DOI 10.18372/2786-5487.1.20292

УДК 377

Кравець Самійло,

викладач спеціальних дисциплін,

Відокремлений структурний підрозділ

«Київський фаховий коледж комп'ютерних технологій та економіки

ДНП ДУ«КАІ»,

м. Київ, Україна

samuilkravec@gmail.com

Морозова Вікторія,

провідний інженер,

Комунальне підприємство «Головний інформаційно-обчислювальний центр»,

м. Київ, Україна

immapouser@gmail.com

Морозова Наталія,

кандидат державного управління, заступник директора

департаменту – начальник управління,

Державна служба статистики України,

м. Київ, Україна

namo-72@ukr.net

Морозова Ярослава,

студентка 1 курсу,

Київський фаховий коледж зв'язку,

м. Київ, Україна

ywashwa.w@gmail.com

ВИКЛАДАННЯ ПРОГРАМУВАННЯ ЯК ПРИКЛАД НЕПЕРЕРВНОСТІ ОСВІТИ

Анотація. *Дане дослідження присвячене викладанню програмування, а саме послідовності етапів вивчення даної дисципліни для початкового ознайомлення з нею. Ми розглянемо це в дещо спрощеному вигляді, щоб це було зрозуміло не тільки викладачам, а й здобувачам освіти і школярам. Покажемо це на конкретних простих прикладах ефекивність мотивації до вивчення програмування.*

Ключові слова: *мотивація, алгоритм, програмування, клас, кроки.*

Annotation. *This study is devoted to teaching programming, namely the sequence of stages of studying this discipline for initial familiarization with it. We will consider this in a somewhat simplified form so that it is clear not only to teachers, but also to students and schoolchildren. We will show this with specific simple examples of the effectiveness of motivation to study programming.*

Key words: *motivation, algorithm, programming, class, steps.*

Вступ. Постановка проблеми у загальному вигляді та її зв'язок з важливими науковими та практичними завданнями. Освіта дозволяє кожній людині бути інтелектуально-збагаченою і мати можливість бути гарним спеціалістом, щоб отримувати гідну зарплату для матеріального забезпечення своєї сім'ї, а це в свою чергу сприяє розвитку української нації та нарощує інтелектуальний та загальний потенціал держави. Викладання є однією із складових освітнього процесу. Тому є дуже важливими перші кроки викладання програмування для закладання бази, яка допоможе у вивченні навчального матеріалу і примножить інтелект конкретної особистості і загальний потенціал суспільства і держави.

Аналіз досліджень і публікацій. На основі аналізу праці авторів [1] та й багатьох інших, що стосується теми дослідження відмітимо, що проблема зводиться, в основному, до демонстрації вивченння мови програмування, елементам ООП та візуального програмування. Іншим питанням на початку викладання приділялося уваги менше, або зовсім не приділялося.

Виділення невирішених раніше частин загальної проблеми. У нашому дослідженні ми показуємо, що перед вивченням основ програмування потрібно вивчити інформатику(або повторити), алгоритми, поняття програми тощо. Саме цим простим способом ми трохи пропагуємо політику держави щодо розвитку системи «навчання впродовж життя», яка залишається актуальною та потребує удосконалення і в сучасних умовах та ще й під час війни [2].

Мета та задачі дослідження. Метою даного дослідження є обрання послідовності етапів вивчення даної дисципліни для початкового ознайомлення з нею та наукова оцінка ефективності зацікавленості студентів і школярів до перших кроків вивчення програмування.

Виклад основного матеріалу дослідження. Початок навчання здобувачів освіти, які навчаються на програмістів, починається з вивчення(або повторення) інформатики, далі вивчаються алгоритми.

Отже, запишемо словесний алгоритм для знаходження найбільшого з трьох чисел a, b, d.

Приклад 1.

1. Якщо a>=b, то перейти до п. 4.
2. x покласти рівним b.
3. Перейти до п. 5.
4. x покласти рівним a.
5. Якщо x>=d, то перейти до п. 7.
6. x покласти рівним d.
7. Виведення x.

Загалом, як показує наш досвід, інтерес до складання словесних алгоритмів зростає, бо студенти на цьому простому прикладі одразу розуміють про що йде мова.

Можна записати словесний алгоритм шляху учня до школи, який буде зрозумілий навіть учням початкової школи.

Приклад 2.

1. Прокидаюсь та вмиваюсь.

2. Снідаю.

3. Одягаюсь і виходжу з дому.

4. Якщо є транспорт, то перейти до п. 6.

5. Іду пішки.

6. Заходжу в школу і прямую до класу.

Далі проілюструємо блок-схемний запис алгоритму, а саме запишемоблок-схему алгоритму для обчислення площі трикутника(s), якщо відомі довжина кожної сторони трикутника(a, b, c).

Для цього використаємо прийняті в програмуванні геометричні фігури. У нашому випадку це буде блок-схема простого лінійного алгоритму.

