УДК 378.14

Сергій КОНЮХОВ

старший викладач кафедри інформатики і кібернетики
Мелітопольського державного педагогічного університету
імені Богдана Хмельницького, м. Мелітополь, Україна
e-mail: konukhov@mdpu.org.ua

НАВЧАЛЬНІ ЗАДАЧІ З ОБ'ЄКТНО-ОРІЄНТОВАНОГО ПРОГРАМУВАННЯ ЯК ЗАСІБ ФОРМУВАННЯ ПРОФЕСІЙНОЇ КОМПЕТЕНТНОСТІ МАЙБУТНІХ ІНЖЕНЕРІВ-ПРОГРАМІСТІВ

Стаття присвячена проблемам здійснення професійної підготовки майбутніх інженерів-програмістів у закладах вищої освіти. Розглядається один з аспектів цієї підготовки: формування у студентів професійної компетентності, зокрема здатності до розробки програм з використанням об'єктно-орієнтованого підходу і здатності до швидкого прийняття рішень у процесі створення програм. Досліджуються можливості використання навчальних задач у якості засобу реалізації цього завдання освітнього процесу. Сформульовано поняття «навчальна задача з об'єктно-орієнтованого програмування», запропоновані способи використання цих задач під час вивчення об'єктно-орієнтованого підходу у курсі програмування, наведені приклади формулювання і розв'язання деяких типів навчальних задач.

Ключові слова: майбутній інженер-програміст; заклад вищої освіти; професійна компетентність; об'єктно-орієнтоване програмування; навчальна задача з об'єктно-орієнтованого програмування.

Майбутній інженер-програміст – випускник закладу вищої освіти (ЗВО) повинен володіти комплексом компетентностей (інтегральна, загальні, професійна), які визначають його здатність і готовність до діяльності за фахом, професійного розвитку і самовдосконалення. Своєю чергою професійна компетентність включає, зокрема, здатність до використання методів об'єктно-орієнтованої парадигми (ООП) розробки програмного забезпечення і здатність до швидкого прийняття рішень у процесі створення програм.

Як зазначає у своїй роботі Р. Мартін [8], нині програмісти вимушені швидко знаходити рішення для виправлення проблем і помилок, оскільки тривалість циклу розробки програм є надзвичайно малою: процес контролю якості програмного продукту (від кодування до випуску фінальної версії) у деяких випадках може займати лічені хвилини. На думку Р. Мартіна здатність до швидкого прийняття рішень ґрунтується на уміннях розпізнавати ситуації і проблеми, а також знаннях способів рішення цих проблем. Наявність таких знань і умінь дозволяє програмістам зосереджуватись не на типових задачах і помилках, які розв'язуються автоматично, а на проблемах вищого рівня складності [8, 98-99].

Отже, у процесі навчання студентів об'єктно-орієнтованого програмування необхідно забезпечити формування і систематичне тренування умінь проектувати системи класів, використовувати об'єктно-орієнтовані засоби конкретних мов програмування, швидко виявляти і виправляти помилки. З цією метою у практиці викладання ООП слід застосовувати метод розв'язання навчальних задач.

У зв'язку з цим перед викладачем постає завдання щодо педагогічно обґрунтованого формування системи задач, яка б задовольняла низці критеріїв, а саме: відповідність змісту задач логіці і послідовності вивчення понять і методів ООП; наявність задач різних рівнів складності; урахування особистісних рис студентів (рівень підготовки з програмування, сформованість позитивної мотивації до тренування, готовність до розв'язування типових задач, готовність до підвищення рівня складності задач, потреба у зміні форм діяльності тощо), урахування обмежень навчального процесу в умовах ЗВО (незначний обсяг часу аудиторної роботи, дотримання вимог освітньо-кваліфікаційних програм, проблеми матеріально-технічної бази тощо).

