Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Кафедра комп'ютерної фізики

"ЗАТВЕРДЖУЮ"

Проректор з науково-педагогічної роботи

_____ Антон ТАНТЕЛЕЙМОНОВ

"30" червня 2020 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

ОБ'ЄКТНО – ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

рівень вищої освіти	перший(бакалаврський)
галузь знань	10 Природничі науки
спеціальність	105 Прикладна фізика та наноматеріали
освітні програми	«Комп'ютерна фізика»
	«Прикладна фізика енергетичних систем»
	«Прикладна фізика нетрадиційної енергетики»
вид дисципліни	обов'язкова
навчально – науковий інститут	комп'ютерної фізики та енергетики

2020 / 2021 навчальний рік

Програму рекомендовано до затвердження вченою радою навчально-наукового інституту комп'ютерної фізики та енергетики
30 червня 2020 року, протокол № 6-2/20

РОЗРОБНИКИ ПРОГРАМИ:
Нємченко Костянтин Едуардович, доктор фізико-математичних наук, професор
Рогова Світлана Юріївна, кандидат фізико-математичних наук, доцент

Програму схвалено на засіданні кафедри комп'ютерної фізики
Протокол від 25 червня 2020 року № 6-3/20

Завідувач кафедри комп'ютерної фізики

_____ Костянтин НЄМЧЕНКО

Програму погоджено з гарантом освітньо – професійної програми «Комп'ютерна фізика»

Гарант освітньо – професійної програми «Комп'ютерна фізика»

_____ Світлана РОГОВА

Програму погоджено з гарантом освітньо – професійної програми
«Прикладна фізика енергетичних систем»

Гарант освітньо – професійної програми «Прикладна фізика енергетичних систем»

_____ Руслан СУХОВ

Програму погоджено з гарантом освітньо – професійної програми
«Прикладна фізика нетрадиційної енергетики»

Гарант освітньо – професійної програми «Прикладна фізика нетрадиційної енергетики»

_____ Ілля МАРУЩЕНКО

Програму погоджено методичною комісією навчально-наукового інституту комп'ютерної фізики та енергетики
Протокол від 30 червня 2020 року № 6/20

Голова методичної комісії ННІ КФЕ

_____ Ольга ЛІСІНА

ВСТУП

Програма навчальної дисципліни «Об'єктно – орієнтоване програмування» складена відповідно до освітньо-професійних програм підготовки «Комп'ютерна фізика», «Прикладна фізика енергетичних систем», «Прикладна фізика нетрадиційної енергетики» першого (бакалаврського) рівня вищої освіти спеціальності 105 Прикладна фізика та наноматеріали.

1. Опис навчальної дисципліни

1.1. Мета викладання навчальної дисципліни

Метою курсу «Об'єктно – орієнтоване програмування» є вивчення сучасної теорії об'єктно орієнтованого програмування з використанням актуальних інтегрованих серед розробки програмних продуктів та набуття практичних навичок програмування.

1.2. Основні завдання вивчення дисципліни

Основними **завданнями** вивчення дисципліни є набуття студентами знань і навичок програмування мовами C++, Java, Python необхідних для автоматизованої обробки великих об'ємів даних та комп'ютерного моделювання.

1.3. Кількість кредитів 4

1.4. Загальна кількість годин 120

1.5. Характеристика навчальної дисципліни	
Нормативна	
Денна форма навчання	Заочна (дистанційна) форма навчання
Рік підготовки	
2-й	-й
Семестр	
3-й	-й
Лекції	
32 год.	год.
Практичні, семінарські заняття	
-	год.
Лабораторні заняття	
32 год.	год.
Самостійна робота	
56 год.	год.
У тому числі індивідуальні завдання	
0 год.	

1.6. Заплановані результати навчання

Згідно з вимогами освітньо-професійної програми студенти повинні досягти таких результатів навчання:

знати: теоретичні засади об'єктно орієнтованого програмування на прикладі мов C++, Java, Python.

