УДК 378:147:51:044.9

Вдовичин Т. Я., Лазурчак Л. В.

Дрогобицький державний педагогічний університет імені Івана Франка, Дрогобич, Україна

НАВЧАННЯ ОСНОВ ПРОГРАМУВАННЯ СТУДЕНТІВ ФІЗИКО-МАТЕМАТИЧНОГО ПРОФІЛЮ

DOI: 10.14308/ite000631

У статті наведено методичні рекомендації щодо вивчення навчальної дисципліни «Інформатика» для підготовки фахівців першого (бакалаврського) рівня вищої освіти галузі знань 01 «Освіта» спеціальності 014.04 «Середня освіта (математика)», 014.08 «Середня освіта (фізика)». Ця дисципліна відіграє особливо важливу роль у підготовці фахівців ВНЗ фізико-математичного профілю, бо поєднує в собі як фундаментальні поняття і принципи різних математичних та інформатичних дисциплін, так і прикладні моделі й алгоритми їх застосування.

Методичні аспекти щодо вивчення дисципліни «Інформатика» передбачають педагогічну доцільність форм, методів та засобів навчання для студентів, які отримують кваліфікацію вчитель математики та вчитель фізики відповідно. Програма дисципліни включає питання теоретичних основ інформатики, прикладного програмного забезпечення, основ програмування.

Студентам пропонується розглянути базові основи програмування у середовищі C++. Основні конструкції мови C++ мають зручний професійний інструментарій для програмування. Інтегроване середовище C++ характеризується швидкістю, зручністю щодо відлагодження та компілювання програми. Тому у статті акцентовано увагу на формування практичних навичок роботи в середовищі C++ для студентів фізико-математичного профілю та висвітлено методичні аспекти застосування мови програмування C++ у процесі навчання дисципліни «Інформатика». Формування практичних навичок відбувається під час виконання лабораторних робіт, а саме: постановка вихідної задачі, побудову алгоритму її розв'язку, аналіз отриманих результатів.

***Ключові слова**: вчитель математики, вчитель фізики, «Інформатика», мова програмування C++.*

Постановка проблеми. Підготовка студентів зорієнтована на розвиток здатності сприймати нові знання, спонукання до педагогічних пошуків, прояву власної активності та вміння реалізовувати набуті знання не тільки під час навчання у ВНЗ, а й у подальшій діяльності. У навчанні студентів фізико-математичного профілю важливим є засвоєння фундаментальних понять, набуття навичок практичної роботи. Основний шлях реалізації завдання – навчити студента методологічному мисленню, надати відомості з предметної галузі, сформувати вміння до практичного застосування, втілювати набуті навички у практичну діяльність.

Значну роль у підготовці фахівців першого (бакалаврського) рівня вищої освіти галузі знань 01 «Освіта» спеціальності 014.04 «Середня освіта (математика)», 014.08 «Середня освіта (фізика)» відведено дисципліні «Інформатика». Оскільки студенти даних спеціальностей отримують кваліфікацію «вчитель математики» та «вчитель фізики» відповідно, то вміння та навички, що формуються при вивченні дисципліни «Інформатика», мають загально навчальний, інтелектуальний характер і можуть бути перенесені на вивчення інших предметів.

Майбутні вчителі математики та фізики, розпочинаючи процес навчання у ВНЗ, знайомляться з нормативними та варіативними дисциплінами. Саме на першому році навчання викладається дисципліна «Інформатика», під час вивчення якої студенти повинні сформувати базу знань, вмінь та отримати навички, які їх будуть супроводжувати і при вивченні наступних дисциплін, а також у подальшій професійній діяльності.

Отже, необхідно відшукати методичні підходи до організації навчання студентів ВНЗ, що сприяли б глибокому засвоєнню і розумінню ними базових понять, правил, принципів і методів навчання дисциплін, їх взаємозв'язку з суміжними дисциплінами, а також шляхів їх використання на практиці.

Одним із перспективних шляхів для навчання студентів фізико-математичного профілю є інтегрування у процес навчання дисципліни «Інформатики» основ програмування. Формування практичних навичок можна продемонструвати з допомогою мови програмування С++. Такий підхід розрахований для студентів з метою вивчення основ алгоритмізації та програмування. Методичні аспекти навчання студентів основ програмування за допомогою мови С++ сприятимуть підготовці у розв'язуванні задач, включаючи способи формалізації та алгоритмізації задач, методи розробки програм, рекомендації по налагодженню і тестуванню програм

Аналіз останніх досліджень та публікацій. На початку 70-х рр. співробітник фірми Bell Telephon Laboratories (США) Деніс Рітчі розробив мову С. Б'ярн Страуструп домігся, що мова С++ стала об'єктно-орієнтованою версією мови С.