Приклад 3. Обчислити площу трикутника(s), якщо відомі довжина кожної сторони(a, b, c) .

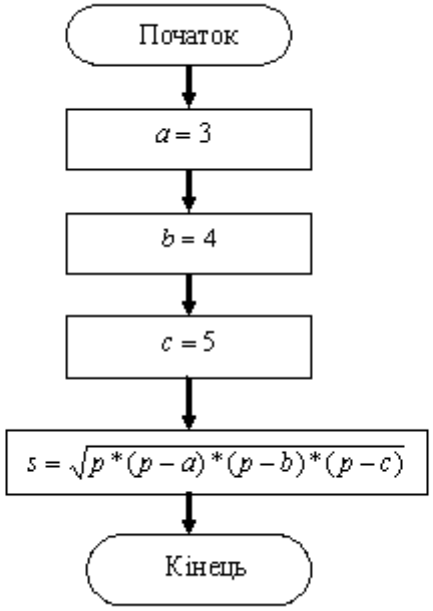

Рис. 1. Схема алгоритму прикладу 3

Наступний крок з вивчення основ програмування починається з алфавіту мови програмування, ключових слів мови, ідентифікаторів, [1, с. 4] тощо.

Після цього більшість авторів пропонують в якості першої програми:

//Моя перша програма мовою C++

#include <iostream.h>

```
int main() {cout<<”Привіт, студенте! Я C++!”;
return 0;}
```
[1, с. 7].

Використовуючи цю або подібні програми, викладачі, в основному демонструють роботу операторів введення/виведення, які зрозумілі студентам і не становлять труднощів для розуміння. В нашій статті пропонується взяти для написання першої програми квадратне рівняння. При цьому не втрачається зрозумілість програми, але, крім операторів введення/виведення, демонструються оператори для виконання математичних операцій та оператор розгалуження.

Наведемо текст програми:

```
//Моя перша програма
#include <stdio.h>   //бібліотечний файл введення/виведення
#include <math.h>   //бібліотечний файл математичних операцій
#include <conio.h>  //бібліотечний файл перетворення даних
main ()  //головна функція{
float a,b,c,d,x1,x2;      //оголошення даних
printf(“Введіть коефіцієнти рівняння”);
scanf(“%f %f %f”,&a,&b,&c); //введення коефіцієнтів рівняння
d=b*b-4*a*c; //обчислення дискримінанта
if(d>=0.0) //перевірка дискримінанта{
x1=(-b+sqrt(d))/2/a;//обчислення
x2=(-b-sqrt(d))/2/a;//коренів
printf(“ корені: x1= %f  x2= %f”,x1,x2);}
else
printf(“\n Дійсних коренів немає”);
return 0; //завершення програми      }
```

За допомогою коментарів ми досягаємо кращого розуміння, щодо роботи кожного оператора. При цьому доречно нагадати студентам про алгоритми і їх записи за допомогою блок-схем. І зразу ж пропонуємо скласти блок-схему алгоритму розв’язання квадратного рівння. Загалом тут ми не втрачаємо, а

навпаки, досягаємо більшої зацікавленості до вивчення кожного оператора, які застосовуються при розв'язанні добре відомого їм ще зі школи квадратного рівняння. Так, як їм відомий алгоритм, тому вони легше розуміють, для чого потрібен той чи інший оператор в даній програмі. При цьому ми досягаємо більшої ефективності до мотивації навчання, ніж коли б ми використовували тільки оператори введення/виведення, а також отримуємо взаємозв'язок з суміжною дисципліною, що вивчає алгоритмічну теорію.Загалом, як показує наш досвід, інтерес до вивчення наступних операторів зростає, бо студенти на цьому простому прикладі одразу розуміють про що йде мова. За нашими спостереженнями активність студентів і школярів до навчання зростає, а це швидше їх формує як особистість. Вони починають усвідомлювати свою значущість, бо в них починає виходити те, до чого вони прагнули ще до вступу в коледж чи інститут. Це в свою чергу спонукає їх ще більшого прикладання зусиль до вивчення дисципліни, що і приводить до здобуття якісної освіти і отримання спеціальності, яка популярна і стала дуже актуальною в усьому світі та ще й в умовах війни.

Далі йде етап вивчення структурного і процедурного програмування.

Структурне програмування - методологія програмування (модель конструювання програмного забезпечення), яку в 1970-х роках запропонував голландський науковець Дейкстра Едсгер (*Edsger Wybe Dijkstra*), була розроблена та доповнена Ніклаусом Віртом. Ґрунтується на теоремі Бьома-Якопіні (Corrrado Bohm, Giuseppe Jacopini), яка була опублікована у 1966 р.

Відповідно до цієї методології, будь – яку програму можна створити, використовуючи три конструкції:

- **послідовне виконання** - одноразове виконання операції в порядку апису їх (операцій) в тексті програми;

- **розгалуження**-виконання певної операції або декількох операцій залежно від стану певної, наперед заданої умови;

- **цикл** - багаторазове виконання операції або групи операцій за умови виконання деякої наперед заданої умови. Таку умову називають

умовою продовження циклу. Кожна конструкція є блоком з одним входом і одним виходом) [3].