вміти: використовувати на практиці теоретичні знання для кодування мовами C++, Java, Python.

2. Тематичний план навчальної дисципліни

Розділ 1. Основи об’єктно орієнтованого програмування.

Тема 1. Основні ідеї об’єктно орієнтованого програмування.

Основні поняття ООП: інкапсуляція, спадкування, поліморфізм, спадкування. Об'єктно-орієнтоване програмування (ООП) як парадигма програмування, що спирається на концепцію класів та об'єктів. Застосовується для структурування програмного забезпечення у прості частини коду (які зазвичай називаються класами), та використовуються для створення окремих екземплярів об’єктів. Оскільки ООП є парадигмою програмування, існує багато об’єктно-орієнтованих мов програмування, включаючи: C ++, Java та Python.

Тема 2. Структурний та об’єктно-орієнтований підходи у програмуванні. Поняття класу. Дані-члени. Методи-члени. Статичні члени класу.

В парадигмі ООП програміст розробляє програмне забезпечення, організовуючи пов’язану інформацію та поведінку разом у шаблон, який називається клас. Потім із шаблону класу створюються окремі об’єкти. Вся програма виконується шляхом взаємодії декількох об'єктів з об'єктами для створення більшої програми. Поняття класу. Структура класу Дані-члени. Методи-члени.

Тема 3. Модифікатори доступу: приватні, захищені, відкриті члени класу. Інкапсуляція даних класу.

Інкапсуляція даних, як це механізм, що об'єднує дані і код, який маніпулює з цими даними, а також захищає і те, і інше від зовнішнього втручання або неправильного використання. В ООП код і дані можуть бути об'єднані разом (в так званий «чорний ящик») при створенні об'єкта. Усередині об'єкту коди і дані можуть бути закритими або відкритими. Закриті коди або дані доступні тільки для інших частин того ж самого об'єкта і, відповідно, недоступні для тих частин програми, які існують поза об'єктом. Відкриті коди і дані, навпаки, доступні для всіх частин програми, в тому числі і для інших частин того ж самого об'єкта.

Тема 4. Конструктор, деструктор. Конструктор та деструктор за замовченням. Вказівники та посилання на об’єкти класів. Конструктор копіювання.

Конструктор класу, як спеціальна функція-член класу, що виконується кожного разу, коли ми створюємо нові об'єкти цього класу. Конструктор та його структура: ім'я, тип повернення. Конструктори та їх використання для встановлення початкових значень для певних змінних-членів.

Тема 5. Оператори new та delete. Вказівник this. Ключове слово const при визначенні вказівника та функції-члена.

Ініціалізація, розподіл і очищення пам'яті. Використання ключового слова this. Інші ключові слова.

Тема 6. Спадкування, його види. Передача аргументів в конструктори базових класів.

Спадкування: новий, або похідний клас може бути визначений на основі вже наявного, або базового класу. При цьому новий клас зберігає всі властивості старого: дані об'єкта базового класу включаються до цих об'єкта похідного, а методи базового класу можуть бути викликані для об'єкта похідного класу, причому вони будуть виконуватися над даними включеного в нього об'єкта базового класу. Тобто, новий клас успадковує як дані старого класу, так і методи їх обробки. Об'єкт, що успадковує свої властивості від одного з батьків – одиночне спадкування. Якщо об'єкт успадковує дані і методи від декількох базових класів, то це – множинне спадкування.

Тема 7. Перевизначення методів у похідних класах. Поняття поліморфізму. Раннє та пізнє зв’язування.

Поліморфізм підтипів (в ООП званий просто «поліморфізмом») - властивість системи, що дозволяє використовувати об'єкти з однаковим інтерфейсом без інформації про тип і внутрішню структуру об'єкта. В межах одного класу можна визначити два або більше методів, які спільно використовують один і той же ім'я, але мають різну кількість параметрів. Коли це має місце, методи називаються перевантаженими, а про процес кажуть як про перевантаження методу.