Програмування мовою С++ розглядаються у наукових публікаціях вітчизняних авторів, зокрема, Дудзяного І.М. [6], Глинського Я.М., Анохіна В.Є., Ряжської В.А. [4], Глушакова С.В., Коваля А.В., Смирнова С.В. [5], Наконечного С.І. [7], Нікольського Ю.В., Пасічника В.В., Щербини Ю.М. [8].

Вайсфельд М. [2], Гилберт С., Маккарти Б. [3], Страуструп Б. [1], Шилдт Г. [11] досліджували об'єктно-орієнтований підхід в С++.

У посібнику [6] систематизовано базові засоби процедурного програмування мовою С++. Для організації ефективної індивідульної роботи студентів в середовищі С++ корисний збірник задач [4].

Об'єкт дослідження: процес навчання дисципліни «Інформатика» для студентів фізико-математичного профілю у педагогічному університеті.

Предмет дослідження: особливості вивчення мови програмування С++ у навчальній дисципліні «Інформатика».

Метою дослідження є розглянути особливості використання мови програмування С++ при вивченні навчальної дисципліни «Інформатика» для студентів фізико-математичного профілю.

Виклад основного матеріалу. Під час навчання дисципліни «Інформатика» для студентів фізико-математичного профілю передбачається вивчення теоретичних основ інформатики, системного та прикладного програмного забезпечення, основ програмування та формування практичних навичок роботи в середовищі С++. Завданнями цієї дисципліни є розкрити значення предмету в загальній і професійній освіті людини, вплив засобів сучасних інформаційних систем на науково-технічний і соціально-економічний розвиток суспільства; поглибити знання студентів щодо роботи із офісним пакетом Microsoft Office, комп'ютерними мережами та комп'ютерною графікою; виробити в студентів навички застосування здобутих знань при виконанні лабораторних та курсових робіт з інших предметів.

Результатами вивчення дисципліни є знання основ інформатики, зокрема, поняття алгоритму, основних структур алгоритму, алгоритмізації завдань; апаратне й програмне забезпечення комп'ютера; базові конструкції мови С++. Студенти повинні вміти використовувати засоби роботи в локальній мережі та в мережі Інтернет; користуватися можливостями програмного забезпечення для реалізації прикладних завдань за допомогою текстового процесора та електронних таблиць; реалізувати типові алгоритмічні конструкції

засобами C++; використовувати складені програми, здійснювати їх супровід, вносити зміни та виконувати відлагодження програм за допомогою вбудованих інструментальних засобів.

Аналізуючи програму навчальної дисципліни «Інформатика» для студентів фізико-математичного профілю, можна зробити висновок про те, що базові алгоритмічні конструкції та основні аспекти програмування пропонується реалізовувати у середовищі C++.

Мова С – сучасна універсальна мова програмування, призначена для створення прикладних програмних продуктів та системних компонентів програмного забезпечення комп'ютерів [10]. Це ефективний і зручний професійний інструментарій для програмування у різних практичних сферах.

На мові C++ розробляють програми практично для всіх типів комп'ютерів, а також для різних операційних середовищ. Інтегроване середовище C++ призначене для швидкого і зручного запису та редагування тексту програм, їх компілювання і відлагодження, компонування великих програмних проектів. C++ включає спеціальні засоби та бібліотеки, що реалізують принципи об'єктно-орієнтованого програмування.

Основними рисами мови C++ є:

- потужність;
- гнучкість;
- ефективність;
- структурованість;
- модульність;
- лаконічність;
- мобільність;
- функціональність.

Мову програмування C++ часто використовують для створення компіляторів, редакторів, комп'ютерних ігор і програм мережевого обслуговування. Крім цього, програмне забезпечення багатьох високоефективних систем побудоване з використанням мови C++, зокрема, найчастіше вибирається для Windows-програмування. Це актуально для студентів фізико-математичного профілю. Мова C++ надає студенту певний набір інструментів для розробки конструкцій програм з використанням вбудованих можливостей. Процедури у C++ виконуються за допомогою виклику бібліотечних функцій, що, зокрема, підвищує функціональну гнучкість мови. Для виконання різних задач, студент може створити бібліотеку функцій, тобто персоніфікувати мову C++ відповідно до своїх потреб.

Широкий набір можливостей розв'язування прикладних задач мови програмування C++ дає можливість для її застосування у професійній діяльності майбутніх учителів математики та фізики. Використання C++ при вивченні «Інформатики» дозволить підвищити рівень професійної підготовки студентів, математичної та інформаційної культури.

Дуже часто при плануванні навчального процесу, розробці різноманітних завдань і критеріїв оцінювання навчальних досягнень студентів викладачі спираються на власний досвід і досвід своїх колег. Часто досить ефективними виявляються ті навчальні роботи, які підготовлені або модифіковані самим викладачем.