Наразі проблеми здійснення професійної підготовки майбутніх інженерів-програмістів й ІТ-фахівців у закладах вищої освіти є достатньо добре дослідженими. Зокрема, її теоретичні і методологічні засади сформульовані у роботах Л. Гришко, В. Круглика, В. Осадчого, С. Семерікова й інших науковців. Реалізацію компетентнісного підходу у закладах вищої освіти досліджували О. Гура, М. Елькін, С. Лісова, О. Мамчич, Н. Побірченко й ін. Реалізацію компетентнісного підходу у процесі професійної підготовки майбутніх інженерів-

програмістів і фахівців з інформаційних технологій вивчали М. Вінник, Л. Зубик, О. Наумук, В. Седов, А. Стрюк, Д. Щедролосьєв й ін. Окремі аспекти навчання студентів ЗВО об'єктно-орієнтованого програмування висвітлені у роботах І. Бернакевич, В. Бублика, П. Вагіна, Ю. Грицюка, П. Кравця, М. Львова, Б. Мейєра, Т. Рака, О. Співаковського й ін.

Теоретико-методологічні основи використання навчальних задач для формування професійної компетентності майбутніх фахівців закладені у роботах Г. Балла, В. Беспалька, Д. Ельконіна, В. Давидова, І. Лернера, В. Серікова, Д. Толінгерової, Л. Фрідмана й ін. науковців. Окремі аспекти застосування навчальних задач у ЗВО досліджували В. Бєлікова (діагностика компетенцій засобами критеріальних задач), Т. Вакалюк, О. Кривонос (використання задач у процесі навчання програмування майбутніх вчителів інформатики), Я. Сікора (розробка системи навчальних задач для вивчення методів оптимізації) й ін. Компетентнісно-орієнтовані задачі як особливий тип навчальних задач досліджують О. Барна, О. Берсенєва, В. Вембер, І. Драч, Н. Жукова, П. Кубрушко, О. Кузьмінська, Н. Морзе, А. Фасоля, В. Хом'юк, М. Шингарьова й ін.

Метою статті є виокремлення поняття «навчальна задача з об'єктно-орієнтованого програмування» і визначення способів використання цих задач у якості засобу формування професійної компетентності майбутніх інженерів-програмістів.

Професійна компетентність інженера-програміста є складним особистісним утворенням. Дослідник професійної підготовки майбутніх інженерів-програмістів у закладах вищої освіти В. Круглик виділяє у її складі фахові (цифрова, математична, інженерна, з програмування) і загальні (комунікативна, особистісно-професійна, управлінська) компетентності [6, 141]. У межах нашого дослідження особливий інтерес становить компетентність з програмування. Як зазначає науковець: «успішний ІТ-фахівець має вміти швидко орієнтуватись у ... потоці інформації, вибирати методологію програмування для забезпечення оптимального вирішення завдань..., а також володіти розвиненим алгоритмічним мисленням для оперативного прийняття оптимальних рішень» [6, 142].

Зазначені уміння входять до складу компетентності з програмування і загалом професійної компетентності інженера-програміста, тому їх формування є важливим завданням професійної підготовки фахівців у ЗВО. Оскільки сучасна промислова розробка програмного забезпечення значною мірою ґрунтується на об'єктно-орієнтованій методології, то доцільно зробити акцент на формуванні здатності до використання ООП.

Одним із засобів формування професійної компетентності майбутнього інженера-програміста у процесі навчання у ЗВО є навчальні задачі з програмування, зокрема, з ООП.

Для визначення поняття «навчальна задача з об'єктно-орієнтованого програмування», спочатку розглянемо поняття «задача», «навчальна задача», «задача з програмування», «об'єктно-орієнтоване програмування».

У педагогічному словнику С. Гончаренка наведене таке визначення: «Задача – дана в певних умовах (наприклад, у проблемній ситуації) мета діяльності, яка повинна бути досягнута перетворенням цих умов згідно з певною процедурою. З. включає в себе вимогу (мету), умови (відоме) і шукане (невідоме), яке формулюється в запитанні.» [4, 130].

Навчальна задача є особливим видом задачі і представляє собою «задачу, пов'язану з перетворенням суб'єкта навчальної діяльності, із засвоєнням ним певних елементів змісту освіти – понять, способів дії, творчого або емоційно-ціннісного досвіду.» [9, 109]. Отже, під час розв'язання таких задач відбуваються зміни в особистості самого студента (учня).