Перевантаження методу - один із способів, за допомогою яких реалізується поліморфізм. Перевантаження методів в класах проводиться за тими ж правилами, що і перевантаження функцій. Якщо для функції, що викликається немає точної відповідності, то компілятор здійснює пошук підходящої функції за трьома рівнями послідовно: пошук серед методів класу; пошук серед методів базових класів, послідовно від найближчого предка до самого першого; пошук серед інших функцій. Якщо точної відповідності ні на одному рівні не знайдено, але знайдено декілька відповідних функцій на різних рівнях, то використовується функція, знайдена на мінімальному рівні.

Тема 8. Віртуальні функції. Поняття абстрактного класу та чисті віртуальні функції. Віртуальні деструктори. Множинне спадкування, його види та правила застосування.

Віртуальні функції - спеціальний вид функцій-членів класу. Віртуальна функція відрізняється про звичайну функції тим, що для нормальної функції зв'язування виклику функції з її визначенням здійснюється на етапі компіляції. Для віртуальних функцій це відбувається під час виконання програми. Для оголошення віртуальної функції використовується ключове слово virtual. Функція-член класу може бути оголошена як віртуальна, якщо клас, що містить віртуальну функцію, базовий в ієрархії породження; реалізація функції залежить від класу і буде різною в кожному породженому класі. Віртуальна функція - це функція, яка визначається в базовому класі, а будь-який породжений клас може її перевизначити. Віртуальна функція викликається тільки через покажчик або посилання на базовий клас. Визначення того, який екземпляр віртуальної функції викликається за висловом виклику функції, залежить від класу об'єкта, що адресується покажчиком або посиланням, і здійснюється під час виконання програми. Цей механізм називається динамічним (пізнім) зв'язуванням або дозволом типів під час виконання.

Тема 9. Друзі класу. Друзі-функції та друзі-класи. Вибір між оператором-членом та оператором-другом.

Дружня функція - це функція, яка має доступ до закритих членів класу, як якщо б вона сама була членом цього класу. У всіх інших відносинах дружня функція є звичайною функцією. Нею може бути, як звичайна функція, так і метод іншого класу. Для оголошення дружньої функції використовується ключове слово friend перед прототипом функції, яку ви хочете зробити дружній класу.

Розділ 2. Практичне використання концепцій об’єктно орієнтованого програмування.

*Тема 10. Перевантаження операторів +, -, /, *, =, +=, -=, *=, /=, [ ], ( ), ->, *, постфіксного и префіксного ++ та --).*

Перевантаження операторів дозволяє визначити дії, які буде виконувати оператор. Перевантаження має на увазі створення вибору опцій, що містить слово operator і символ перевантаження оператора. Функція оператора може бути визначена як член класу, або поза класом. Перевантажити можна тільки ті оператори, які вже визначені в C ++. Створити нові оператори не можна.

Тема 11. Перевантаження операторів потокового введення/виведення. Форматування виводу. Обробка виключень.

За допомогою перевантаження операторів домагаються того, щоб введення-виведення значень об'єктів через потоки можна було використовувати за допомогою стандартних в цьому випадку операторів "<<" і ">>". Перевантажити можна оператор виведення в потік ostream, що дозволить використовувати екземпляри класу в рядку потоку виведення з таким же синтаксисом, як і для вбудованих типів. При цьому в якості лівого операнда оператор << повинен мати неконстантне посилання на потік ostream, тобто даний оператор завжди визначається зовнішньої функцією, а не методом класу.

Тема 12. Ієрархія потокових класів. Класи istream, ostream.

Потоки даних. Формування потоків. Введення та виведення даних до потоків. Аналіз та управління станом потока.