Зосередимо увагу на процес навчання студентів фізико-математичного профілю основам програмування з допомогою мови C++ на дисципліні «Інформатика». Наведемо методичні рекомендації щодо вивчення середовища програмування C++.

На **лекційних заняттях** викладач знайомить студентів з базовими конструкціями мови програмування C++ (основні поняття, типи даних, оператор присвоєння, потоки, введення-виведення даних); організація розгалужень – команда **if** та команда вибору **switch**; організація циклічних процесів (циклічна настанова **for**; ітераційні настанови **while** та **do-while**; механізм використання настанов **continue** та **break**); одновимірні та двовимірні масиви (оголошення та ініціалізація елементів одновимірних та двовимірних масивів); функції користувача (оголошення, опис та виклик функцій користувача); приклади розв'язування типових задач.

На **лабораторних заняттях** студентам пропонується ознайомитися з середовищем програмування С++, лінійними програмами та реалізацією найпростіших математичних обчислень у середовищі С++, програмуванням розгалужених алгоритмів, циклічними програмами, одновимірними та багатовимірними масивами, функціями користувача. Структура лабораторних робіт передбачає: формування вихідної задачі, побудову алгоритму її розв'язку, аналіз отриманих результатів.

Метою першої лабораторної роботи є вивчення особливостей середовища С++, компілювання програми, базових елементів мови, основних типів даних у С++, способів введення та виведення даних. Під час виконання даної лабораторної роботи студентам пропонується виконати такі завдання: запустити середовище С++; активізувати головне меню; створити програму на мові С++, яка виводитиме на екран домашню адресу студента; здійснити компіляцію програми та запустити її на виконання; переглянути вікно результатів.

На лабораторних роботах ***«Лінійні програми» та «Реалізація найпростіших математичних обчислень в середовищі програмування С++»*** студенти ознайомлюються зі змінними та сталими, керуючими послідовностями, арифметичними операціями та операціями порівняння, математичними функціями, операціями інкременту та декременту. Зокрема, оскільки мова програмування С++ викладається для майбутніх вчителів математики та фізики, то вважатимемо, що їм актуально та цікаво буде розв'язати такі задачі:

- Обчислити периметр і площу ділянки лісу за заданими сторонами, що має форму рівнобічної трапеції.
- Скільки хвилин мають доба, тиждень, рік?
- Яку відстань долає світло за годину, добу?
- Обчислити площу поверхні та об'єм Місяця, якщо його радіус становить 1740 км.

Лабораторні роботи ***«Програмування розгалужених алгоритмів»*** демонструють можливості мови С++ з допомогою операторів **if** та **switch**.

Приклад 1 (використання оператора **if**).

Використовуючи повну форму умовного оператора мови програмування С++, визначити, чи належить точка з координатами (x, y) заштрихованій області, зображеній на рисунку 1.

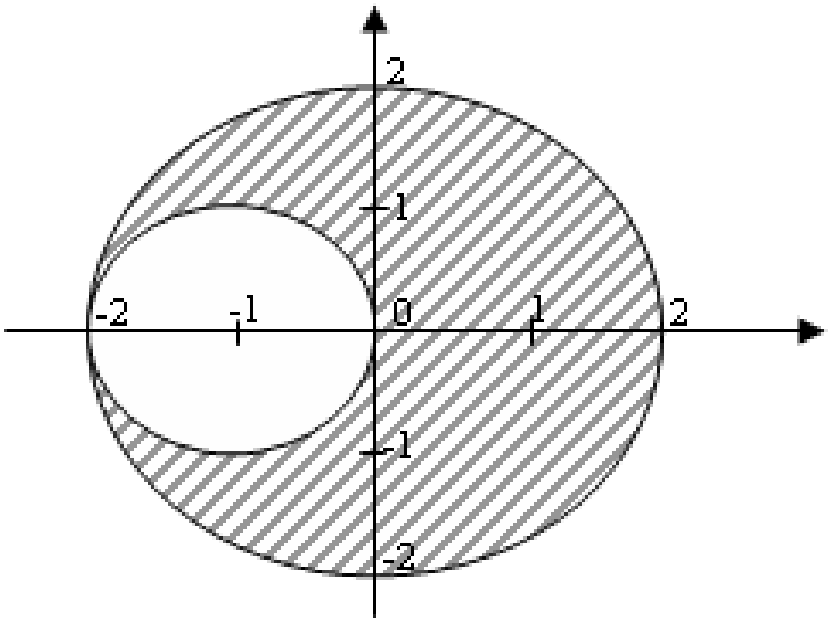


Рис. 1. *Умова для прикладу 1.*

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
void main()
{
clrscr();
float x,y;
cout<<"\n x=";cin>>x;
cout<<"\n y=";cin>>y;
pow(x,2)+pow(y,2)<=4 && pow(x+1,2)+pow(y,2)>=1 ?
cout<<"Tochka nalegyt oblasti":cout<<"Tochka ne nalegyt oblasti";
getch();
}
```

Рис. 2. *Реалізація прикладу 1 в С++, використовуючи оператор* ***if***

```
x=-0.5

 y=0.1
Tochka ne nalegyt oblasti_
```

```
x=-0.5

y=1

Tochka nalegyt oblasti
```

Рис. 3. *Результати виконання прикладу 1*

Приклад 2 (використання оператора **switch**).

Записати програму, що виконує одну з операцій «+», «-», «*», «/» над заданими дійсними числами a, b.