Дослідники виділяють у структурі [навчальної] задачі такі компоненти: 1) умова (дані, які обумовлюють рішення, є посилками для міркування, що приведе до розв'язання задачі); 2) вимога (інформація про те, чого необхідно досягти у заданих умовах); 3) конструкт (сукупність дій, які необхідно виконати над умовами, щоб досягти розв'язання задачі) [1, 109].

За визначенням Т. Вакалюк «задача з програмування – це така задача, яка передбачає пошук алгоритму рішення задачі засобами деякої мови програмування» [3, 109]. Зауважимо, що вчена розглядає задачі з програмування у контексті навчання. На нашу думку таке визначення може стосуватися і реальних задач, які виконуються під час діяльності за фахом, тому доцільно зробити акцент на меті і результатах застосування навчальних задач.

Об'єктно-орієнтоване програмування (за Г. Бучем) – це «метод програмування, заснований на представленні програми у вигляді сукупності взаємодіючих об'єктів, кожний з яких є екземпляром певного класу, а класи є членами певної ієрархії спадкування» [2, 69]. Отже, об'єктно-орієнтовані програми повинні містити класи, пов'язані між собою відношенням спадкування, а також об'єкти, які є екземплярами цих класів.

Узагальнюючи наведені визначення, вважатимемо, що *навчальна задача з об'єктно-орієнтованого програмування* – це задача, що передбачає написання програми, яка містить ієрархічну

систему класів, адекватну предметній області, і екземпляри цих класів, а також ілюструє набір принципів ООП у межах визначеної теми.

Можна виділити такі типи навчальних задач з об'єктно-орієнтованого програмування за ознакою мети використання у процесі вивчення ООП: задачі – вправи, призначені для формування і закріплення умінь з об'єктно-орієнтованого програмування; задачі, призначені для діагностування рівня сформованості знань і умінь студентів; компетентнісно-орієнтовані задачі, спрямовані на формування професійної компетентності майбутніх інженерів-програмістів, а також діагностування її сформованості.

Розглянемо способи застосування навчальних задач з об'єктно-орієнтованого програмування для формування у майбутніх інженерів-програмістів професійної компетентності, а також окремих умінь і здатностей.

На початку вивчення ООП (тема «Класи і об'єкти» вступного курсу об'єктно-орієнтованого програмування) велике значення мають навчальні задачі – вправи, спрямовані на формування навичок визначення класів з заданими полями і методами. У наступних навчальних курсах їхня роль зменшується, оскільки вони є достатньо легкими для розв'язання, мають шаблонний зміст, дозволяють закріплювати лише найпростіші уміння. Вирішення таких задач є достатньо рутинною для студентів діяльністю, яка часто вимагає автоматичних, одноманітних дій. Разом із тим саме автоматизм написання подібного коду забезпечує зниження витрат часу програміста під час створення програмного продукту. Приклад формулювання умови навчальної задачі наведено на рис. 1, а програма для першого варіанту вихідних даних мовою C++ – на рис. 2.

Приклад, наведений на рис. 2, демонструє проблему, яка постає перед викладачем на початку вивчення ООП, а саме: кількість рядків коду значно більша, ніж у випадку використання структурного підходу. Отже, викладач має бути готовим довести студентам, що час, витрачений на написання коду, буде надалі компенсований за рахунок більш легкого і зрозумілого його розширення і повторного використання.

Створити клас з двома приватними полями і методами:
– сетери і гетери для обох полів;
– метод введення об'єкта з клавіатури;
– метод виведення об'єкта на екран;
– метод обробки значень полів.

В основній програмі необхідно створити екземпляр класу і викликати усі методи не менше одного разу.