Тема 13. Стандартне виведення та введення даних

Стандартні консольні потоки введення/виведення. Введення даних з використанням cin. Методи об'єкта cin. Введення одного символу та строки. Виведення даних з використанням cout. Методи об'єкта cout.

Тема 14. Введення та виведення даних в файли

Використання файлів для введення/виведення даних. Класи ofstream та ifstream. Бінарні та текстові файли.

Тема 15. Шаблони та бібліотека стандартних шаблонів

Поняття шаблону. Шаблонні функції. Шаблонні класи. Статичні члени класу та шаблони. Стандартна бібліотека шаблонів. Контейнерні класи. Ітератори. Послідовні контейнери: vector, list, stack, queue. Асоціативні контейнери: map, multimap, set, multiset.

Тема 16. Класи алгоритмів. Алгоритми, що не змінюють послідовність. Алгоритми, що змінюють послідовність.

Алгоритм як послідовність точно визначених дій, що однозначно призводять до вирішення поставленого завдання. Процес алгоритмізації передує процесу програмування. Властивості алгоритмів: дискретність , масовість, результативність, детермінованість. Використання блок-схем, на яких кожен крок алгоритму позначається спеціальною геометричною фігурою, а всередині її записуються прості операції. Напрямок виконання алгоритму позначається стрілками. Типи алгоритмів: лінійний алгоритм, розгалужений алгоритм, циклічний алгоритм.

3. Структура навчальної дисципліни

Назви розділів	Кількість годин											
	денна форма						заочна форма					
	усього	у тому числі					усього	у тому числі				
		л	п	лаб.	інд.	с. р.		л	п	лаб.	інд.	с. р.
1	2	3	4	5	6	7	8	9	10	11	12	13
Розділ 1. *Основи об'єктно орієнтованого програмування.*												
Тема 1	8	2		2		4						
Тема 2	8	2		2		4						
Тема 3	8	2		2		4						
Тема 4	8	2		2		4						
Тема 5	8	2		2		4						
Тема 6	8	2		2		4						
Тема 7	8	2		2		4						
Тема 8	8	2		2		4						
Тема 9	7	2		2		3						
Разом за розділом 1	71	18		18		35						
Розділ 2. *Практичне використання концепцій об'єктно орієнтованого програмування.*												
Тема 1	7	2		2		3						
Тема 2	7	2		2		3						
Тема 3	7	2		2		3						
Тема 4	7	2		2		3						
Тема 5	7	2		2		3						
Тема 6	7	2		2		3						
Тема 7	7	2		2		3						
Разом за розділом 2	49	14		14		21						
Усього годин	120	32		32		56						

4. Теми лабораторних занять

№ з/п	Назва теми	Кількість годин
1	**Тема 1.** Основні ідеї об’єктно орієнтованого програмування.	2
2	**Тема 2.** Структурний та об’єктно-орієнтований підходи у програмуванні. Поняття класу. Дані-члени. Методи-члени. Статичні члени класу.	2
3	**Тема 3.** Модифікатори доступу: приватні, захищені, відкриті члени класу. Інкапсуляція даних класу.	2
4	**Тема 4.** Конструктор, деструктор. Конструктор та деструктор за замовченням. Вказівники та посилання на об’єкти класів. Конструктор копіювання.	2
5	Т**ема 5.** Оператори new та delete. Вказівник this. Ключове слово const при визначенні вказівника та функції-члена.	2
6	**Тема 6.** Спадкування, його види. Передача аргументів в конструктори базових класів.	2
7	**Тема 7.** Перевизначення методів у похідних класах. Поняття поліморфізму. Раннє та пізнє зв’язування.	2
8	**Тема 8.** Віртуальні функції. Поняття абстрактного класу та чисті віртуальні функції. Віртуальні деструктори. Множинне спадкування, його види та правила застосування.	2
9	**Тема 9.** Друзі класу. Друзі-функції та друзі-класи. Вибір між оператором-членом та оператором-другом.	2
10	**Тема 10.** Перевантаження операторів +, -, /, *, =, +=, -=, *=, /=, [ ], ( ), ->, *, постфіксного и префіксного ++ та --).	2
11	**Тема 11.** Перевантаження операторів потокового введення/виведення. Форматування виводу. Обробка виключень.	2
12	**Тема 12.** Ієрархія потокових класів. Класи istream, ostream. Аналіз та управління станом потока.	2
13	**Тема 13.** Стандартні консольні потоки введення/виведення. Введення даних з використанням cin. Методи об’єкта cin. Введення одного символу та строки. Виведення даних з використанням cout. Методи об’єкта cout.	2
14	**Тема 14.** Використання файлів для введення/виведення даних. Класи ofstream та ifstream. Бінарні та текстові файли.	2
15	**Тема 15.** Поняття шаблону. Шаблонні функції. Шаблонні класи. Статичні члени класу та шаблони. Стандартна бібліотека шаблонів. Контейнерні класи. Ітератори. Послідовні контейнери: vector, list, stack, queue. Асоціативні контейнери: map, multimap, set, multiset.	2
16	**Тема 16.** Класи алгоритмів. Алгоритми, що не змінюють послідовність. Алгоритми, що змінюють послідовність.	2
	Усього	32