```
#include <iostream.h>
#include <conio.h>
void main()
{ clrscr();
int f=1;
float a,b,rez;
char op;
cout<<"\n a=";cin>>a;
cout<<"\n b=";cin>>b;
cout<<"\n op=";cin>>op;
switch (op){
case '+':rez=a+b;break;
case '-':rez=a-b;break;
case '*':rez=a*b;break;
case '/':rez=a/b;break;
default: cout<<"\n  Error operation";f=0;
}
if (f==1) cout<<"\n rez="<<rez;
getch();
}
```

Рис. 4. *Реалізація прикладу 2 в С++, використовуючи оператор* ***switch***

Для закріплення оператора **switch** студентам можна запропонувати виконати такі завдання, зокрема, скласти програму, яка б виводила на екран розклад роботи на день; за порядковим номером місяця, вивести його кількість днів.

Метою лабораторної роботи ***«Циклічні програми»*** є вивчити дію операторів повторення та їх практичне застосування. Оскільки, у мові С++ є три команди циклу – **for, while** та **do-while**, то студентам пропонується створити програму, використавши всі задані типи циклів.

Таблиця 1.

Результати виконання програми, реалізованої оператором switch

Результати виконання програми				
Операція «+»	**Операція «-»**	**Операція «*»**	**Операція «/»**	**Помилкова операція**
a=45 b=55 op=+ rez=100_	a=100 b=50 op=- rez=50_	a=60 b=3 op=* rez=180	a=250 b=5 op=/ rez=50	a=25 b=5 op=t Error operation

Приклад 3.

Обчислити $\sum_{i=1}^{15} \frac{\sin 10i + \cos\frac{10}{i}}{\sqrt{i}}$.

Таблиця 2.

Результати виконання циклічної програми

Результати виконання програми з використанням:	**Програмна реалізація**

Циклу з лічильником for:	`#include <iostream.h>`
	`#include <math.h>`
	`#include <conio.h>`
	`#define n 10`
	`void main()`
	`{`
	`clrscr();`
	`float s=0;`
	`for(int i=1;i<=n;i++)`
	`s+=(sin(10*i)+cos(10/i)/sqrt(i));`
	`cout<<"\n s="<<s;`
	`getch();`
	`}`

Циклу з передумовою:	`#include <iostream.h>`
	`#include <math.h>`
	`#include <conio.h>`
	`#define n 10`
	`void main()`
	`{`
	`clrscr();`
	`float s=0;int i=1;_`
	`while (i<=n)`
	`{s+=(sin(10*i)+cos(10/i)/sqrt(i));`
	`i++;}`
	`cout<<"\n s="<<s;`
	`getch();`
	`}`

Циклу з після умовою:	`#include <iostream.h>`
	`#include <math.h>`
	`#include <conio.h>`
	`#define n 10`
	`void main()`
	`{ clrscr();`
	`float s=0;int i=1;`
	`do`
	`{`
	`s+=(sin(10*i)+cos(10/i)/sqrt(i));`
	`i++;`
	`}`
	`while (i<=n);`
	`cout<<"\n s="<<s; getch();}`

Результати виконання програми:	`s=-0.911013`

Важливо зосередити увагу студентів на лабораторній роботі «***Ітераційні цикли***». Продемонструємо організацію ітераційних циклів на конкретному прикладі.

Приклад 4.

Нехай x – деяке число, яке необхідно ввести з клавіатури під час виконання програми, $e = 0.001$ – точність обчислень. Обчислити суму елементів знакозмінного ряду $\sum_{n=1}^{\infty} a_n$, де $a_n = (-1)^n \frac{(2x)^n}{n!}$. Визначити кількість доданків. Вивести на екран результати обчислень.

Розв'язання:

Визначимо значення доданка при $n = 1$.

Отримаємо - $d = -2 * x$.