Вихідні дані для створення класів і обробки значень полів наведені у таблиці

№ вар.	Поля	Метод обробки
1	Ціна товару Кількість одиниць товару	Обчислити вартість товару
2	Довжини катетів прямокутного трикутника	Обчислити довжину гіпотенузи
3	Швидкість руху (м/с) Час руху (с)	Обчислити пройдену за даний час відстань

Рисунок 1 – Приклад задачі початкового рівня з теми «Класи і об'єкти»

Розв'язування подібних задач у процесі навчання об'єктно-орієнтованого програмування можна також використовувати з метою формування у студентів здатності до аналізу, що забезпечується завдяки порівнюванню реалізації механізмів ООП у різних мовах програмування. Це доцільно робити після вивчення вступного курсу, коли у студентів сформовані необхідні базові знання. З цією метою слід пропонувати їм розв'язувати деякі задачі різними мовами. Так, наприклад, вивчаючи мову JavaScript, можна додатково виконувати окремі завдання мовою C++ або Java. На рис. 3 наведено фрагмент коду програми, написаної для вирішення сформульованої вище задачі мовами JavaScript і Java. Порівнюючи отримані програми, студенти повинні самостійно зробити висновок про наявні відмінності.

Дещо підвищити складність задач такого типу можна за рахунок зміни формулювання умови таким чином, щоб не було явно вказано, які поля і методи необхідно реалізувати. У такому випадку студент має самостійно визначити поля і методи, щоб отримати вірний розв'язок, не порушуючи правил ООП. Наприклад: «Створити клас *Трикутник*. Обчислити його периметр, площу, величини кутів, вид трикутника (прямокутний, рівносторонній, рівнобедрений, тупокутний). Реалізувати мінімально необхідну кількість полів і методів. Створити два перевантажених конструктори з параметрами.».

У процесі вивчення теми «Класи і об'єкти» для формування початкових навичок розробки класів можна запропонувати також навчальні задачі на композицію класів і об'єктів, які передбачають реалізацію декількох незалежних класів (один клас – основний, інші – допоміжні). Об'єкти допоміжних класів використовуються у якості полів основного класу [7, 25]. Такі задачі дозволяють поєднати в одній програмі відразу декілька класів, використовуючи типові операції створення класів і об'єктів.

З метою формування навичок аналізу коду, написаного іншим розробником, а також діагностики сформованості знань і умінь з ООП доцільно використовувати задачі на пошук помилок і заповнення пропусків у програмах. Формулювання такої задачі містить текст умови і програмний код з помилками та/або пропусками. Завдання

студента полягає у тому, щоб отримати робочу програму, результат виконання якої відповідає умові.

Як було зазначено вище, навчальні задачі, призначені для тренування, часто є доволі нецікавими для студентів, оскільки їхні формулювання часто жорстко формалізовані, а також достатньо віддалені від реальної практики програмної розробки. Для того, щоб подолати цю проблему, зробити процес розв'язування задач більш цікавим і забезпечити формування у студентів позитивної мотивації, можна використовувати способи тренування, описані Р. Мартіном: 1) програмні ката – строго визначений набір дій, які забезпечують вирішення певної програмної задачі, систематичне виконання ката дозволяє досягти автоматизму

```
#pragma argsused
#include <iostream>
using namespace std;

class TovarAmount
{
  private:
    double price;
    double quantity;
  public:
    double getPrice(){return price;}
    void setPrice(double value);
    double getQuantity(){return quantity;}
    void setQuantity(double value);
    void read();
    void print();
    double calcAmount(){return price*quantity;};
};


void TovarAmount::setPrice(double value)
{
    if(value<=0) throw "Price must be positive";
    price=value;
}

void TovarAmount::setQuantity(double value)
{
    if (value<0) throw "Quantity must be positive";
    quantity=value;
}
```

```
void TovarAmount::read()
{  double price, quantity;
   cout<<"Enter Price ";
   cin>>price;
   setPrice(price);
   cout<<"Enter Quantity  ";
   cin>>quantity;
   setQuantity(quantity);
}
void TovarAmount::print()
{
   cout<<"Price="<<price<<" Quantity="<<quantity<<endl;
}


int main(int argc, char* argv[])
{
    TovarAmount tovar;
    int a;
    tovar.read();
    tovar.print();
    cout<<"Cost "<<tovar.calcAmount()<<endl;
    tovar.setPrice(10);
    tovar.setQuantity(5);
    cout<<"Quantity "<<tovar.getQuantity()<<endl;
    cout<<"Price "<<tovar.getPrice()<<endl;
    cin>>a;
    return 0;
}
```

