно та логічно, з урахуванням особливостей учнів і їхнього розвитку [10]. Як видається, викладання об'єктно-орієнтованих концепцій повинно відбуватися відповідно до загальної схеми:

1. Визначення вікових особливостей учнів:

– врахування психологічних особливостей учнів різного віку;

– врахування рівня математичної підготовки учнів [6].

2. Загальне введення в об'єктно-орієнтоване програмування:

– пояснення базових ідей та понять, таких як об'єкти, класи, спадкування, інкапсуляція та поліморфізм;

– використання простих прикладів та аналогій для роз'яснення концепцій [6].

3. Практичні завдання та проєкти:

– впровадження практичних завдань та проєктів для закріплення знань;

– створення малих програм або ігор, що дозволяють учням використовувати об'єктно-орієнтований підхід на практиці [6].

4. Засоби викладання:

– використання інтерактивних методів, таких як графічні інтерфейси, демонстрації, анімації;

– використання мов програмування, які підтримують об'єктно-орієнтований підхід, таких як Java, Python, або С++ [6].

5. Зв'язок із реальними прикладами:

– пояснення користі та застосування об'єктно-орієнтованого програмування у реальному житті;

– залучення прикладів з реальних проектів та індустрії, щоб показати учням, як об'єктно-орієнтований підхід використовується на практиці [6].

6. Зростання складності:

– поступове збільшення складності завдань та проектів для забезпечення поступового розвитку вмінь учнів;

– запровадження додаткових тем, таких як шаблони проектування, робота з базами даних, і т. д., на більш пізніх етапах [6].

7. Оцінювання та звітність:

– використання різних методів оцінювання, включаючи тести, проекти, лабораторні роботи та взаємну оцінку;

– регулярне оновлення програми на основі відгуків та власного досвіду

Важливо створити позитивне та зацікавлене середовище для учнів, де вони можуть ефективно вивчати та застосовувати об'єктно-орієнтовані концепції [6].

Не менш важливе питання – *яким має бути середовище навчання?* Для успішного вивчення вступного курсу з ООП важливим є відповідне середовище програмування, яке складається з редактора й операційної системи. Використання у навчанні інтегрованих графічних систем стає все поширенішим. Однак ці системи, по суті, є середовищами для професійного програмування, їх придатність як допоміжних засобів навчання під час вступного курсу є сумнівною, оскільки вони набагато потужніші та складніші, ніж потрібно, а їхні вдосконалені функції роблять їх менш зручними для такого курсу. Як компроміс, розроблено спеціальні середовища для об'єктно-орієнтованого навчання. Популярним прикладом у такій ситуації є система BLUE. Версії BLUE на С++ і Java забезпечують дуже просте та зручне середовище для створення методів, об'єктів і керування ними [5].

Відповідаючи на запитання: «*Коли і наскільки глибоко варто включати елементи моделювання та об'єктно-орієнтованого дизайну?»,* варто зазначити, що об'єктно-орієнтовані програмні продукти базуються на об'єктах та їхніх зв'язках. Основна

увага має бути зосереджена на розв'язанні проблем належного моделювання та проєктування об'єктів і класів. Тому починати з «Hello World!» є недоцільно, оскільки в такому випадку синтаксис ставиться перед вирішенням проблем програмування. Ефективним і, переважно, рекомендованим є підхід, який грунтується на використанні абстракції та дизайну з самого початку. Хоча абстрактне мислення може вважатися складним для засвоєння багатьма учнями, але воно є важливим компонентом об'єктно-орієнтованого програмування. Абстракція, з другого боку, виступає як компонент моделювання та дизайну також в інших предметах початкового рівня [5].

Отже, побудова ефективного підходу до вивчення об'єктно-орієнтованого програмування є важливою, оскільки значною мірою впливає на структуру даного курсу та успішність його засвоєння. Вивчення ООП повинно починатися з акценту на моделювання та дизайн об'єктів і класів, замість простого синтаксису. Абстракція виступає як важливий компонент навчання, хоча може бути важкою для розуміння для деяких учнів. Важливо зробити правильний вибір мови для навчання ООП – хоча мови, такі як C++, Java та Python, є популярними, все ж Java може бути кращим вибором для початківців через її чисто об'єктно-орієнтований характер та ринковий успіх. У контексті досліджуваних питань, важливою проблемою є ефективний порядок викладання ООП на початковому рівні – необхідно зосередитися на питаннях проєктування програм, використання графічних інструментів для полегшення розуміння концепцій та вирішення питань щодо взаємодії об'єктів. Водночас для успішного вивчення ООП необхідне відповідне середовище програмування. Спростити навчання можуть спеціальні середовища такі, наприклад, як система BLUE, яка надає зручне середовище для створення та управління об'єктами.

**References:**

1. Tsibulko M. Object-oriented programming: where to start. Computer in school and family. URL: http://nbuv.gov.ua/UJRN/komp_2011_4_4 2. [Опубліковано українською мовою]

2. Rudenko Yu. Professional adaptation of young computer science teachers. Physical and mathematical education. URL: http://fmo-journal.fizmatsspu.sumy.ua/publ/4-1-0-351 [Опубліковано українською мовою]

3. Morse N. Methodology of teaching informatics: teaching. manual: At 4 p.m. / Ed. Acad. E. Zhaldaka. K.: Educational Book, 2003. Part 1: General method of teaching computer science. 254 p. [Опубліковано українською мовою]

4. Semenikhina O., Rudenko Yu. Problems of teaching high school students to program and ways to overcome them. Information technologies and teaching aids. URL: https://www.researchgate.net/ publication/331403318_problemi_navcanna_programuvati_ucniv_starsih_klasiv_ta_slahi_ih_podolanna [Опубліковано українською мовою]

5. Ischeryakov S. Should you learn programming at school or university? URL: http://osvita.ua/school/54063/ [Опубліковано українською мовою]

6. Buzurin V. Methodology of teaching the basics of object-oriented programming to students of general secondary education institutions. URL: https://repository.sspu.edu.ua/handle/123456789/8442 [Опубліковано українською мовою]

7. Mahenko Ya., Stelmashenko Ya. Learning object-oriented programming at school. URL: http://eprints.zu.edu.ua/35740/1/%D0%95%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD_%D0%91%D0%BB%D0%BE%D0%BA_%D0%A2%D0%B5%D0%B7%D0%B8_%D0%A4%D0%BE%D1%80%D1%83%D0 %BC-102-104.pdf [Опубліковано українською мовою]

8. Lvov M., Spivakovskyi O. Introduction to object-oriented programming. Tutorial. Harkiv, 2000. 238 p. [Опубліковано українською мовою]

9. Programming should be learned at school or university. URL: https://osvita.ua/school/54063/ [Опубліковано українською мовою]

10. Vasenko O. Preparation of future mathematics teachers to implement the basic principles of object-oriented programming in the educational process of a modern school. URL: https://molodyivchenyi.ua/index.php/journal/article/view/5287 [Опубліковано українською мовою]