Обчислимо значення коефіцієнта, на який необхідно помножити попередній елемент, щоб отримати наступний - $m = \frac{a_{n+1}}{a_n} = -\frac{2*x}{(n+1)}$.

Сумування будемо проводити до тих пір, поки не виконається наступна умова - $|d| < \varepsilon$.

Таблиця 3.

Результати виконання ітераційної циклічної програми

Результати виконання програми з використанням:	**Програмна реалізація**

Циклу з передумовою (while)	`#include<iostream.h>`
	`#include<math.h>`
	`#include<conio.h>`
	`void main()`
	`{`
	`clrscr();`
	`const float e=0.0001;`
	`float x,d,m,s=0;`
	`int n=1;`
	`cout<<"\n x=";cin>>x;`
	`d=-2*x;`
	`while (fabs(d)>e)`
	`{s+=d;`
	`n++;`
	`m=-2*x/(n+1);`
	`d*=m;`
	`}`
	`cout<<"\n s="<<s;`
	`cout<<"\n Kil dod="<<n;`
	`}`

Цикл з післяумовою (do-while):	`#include<iostream.h>`
	`#include<math.h>`
	`#include<conio.h>`
	`void main()`
	`{`
	`clrscr();`
	`const float e=0.0001;`
	`float x,d,m,s=0;`
	`int n=1;`
	`cout<<"\n x=";cin>>x;`
	`d=-2*x;`
	`do`
	`{s+=d;`
	`n++;`
	`m=-2*x/(n+1);`
	`d*=m; }`
	`while (fabs(d)>e);`
	`cout<<"\n s="<<s;`
	`cout<<"\n Kil dod="<<n;`
	`}`

Результати виконання програми:	`x=5.67`
	
	`s=-1.823745`
	`Kil dod=35`

На лабораторних роботах «***Одновимірні та багатовимірні масиви***» студентам пропонуються завдання, в яких зустрічаються такі типові умови обробки елементів масиву, що наведені в таблиці 4 [1].

Таблиця 4.

Умови обробки елементів масиву

Умови	Реалізація умови на С++
Парні елементи масиву	$A[i]\%2==0$
Непарні елементи масиву	$A[i]\%2!=0$
Елементи масиву кратні k	$A[i]\%k==0$
Елементи масиву не кратні k	$A[i]\%k!=0$
Елементи масиву, що стоять на парних місцях	$i\%2==0$
Елементи масиву, що стоять на непарних місцях	$i\%2!=0$
Додатні елементи масиву	$A[i]>0$
Від'ємні елементи масиву	$A[i]<0$
Елементи масиву, що знаходяться в інтервалі (n_1,n_2)	$(A[i]>n_1 \,\&\&\, A[i]<n_2)$

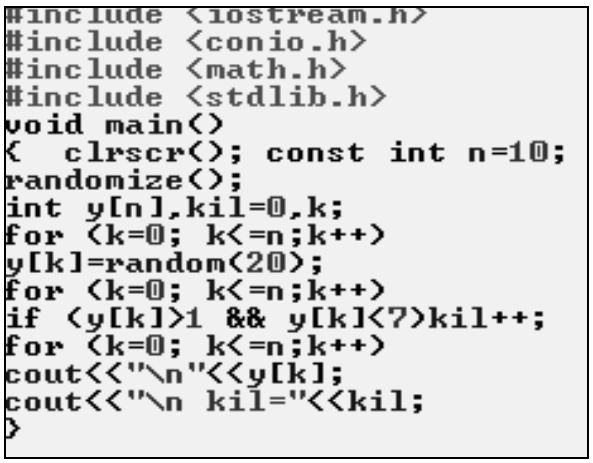
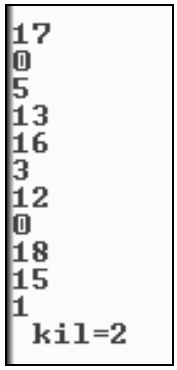
Приклад 5.

Утворити масив з 10-ти елементів за допомогою генератора випадкових чисел (числа з діапазону від 0 до 20). Обчислити кількість елементів масиву, що попадають в інтервал $(1,7)$.

Варто зауважити, для утворення масиву слід скористатися функцією random(n), що генерує випадкове число з діапазону від 0 до *n,* яка описана у модулі stdlib.h. Для отримання різних значень елементів масиву при багаторазовому записку програми на виконання, треба записати функцію randomize().

Таблиця 5.

Результати виконання програми (приклад 5)

Програмна реалізація	Результати виконання програми:
1 спосіб – звернення до елементів масиву по імені масиву та порядковому номеру елемента в масиві: 	

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
void main()
{  clrscr(); const int n=10;
randomize();
int y[n],kil=0,k;
for (k=0; k<=n;k++)
y[k]=random(20);
for (k=0; k<=n;k++)
if (y[k]>1 && y[k]<7)kil++;
for (k=0; k<=n;k++)
cout<<"\n"<<y[k];
cout<<"\n kil="<<kil;
}
```

```
17
0
5
13
16
3
12
0
18
15
1
 kil=2
```