Рисунок 2 – Розв'язання навчальної задачі мовою C++

Java

```
public class TovarAmount
{

    private double price;
    private double quantity;

    public double getPrice(){return price;}
    public void setPrice(double value)
    {
       if(value<=0) throw new
       IllegalArgumentException("Price must
       be positive");
       price=value;
    }

                    ...

    public double calcAmount()
    {
       return price*quantity;
    }
}
```

JavaScript

```
function TovarAmount()
  {
      var price;
      var quantity;
      var self=this;

      this.getPrice=function ()
      {
        return this.price;
      }
      this.setPrice=function(value)
      {
        value=+value;
        if(value<=0) throw new RangeError("Price
        must be positive ");
         this.price=value;
      };

                      ...

      this.calcAmount=function()
      {
        return this.price*this.quantity;
      };
  }
```

Рисунок 3 – Порівняння реалізації механізмів ООП у різних мовах програмування

у написанні типових фрагментів коду (навіть на рівні «гарячих клавіш» і рухів мишею); 2) парне відпрацювання техніки: один програміст пише тест до задачі, а другий – програму, яка має пройти цей тест, потім вони обмінюються ролями (можна використовувати типові задачі для досягнення автоматизму їх розв'язування або нові задачі для формування нових ідей) [8, 100–102].

Застосування цих методів у процесі вивчення ООП має деякі обмеження, зокрема: достатньо великий розмір текстів програм і витрати часу на їх написання; різний рівень сформованості базових умінь у студентів. Перше обмеження можна подолати, якщо надавати студентам готові фрагменти коду або шаблони, які вони можуть використовувати під час тренування. Для подолання другого обмеження доцільно підготувати диференційовані за складністю задачі.

Наприкінці розглянемо ще один вид навчальних задач з ООП – компетентнісно-орієнтовані, спрямовані на формування у майбутніх інженерів-програмістів професійної компетентності.

За визначенням М. Шингарьової «Компетентнісно-орієнтована задача – це відображена у свідомості студента і об'єктивована у знаковій моделі проблемна ситуація, яка відповідає певному виду професійної діяльності та компетенції випускника.» [10, 7–8]. Дослідниця виділяє три кластери компетентнісно-орієнтованих задач: предметні (спрямовані на формування компетенцій у межах окремого навчального курсу), міждисциплінарні (спрямовані на формування компетенцій з урахуванням міждисциплінарних зв'язків), професійні компетентнісно-орієнтовані (спрямовані на формування професійної компетентності фахівця) [10, 8].

Аналізуючи наведене визначення, можна виділити такі суттєві характеристики компетентнісно-орієнтованих задач: наявність проблемної ситуації, пов'язаної з професійною діяльністю; подання проблемної ситуації у знаковій формі; свідоме ставлення студента до розв'язання задачі.

Навчальні задачі – вправи з ООП, розглянуті вище, не можуть використовуватися у якості повноцінних компетентнісно-орієнтованих задач, оскільки вони здебільшого спрямовані на формування окремих умінь, навичок, здатностей. Компетентнісно-орієнтовані задачі утворюють основу для виконання різних типів навчальних проектів у процесі вивчення ООП. Прикладом такого проекту є розробка простого векторного графічного редактора. Особливості реалізації проектного підходу описані нами у роботі [5]. Виконання курсових робіт передбачає розв'язання міждисциплінарних компетентнісно-орієнтованих задач, а дипломних робіт – професійних компетентнісно-орієнтованих задач. Прикладом професійної компетентнісно-орієнтованої задачі є розробка комп'ютерної гри.

Таким чином, нами були проаналізовані можливості застосування навчальних задач у якості засобу формування професійної компетентності майбутніх інженерів-програмістів. На основі аналізу понять «задача», «навчальна задача», «задача з програмування», «об'єктно-орієнтоване програмування» запропоновано визначення поняття «навчальна задача з об'єктно-орієнтованого програмування» і наведені приклади таких задач. Подальші дослідження спрямовані на розробку системи компетентнісно-орієнтованих задач, спрямованих на формування професійної компетентності майбутніх інженерів-програмістів і діагностування рівня її сформованості.