5. Завдання для самостійної робота

№ з/п	Назва теми	Кількість годин
1	Ознайомитись з основними ідеями об’єктно орієнтованого програмування.	4
2	Дослідження структурного та об’єктно-орієнтованого підходів у програмуванні.	4
3	Вивчення модифікаторів доступу: приватні, захищені, відкриті члени класу.	4

4	Вивчення понять конструкторів та деструкторів.	4
5	Дослідження понять операторів new та delete.	4
6	Спадкування, його види. Передача аргументів в конструктори базових класів.	4
7	Перевизначення методів у похідних класах.	4
8	Віртуальні функції. Поняття абстрактного класу та чисті віртуальні функції.	4
9	Друзі класу. Друзі-функції та друзі-класи. Вибір між оператором-членом та оператором-другом.	3
10	Перевантаження операторів +, -, /, *, =, +=, -=, *=, /=, [ ], ( ), ->, *, постфіксного и префіксного ++ та --).	3
11	Перевантаження операторів потокового введення/виведення. Форматування виводу.	3
12	Ієрархія потокових класів.	3
13	Стандартні консольні потоки введення/виведення. Введення даних з використанням cin.	3
14	Використання файлів для введення/виведення даних.	3
15	Поняття шаблону. Шаблонні функції. Шаблонні класи.	3
16	Класи алгоритмів. Алгоритми, що не змінюють послідовність. Алгоритми, що змінюють послідовність.	3
	Усього	56

6. Індивідуальні завдання

Індивідуальні завдання не передбачені навчальним планом

7. Методи навчання

При викладанні дисципліни використовуються всі основні методи навчання

1. пояснювально – ілюстративний метод з використанням мультимедійних презентацій
2. проблемні методи навчання з застосуванням
   - розв’язання проблемних задач
   - тестових завдань
   - навчальних дискусій
   - активізації самостійного вивчення студентами літератури
3. метод проблемного викладання з постановкою проблеми на початку нової теми
4. частково-пошуковий (евристичний) метод з самостійною або керованою викладачем роботою студентів над комп’ютерними програмами при виконанні поточних завдань
5. Дослідницький метод при самостійному вирішенні студентами завдань контрольних робіт

8. Методи контролю

На лабораторних заняттях – експрес-опитування, виконання лабораторних робіт. Оцінка виставляється за результатами поточного контролю.

Навчальна програма нормативної дисципліни «Об’єктно – орієнтоване програмування» відповідає чинним нормативним документам, рекомендованим Міністерством освіти і науки України як навчальна програма для студентів фізичного спрямування для вищих навчальних закладів. Навчальна дисципліна «Об’єктно – орієнтоване програмування» є складовою циклу професійної підготовки фахівців освітньо–кваліфікаційного рівня «бакалавр».