2 спосіб – використання динамічного розподілу пам'яті:

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
#include <stdlib.h>
void main()
{  clrscr(); const n=10;
randomize();
int *y=new int[n],kil=0;
for (int k=0; k<n;k++)
{  *(y+k)=random(20);
  cout<<"\n y["<<k<<"]="<<*(y+k);  }
for (k=0; k<n;k++)
if (*(y+k)>1 && *(y+k)<7)kil++;
cout<<"\n \n kil="<<kil;
delete[]y;_
}
```

```
y[0]=18
y[1]=6
y[2]=14
y[3]=4
y[4]=8
y[5]=4
y[6]=18
y[7]=11
y[8]=17
y[9]=19

kil=3
```

Елементи багатовимірних масивів визначаються іменем масиву та двома індексами: перший індекс означає номер рядка, інший – номер стовпця, на перетині яких розміщений елемент. Нумерація індексів масиву завжди починається з нуля. Багатовимірні масиви компілятор розглядає як послідовність одновимірних. Тому до елементів багатовимірного масиву, як і для одновимірних, можна також звертатись через вказівники. Багатовимірні масиви автоматично ініціалізуються «по рядках», тобто спочатку модифікується зовнішній (правіший) індекс.

Приклад 6.

Знайти максимальний та мінімальний елементи двовимірного масиву a та поміняти їх місцями. Доступ до елементів масиву здійснювати через вказівники. Вивести на екран вихідний масив у вигляді таблиці.

```
#include <iostream.h>
#include <conio.h>
#include <math.h>
void main()
{  clrscr(); const int n=4;
int i,j,r1,s1,r2,s2,z,max,min;
 int y[n][n]={{4,13,-6,9},{67,-9,11,123},{3,7,8,99},{4,8,55,2}};
max=**y;min=**y;
for (i=0; i<n;i++)   {
for (j=0; j<n;j++)
cout<<*(*(y+i)+j)<<"\t";cout<<"\n";  }
for  (i=0; i<n;i++)
for  (j=0; j<n;j++)
{ if  (*(*(y+i)+j)>max) {max=*(*(y+i)+j);r1=i;s1=j;}
if  (*(*(y+i)+j)<min) {min=*(*(y+i)+j);r2=i;s2=j;}}
cout<<"\n max="<<max<<"\n min="<<min;
z=*(*(y+r1)+s1); *(*(y+r1)+s1)=*(*(y+r2)+s2);  *(*(y+r2)+s2)=z;
cout<<"\n";
for (i=0; i<n;i++) {
for (j=0; j<n;j++)
cout<<*(*(y+i)+j)<<"\t"; cout<<"\n";  } }
```

Рис. 5. Програмна реалізація прикладу 6.

```
4       13      -6      9
67      -9      11      123
3       7       8       99
4       8       55      2

 max=123
 min=-9
4       13      -6      9
67      123     11      -9
3       7       8       99
4       8       55      2
```

Рис. 6. *Результати виконання прикладу 6*

Метою лабораторної роботи «***Функції користувача***» є освоїти способи організації функцій користувача та їх практичне застосування, зокрема, оголошення функції користувача, їх опис та виклик.

Приклад 7.

Дано дійсні a, b.

Обчислити: $v = \dfrac{h(a/b,1,b)+h(b,1,ab)}{h(1.7,ab,9)}$, де $h(x,y,z)=\dfrac{x+y+z}{xy}$.

```
#include <iostream.h>
#include <math.h>
#include <conio.h>
float h(float x,float y,float z)
{
float v;
v=(x+y+z)/(x*y);
return(v);
}
void main()
{
clrscr();
float a,b,rez;
cout<<"\n a=";cin>>a;
cout<<"\n b=";cin>>b;
rez=(h(a/b,1,b)+h(b,1,a*b))/h(1.7,a*b,9);
cout<<"\n rez="<<rez;
}
```

Рис. 7. *Програмна реалізація прикладу 7*