Процедурне програмування – парадигма програмування, заснована на концепції виклику *процедури*. Процедури, також відомі як підпрограми, методи, або функції (це не математичні функції, але функції, подібні до тих, які використовуються в функціональному програмуванні).

Процедури містять певну послідовність кроків для виконання. В ході виконання програми будь-яка процедура може бути викликана з будь-якого місця програми, включно з самої процедури, яка викликається (рекурсивний виклик) [4].

Наведемо текст програми:

```
#include <iostream.h>
main() {
cout << "Введіть число ";
int a;
cin >> a;
if (a > 11) {
// Обидві операції будуть виконані, якщо a > 11
cout << "Ви ввели " << a << "\n";
cout << a << " більше ніж 11\n"; }
else {
// Обидві операції будуть виконані, якщо a <= 11
cout << "Ви ввели " << a << "\n";
cout << a << " не більше ніж 11\n";}
return 0;  }
```

Наступним етапом є вивчення об’єктно – орієнтованого програмування. Ключовим поняттям є поняття класу. Клас використовують для означення множини об’єктів, які мають однакову структуру, поведінку та відношення між об’єктами класів.

Визначення (оголошення) класу записується таким чином:

class <ім’я класу>

{

[специфікатор доступу:]

<тип атрибуту 1>< ім’я атрибуту 1 >;

<тип атрибуту 2>< ім’я атрибуту 2>;

...

[специфікатор доступу:]

<тип функції 1>< ім’я функції 1>;

<тип функції 2>< ім’я функції 2>;

....

};

Дані та методи класу захищені оголошенням класу:

class ім’я_класу {

private:

дані та методи класу

public:

дані та методи класу

protected:

дані та методи класу

}

С++ надає даним та методам 3 рівні захисту:

- закриті (використовуються за замовчуванням) – private – можуть використовуватись тільки у межах класу;

- відкриті – public – досяжні у класі та поза класом у межах дії встановленого простору імен;

- захищені – protected – доступні у самому класі та похідних від нього.

Дозволяється довільний порядок розміщення розділів private, public та protected у протоколі класу. Оголошення класу синтаксично є подібним до оголошення структури [1, с. 112].

Наведений нижче клас під іменем ClassPoint визначає тип майбутнього об’єкта, призначеного для зберігання координат точки на площині:

```
class ClassPoint { // оголошення класового типу
 double x;
 double y;
 public:
 void Init();// ініціалізація даних класу ClassPoint
 void Set(double, double); // зміна значень даних об’єкта класу
 void Get(); // виведення з об’єкта значення
 };
```

Наступним етапом є вивчення візуального програмування. Серед них є C++ BUILDER [1, с. 139] і Visual C++ та інші. У візуальному програмуванні структурною одиницею є візуальний об’єкт, який називається компонентом.

Існують компоненти форма та кнопка. Проект – це сукупність файлів з яких складається візуальна програма.

Висновки. Таким чином, ми можемо бачити, як ми можемо досягати професійного розвитку при проходженні кожного етапу в процесі викладаннян навчальної дисципліни. І саме тому в процесі викладання перших кроків програмування слід керуватися і враховувати спеціальність, яку здобувають студенти і враховувати свій і їхній практичний досвід.

Бо, як сказано в законі про освіту [5] важливою складовою у моделі «освіта впродовж життя» має стати мотивація до професійного розвитку та до самовдосконалення, формування свідомості, фахової багатопрофільності, творчого зростання, ефективної реалізації знань та навичок у праці.

У формуванні мотиваційної системи, яка покликана стимулювати розвиток повинна брати активну участь держава і розробляти відповідну політику.

Список використаних джерел

1. Глинський Я.М., Анохін В.Є., Ряжська В.А. С++ і С++Builder: Навч. посібн. 4-те вид. Львів : СПД Глинський, 2008. 192 с.

2. Про введення воєнного стану в Україні: Указ Президента України від 24.02.2022 р. № 64/2022 (з останніми змінами, внесеними згідно з Указом Президента № 235/2025 від 15.04.2025). URL: https://zakon.rada.gov.ua/laws/show/64/2022#Text (дата звернення: 03.02.2025).

3. Структурне програмування. *Вікіпедія.* URL: https://uk.wikipedia.org/wiki/структурне_програмування (дата звернення: 03.02.2025).

4. Процедурне програмування. *Вікіпедія.* URL: https://uk.wikipedia.org/wiki/процедурне_програмування (дата звернення: 05.02.2025).

5. Про освіту: Закон України від 5 вересня 2017 р. № 2145-VIII (з останніми змінами, внесеними згідно із Законом № 4353-IX від 15.04.2025). URL: https://zakon.rada.gov.ua/laws/show/2145-19#Text (дата звернення: 03.02.2025).