Список використаних джерел

1. Бєлікова В. В. Критеріальна задача як засіб діагностики професійних компетенцій майбутнього інженера-педагога. *Проблеми інженерно-педагогічної освіти*. 2014. № 45. С. 107-114.
2. Буч Г., Максимчук Р. А., Энгл М. У., Янг Б. Дж., Коналлен Д., Хьюстон К. А. Объектно-ориентированный анализ и проектирование с примерами приложений. Москва: ООО «И. Д. Вильямс», 2008. 720 с.
3. Вакалюк Т. А. Розв'язування творчих задач з програмування майбутніми учителями інформатики. *Вісник Чернігівського національного педагогічного університету. Серія: Педагогічні науки*. 2013. Вип. 113. С. 109-114.
4. Гончаренко С. У. Український педагогічний словник. Київ: Либідь, 1997. 373 с.
5. Конюхов С.Л. Применение проектных технологий в процессе изучения объектно-ориентированного программирования в университетах. *Университетская Наука. University Science. «Достижения, исследования, практика вузовской науки»*. Минеральные Воды: СКФ БГТУ им. В.Г.Шухова. 2016. № 2. С. 123-126.
6. Круглик В. С. Система підготовки майбутніх інженерів-програмістів до професійної діяльності у вищих навчальних закладах: дис. на здобуття наук. ступеня доктора пед. наук: спец. 13.00.04 – теорія і методика професійної освіти / Круглик Владислав Сергійович; Запорізький національний університет. Запоріжжя, 2018. 682 с.
7. Лаптев В. В., Морозов А. В., Бокова А. В. С++. Объектно-ориентированное программирование. Задачи и упражнения. Санкт-Петербург: Питер, 2007. 288 с.
8. Мартин Р. Идеальный программист. Как стать профессионалом разработки ПО. Санкт-Петербург: Питер, 2012. 224 с.
9. Сериков В. В. Образование и личность. Теория и практика проектирования педагогических систем. Москва: Издательская корпорация «Логос», 1999. 272 с. URL: http://pedlib.ru/Books/1/0157/.
10. Шингарева М. В. Проектирование компетентностно-ориентированных задач по учебным дисциплинам вуза: автореф. дис. на соискание ученой степени канд. пед. наук: спец. 13.00.08 – теория и методика профессионального образования / Шингарева Марина Валентиновна; Московский государственный агроинженерный университет имени В. П. Горячкина. Москва, 2012. 23 с.

References

1. Bielikova, V. V. (2014). Kryterialna zadacha yak zasib diahnostyky profesiinykh kompetentsii maibutnoho inzhenera-pedahoha [Criterion task as means of diagnostics of professional competenses of future engineer teacher]. In *Problems of engineering pedagogic education*, (45), 107-114. [in Ukrainian].
2. Buch, G., Maksimchuk, R. A., Ehngl, M. U., Yang, B. Dzh., Konallen, D., & H'yuston, K. A. (2008). *Ob"ektno-orientirovannyj analiz i proektirovanie s primerami prilozhenij [Object-oriented analysis and design with applications]*. Moscow: OOO «I. D. Vil'yams», 720. [in Russian].
3. Vakaliuk, T. A. (2013). Rozviazuvannia tvorchykh zadach z prohramuvannia maibutnimy uchyteliamy informatyky [Solution creative programming tasks of future teachers informatics]. In *Visnyk Chernihivskoho natsionalnoho pedahohichnoho universytetu. Seriia: Pedahohichni nauky [Bulletin of the Chernihiv National Pedagogical University. Series: Pedagogy]*, (113), 109-114. [in Ukrainian].
4. Honcharenko, S. U. (1997). *Ukrainskyi pedahohichnyi slovnyk [Ukrainian Pedagogical Dictionary]*. Kyiv: Lybid, 373. [in Ukrainian].
5. Konjuhov, S. L. (2016). Primenenie proektnyh tekhnologij v processe izucheniya ob"ektno-orientirovannogo programmirovaniya v universitetah [Application of project technologies in the process of studying object-oriented programming in universities]. In *Universitetskaya Nauka. University Science. «Dostizheniya, issledovaniya, praktika vuzovskoj nauki»*. Mineral'nye Vody: SKF BGTU im. V. G. Shuhova, (2), 123-126. [in Russian].
6. Kruhlyk, V. S. (2018). *Systema pidhotovky maibutnikh inzheneriv-prohramistiv do profesiinoi diialnosti u vyshchykh navchalnykh zakladakh [The system of training of future engineer-programmers for the professional activity at higher educational institutions]*. (Doctor of Pedagogic Sciences Thesis). Zaporizhzhia National University, Zaporizhzhia. [in Ukrainian].
7. Laptev, V. V., Morozov, A. V., & Bokova, A. V. (2007). *C++. Ob"ektno-orientirovannoe programmirovanie. Zadachi i uprazhneniya [Object-oriented programming. Issues and exercises]*. Saint-Petersburg: Piter, 288. [in Russian].
8. Martin, R. (2012). *Ideal'nyj programmist. Kak stat' professionalom razrabotki PO [The clean coder: A code of conduct for professional programmers]*. Saint-Petersburg: Piter, 224. [in Russian].
9. Serikov, V. V. (1999). *Obrazovanie i lichnost'. Teoriya i praktika proektirovaniya pedagogicheskih sistem [Education and personality. Theory and practice of designing pedagogical systems.]*. Moscow: Izdatel'skaya korporaciya «Logos», 272. Retrieved from http://pedlib.ru/Books/1/0157/. [in Russian].
10. Shingareva, M. V. (2012). Proektirovanie kompetentnostno-orientirovannyh zadach po uchebnym disciplinam vuza [Designing competency-oriented tasks in the academic subjects of the university]. (Author's abstract of Pedagogic Sciences Thesis). Moskovskij gosudarstvennyj agroinzhenernyj universitet imeni V. P. Goryachkina, Moscow. [in Russian].