```
a=3.4
b=6.75
rez=24.271826
```

Рис. 8. *Результати виконання прикладу 7*

Кожна лабораторна робота супроводжується списком запитань для самоперевірки і низкою завдань для виконання під час самостійної роботи студентів. Основним завданням є формування у майбутніх фахівців практичних навичок програмування на мові С++.

Отже, при складанні завдань для виконання лабораторних робіт, слід ретельно підходити до визначення рівня складності. Навчальний матеріал поданий таким чином, що, починаючи вже з першої теми, можна приступити до практичної реалізації в середовищі С++. Теми розташовані в такому порядку, щоб кожна наступна не тільки надавала нові знання, але й закріплювала і розширювала знання з попередньої теми. Для лабораторних робіт доцільніше ставити завдання, де виконання задач більш високого рівня складності можливе за умови виконання завдань попереднього рівня складності. Задачі різних рівнів складності доцільно використовувати під час проведення контролів і в екзаменаційних білетах, щоб охопити весь навчальний матеріал.

Підсумовуючи розгляд навчальної дисципліни «Інформатика» щодо вивчення базових конструкцій мови програмування С++, слід зазначити, що широкий набір інструментів даного середовища (зокрема, розробка інтерфейсу текстових та графічних редакторів, швидкодія та ефективність, реалізація принципів структурного програмування, модульність, орієнтація на професіоналізм програміста, лаконічність, можливість переносу програм з однієї операційної платформи на іншу, тобто програми можуть бути виконані на комп'ютерах різних виробників і в різних операційних системах) дає можливість його адаптації у навчальний процес педагогічного університету.

Висновки. Педагогічна практика свідчить, що міжпредметність відіграє важливу роль у навчальному процесі педагогічного університету, зокрема, сприяє підвищенню прикладної, практичної і науково-теоретичної підготовки студентів. Крім цього, узагальнення набутих

знань, умінь та навичок дає можливість для їх застосування не тільки в конкретних ситуаціях, а й у майбутній професійній діяльності. Навчальна дисципліна «Інформатика» для майбутніх учителів математики та фізики, окрім теоретичних основ інформатики, системного та прикладного програмного забезпечення, розширюється, поглиблюється вивченням основ програмування з допомогою С++. Це сприятиме її відповідності принципам міжпредметності та узагальненості набутих практичних знань та вмінь. Базові конструкції мови програмування С++ дають можливість моделювати прикладні задачі, побудувати алгоритми їх розв'язку, програмно реалізувати та отримати результати.

Реалізація на мові С++ технологій об'єктно-орієнтованого програмування забезпечує ефективне застосування практично до будь-якого поставленого завдання для студентів фізико-математичного профілю. Ще одним важливим аспектом мови програмування С++ є структурованість (підтримка концепції функцій і локальних змінних), найхарактернішою особливістю якої є використання блоків, тобто набору настанов, які логічно пов'язані між собою.

Використання мови програмування С++ у процесі навчання дисципліни «Інформатика» дозволило змінити акценти у доборі теоретичного матеріалу; збільшити частку реалізації математичних та фізичних задач у середовищі С++; практикувати завдання на порівняння результатів, щодо реалізації програм різними способами та підходами; підбивати підсумки виконання поставлених завдань.

Перспективи подальших досліджень. Використання мови програмування С++ для навчання студентів фізико-математичного профілю педагогічного університету у подальших дослідженнях варто спрямувати щодо застосування типових алгоритмів опрацювання скалярних величин, скінченних послідовностей, рядів, рядків, задач комбінаторної оптимізації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Бьерн Страуструп. Язык программирования С++. Второе дополненное издание / Страуструп Бьерн. – М., БИНОМ, 2015. – 1136 с.
2. Вайсфельд Мєтт. Обьектно-ориентированный подход: Java, .Net, C++ / Метт Вайсфельд. – М. : КУДИЦА-ОБРАЗ, 2005.
3. Гилберт С. Самоучитель Visual C++ в примерах / С. Гилберт, Б. Маккарти. – К. : ТИД ДС, 2003.
4. Глинський Я.М., Анохін В.Є., Ряжська В.А. С++ і С++ Builder: Навч. посібн. 5-те вид.– Львів: СПД Глинський, 2011. – 192 с.
5. Глушков С.В. Язык программирования С++ / С.В. Глушаков, А.В. Коваль, С.В. Смирнов. – Харьков : Фолио, 2003.
6. Дудзяний І.М. Програмування мовою С++. Частина 1 : Парадигма процедурного програмування: навчальний посібник / І.М. Дудзяний. – Львів : ЛНУ імені Івана Франка, 2013. – 468 с.
7. Наконечний С.І. Математичне програмування : навч. посібник / С.І. Наконечний, С.С. Савіна. – К. : КНЕУ, 2003.
8. Нікольський Ю.В. Дискретна математика : підручник / Ю.В. Нікольський, В.В. Пасічник, Ю.М. Щербина. – К. : BHV, 2007.
9. Павловська Т.А., Щупак Ю.А. С/С++Структурне програмування. Практикум. –Пітер, 2007.
10. Павловськая Т.А. С/С++ Программирование на языке высокого уровня / Т.А. Павловськая. – СПб. : Питер, 2008.
11. Шилдт Г. С++: базовый курс. 3-е издание / Г. Шилдт. – М.: Вильямс, 2010.

REFERENCES (TRANSLATED AND TRANSLITERATED)

1. Bern, Straustrup. (2015). Yazyik programmirovaniya C++ . Vtoroe dopolnennoe izdanie. M., BINOM, 1136.