Зміст програми, об’єм учбових питань дисципліни «Об’єктно – орієнтоване програмування» визначаються потребою загальнонаукової, загально-інженерної та технічної підготовки.

Вивчення дисципліни «Об’єктно – орієнтоване програмування» передбачає викладання лекцій, проведення практичних занять, виконання лабораторних робіт. Підсумковий контроль знань здійснюється на екзамені.

На лекціях викладається теоретичний матеріал, який ілюструється типовими прикладами і задачами за профілем підготовки фахівців з прикладної фізики та наноматеріалів. Викладання лекційного матеріалу має закінчений характер, здійснюється у доступній і наочній формі, містить проблемні ситуації.

Основною метою лабораторних занять є розвиток навичок практичного застосування і закріплення теоретичного матеріалу. При вирішенні задач рекомендується користуватися стандартними прийомами і методиками. На кожному практичному занятті частину учбового часу доцільно використовувати для самостійного розв’язку задач, контролюючи при цьому правильність обраного методу і ходу рішення.

Важливим фактором засвоєння курсу «Об’єктно – орієнтоване програмування» й оволодіння її методами є самостійна робота студентів. Для самостійного відпрацювання розділів і тем дисципліни пропонуються лабораторні роботи, при перевірці виконання яких здійснюється ефективний контроль за рівнем засвоєння матеріалу.

На всіх заняттях підкреслюється роль і значення предмету на конкретних прикладах і задачах прикладного характеру, висвітлюється її зв’язок з іншими дисциплінами і майбутньою професією.

9. Схема нарахування балів

Поточний контроль, самостійна робота	
Розділ 1	
Теми розділів	Кількість балів
Тема 1	6
Тема 2	6
Тема 3	6
Тема 4	6
Тема 5	6
Тема 6	6
Контрольна робота	6
Тема 7	6
Тема 8	6
Тема 9	6
Разом за розділом 1	60
Розділ 2	
Тема 10	5
Тема 11	5
Тема 12	5
Контрольна робота	5
Тема 13	5
Тема 14	5
Тема 15	5
Тема 16	5
Разом за розділом 2	40
Сума	100

Шкала оцінювання

Сума балів за всі види навчальної діяльності протягом семестру	Оцінка	
	для екзамену	для заліку
90 – 100	відмінно	зараховано
70-89	добре	
50-69	задовільно	
1-49	незадовільно	не зараховано

10. Рекомендована література

Основна література

1. Т.А. Павловская, Ю.А.Щупак. С++. Объектно-ориентированное программирование. Практикум. – СПб, Питер, 2005.

Допоміжна література

1. А.Фридман, Л.Кландер, М.Михаэлис, Х.Шилдт. С/С++. Алгоритмы и приемы программирования. – Москва. ООО «Бином-Пресс», 2003 г. – 560 с.
2. Культин Н.Б. С/С++ в задачах и примерах. – СПб: «БХВ–Петербург», 2005. – 288 с.
3. Глушаков С.В., Коваль А.В., Смирнов С.В. Практикум по С++. – Харьков, Фолио, 2006. – 525 с.
4. С. Прата. Язык программирования С++. Лекции и упражнения. – М.: ООО «ДиаСофтЮП», 2005. – 1104 с.
5. Рэй Лишнер. STL. Карманный справочник. Руководство по использованию. – СПб, Питер, 2005.
6. Липпман С., Лажойе Ж. Язык программирования С++. Вводный курс. 3-е изд. – СПб.-М.: Невский проспект – ДМК Пресс, 2001.

11. Посиланная на інформаційні ресурси в Інтернеті, відео-лекції, інше методичне забезпечення

1. Мережа Internet.
2. Бібліотеки ХНУ імені В.Н.Каразіна.