Конюхов С. Л. Учебные задачи по объектно-ориентированному программированию как средство формирования профессиональной компетентности будущих инженеров-программистов

Статья посвящена проблемам осуществления профессиональной подготовки будущих инженеров-программистов в учреждениях высшего образования. Рассматривается один из аспектов этой подготовки: формирование у студентов профессиональной компетентности, в частности способности к разработке программ с использованием объектно-ориентированного подхода и способности к быстрому принятию решений в процессе создания программ. Исследуются возможности использования учебных задач в качестве средства реализации этой задачи образовательного процесса. Сформулировано понятие «учебная задача по объектно-ориентированному программированию», предложены способы применения этих задач при изучении объектно-ориентированного подхода в курсе программирования, приведены примеры формулировки и решения некоторых типов учебных задач.

Ключевые слова: будущий инженер-программист; высшее учебное заведение; профессиональная компетентность; объектно-ориентированное программирование; учебная задача по объектно-ориентированному программированию.

Koniukhov S. Educational tasks on object-oriented programming as a means of forming a professional competence of future software engineers

The future software engineers must have a wide set of competencies (integral, general, professional). These competencies determine their ability and readiness for working in the industry of software development, professional and personality improvement. This article is devoted to forming the professional competence of future software engineers during learning in universities. Professional competence has sophisticated structure. It includes, in particular, the ability to use object-oriented paradigm methods for software development and the ability to make fast decisions whilst creating, testing, and release applications. Thus if students want to be qualified specialists they should work on forming of these skills. On the other hand, university lecturers have to promote formation and development of these skills. We offer to use educational tasks as means of forming future software engineers' professional competence by training their object-oriented programming skills. We examined such concepts as "task", "educational task", "task on programming". As a result we formulated definition of concept "educational task on object-oriented programming". Main characteristic of such tasks is creation a programs with a hierarchical system of classes and instances of these classes which illustrates a set of object-oriented programming principles. We recognized three sorts of these tasks: tasks – exercises (for using as training means), tasks for using as control and assessment means, competence-oriented task (for using as professional competence forming means). Then we gave examples of using these tasks for professional training of future software engineers. Finally we formulated the further research direction that is creation of system of competence-oriented tasks on object-oriented programming.

Key words: future software engineer; institution of higher education; professional competence; object-oriented programming; educational task on object-oriented programming.

Стаття надійшла до редколегії 08.05.2018