2. Vaysfeld, Mett. (2005). Obektno-orientirovannyiy podhod: Java, .Net, C++. M.: KUDITSA-OBRAZ.
3. Gilbert, S. (2003). Samouchitel Visual C++ v primerah / S. Gilbert, B. Makkarti. K.: TID DS.
4. Glinskiy, Ya. M., AnohIn, V.E., Ryazhska, V.A. (2011). C++ i C++ Builder: Navch. posIbn. 5-te vid. LvIv: SPD Glinskiy, 192.
5. Glushkov, S. V., Koval, A. V., Smirno, S. V. (2003). Yazyik programmirovaniya C++. Harkov : Folio.

6. Dudzyaniy, I. M. (2013). Programuvannya movoyu C++ . Chastina 1 : Paradigma protsedurnogo programuvannya: navchalniy posIbnik. Lviv : LNU ImenI Ivana Franka, 468.
7. Nakonechniy, S. I., Savina, S. S. (2003). Matematichne programuvannya : navch. posIbnik. K.: KNEU.
8. Nikolskiy, Yu. V., PasIchnik, V. V., Scherbina, Yu. M. (2007). Diskretna matematika: pidruchnik. K.: BHV.
9. Pavlovska, T. A., Schupak, Yu. A. (2007). C/C++ Strukturne programuvannya. Praktikum. Piter.
10. Pavlovskaya, T. A. (2008). C/C++ Programmirovanie na yazyike vyisokogo urovnya. SPb.: Piter.
11. Shildt, G. (2010). C++: bazovyiy kurs. 3-e izdanie. M.: Vilyams.

Стаття надійшла до редакції: 01.04.2017

Vdovychyn Tatiana Yaroslavivna, Lazurchak Liubov Vasylivna
Drohobych Ivan Franko State Pedagogical University, Drohobych, Ukraine

PROGRAMMING FUNDAMENTALS TEACHING TO THE STUDENTS OF PHYSICO-MATHEMATICAL PROFILE

The article provides methodical recommendations on studying of the discipline "Informatics" for the specialists preparation of the first (Bachelor) level of higher education of the field of knowledge 01 "Education" of the specialty 014.04 "Secondary education (mathematics)", 014.08 "Secondary education (physics)". This discipline plays a particularly important role in the higher education establishments physical and mathematical field specialists training, since it combines both the fundamental concepts and principles of various mathematical and informatics disciplines, as well as applied models and algorithms for their application.

The methodological aspects of the discipline "Informatics" study include the pedagogical feasibility of the forms, methods and means of training for students who are qualified as a teacher of mathematics and a physics teacher respectively. The discipline program includes issues on informatics theoretical foundations, applied software, and the basics of programming.

Students are encouraged to consider the basics of programming in the C ++ environment. Basic C ++ language designs have a convenient, professional programming toolkit. Integrated C ++ environment is characterized by speed, convenience in debugging and compiling of the program. Therefore, the article focuses on the practical skills formation in the C ++ environment for the students of the physical and mathematical profile and highlights the methodological aspects of the C ++ programming language use in the course of the discipline "Informatics" teaching. The formation of practical skills takes place during the performance of laboratory works, namely: the original problem setting, the construction of an algorithm for its solution, analysis of the received results.

Key words: teacher of mathematics, teacher of physics, "Informatics", programming language C++.

Вдовичин Татьяна Ярославовна, Лазурчак Любовь Васильевна
Дрогобычский государственный педагогический университет имени Ивана Франко, Дрогобыч, Украина

ОБУЧЕНИЯ ОСНОВАМ ПРОГРАММИРОВАНИЯ СТУДЕНТОВ ФИЗИКО-МАТЕМАТИЧЕСКОГО ПРОФИЛЯ

В статье приведены методические рекомендации по изучению учебной дисциплины «Информатика» для подготовки специалистов первого (бакалаврского) уровня высшего образования области знаний 01 «Образование» специальности 014.04 «Среднее образование (математика)», 014.08 «Среднее образование (физика)». Эта дисциплина играет важную роль в подготовке специалистов вузов физико-математического профиля, потому что совмещает в себе как фундаментальные понятия и принципы различных математических и информатических дисциплин, так и прикладные модели и алгоритмы их применения.

Методические аспекты по изучению дисциплины «Информатика» предусматривают педагогическую целесообразность форм, методов и средств обучения для студентов, получающих квалификацию учитель математики и учитель физики соответственно.

Программа дисциплины включает вопросы теоретических основ информатики, прикладного программного обеспечения, основ программирования.

Студентам предлагается рассмотреть базовые основы программирования в среде С ++. Основные конструкции языка С ++ имеют удобный профессиональный инструментарий для программирования. Интегрированная среда С ++ характеризуется скоростью, удобством по отладки и компиляции программы. Поэтому в статье акцентировано внимание на формирование практических навыков работы в среде С ++ для студентов физико-математического профиля и освещены методические аспекты применения языка программирования С ++ в процессе обучения дисциплине «Информатика». Формирование практических навыков происходит при выполнении лабораторных работ, а именно: постановка исходной задачи, построение алгоритма ее решению, анализ полученных результатов.

Ключевые слова: учитель математики, учитель физики, «Информатика», язык программирования С ++.