ХАРКІВСЬКА ДЕРЖАВНА АКАДЕМІЯ КУЛЬТУРИ

Кафедра інформаційних технологій

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

ПРОГРАМУВАННЯ ОБ'ЄКТНО-ОРІЄНТОВАНИХ ДОДАТКІВ

(шифр і назва навчальної дисципліни)

перший рівень

(назва рівня вищої освіти)

галузь знань 12 Інформаційні технології

(код і назва спеціальності)

спеціальність 126 Інформаційні системи та технології

(код і назва спеціальності)

кваліфікація бакалавр з інформаційних систем та технологій

Харків 2017

Робоча програма **ПРОГРАМУВАННЯ ОБ'ЄКТНО-ОРІЄНТОВАНИХ ДОДАТКІВ**

РОЗРОБЛЕНО ТА ВНЕСЕНО: Харківська державна академія культури

Укладач:Побіженко І.О.

Робоча програма затверджена на кафедрі інформаційних технологій

Протокол від «06» жовтня 2017 року № 2

Завідувач кафедри інформаційних технологій

(підпис) (Асєєв Г. Г.) (прізвище та ініціали)

«06» жовтня 2017 року

ХДАК 2017 рік

1. Опис навчальної дисципліни

<table>
<tr><th>Найменування показників</th><th>Галузь знань, напрям підготовки, освітньо-кваліфікаційний рівень</th><th>Характеристика навчальної дисципліни</th></tr>
<tr><td rowspan="2">Кількість кредитів – 5</td><td>Галузь знань
12 Інформаційні технології</td><td rowspan="2">Обов’язкова</td></tr>
<tr><td>Напрям підготовки
126 Інформаційні системи та технології</td></tr>
<tr><td>Модулів – 1</td><td rowspan="4">Спеціальність (професійне спрямування):
126 Інформаційні системи та технології</td><td>Рік підготовки:</td></tr>
<tr><td>Змістових модулів – 8</td><td>2-й</td></tr>
<tr><td>Індивідуальне науково-дослідне завдання</td><td>Семестр</td></tr>
<tr><td>Загальна кількість годин – 150</td><td>3-й</td></tr>
<tr><td rowspan="11">Тижневих годин для денної форми навчання:
аудиторних – 4 год;
самостійної роботи студента – 4,5 год.</td><td rowspan="11">Освітньо-кваліфікаційний рівень:
бакалавр</td><td>Лекції</td></tr>
<tr><td>22 год.</td></tr>
<tr><td>Семінарські</td></tr>
<tr><td>-</td></tr>
<tr><td>Практичні</td></tr>
<tr><td>46 год.</td></tr>
<tr><td>Самостійна робота</td></tr>
<tr><td>82 год.</td></tr>
<tr><td>Індивідуальні завдання:</td></tr>
<tr><td>-</td></tr>
<tr><td>Вид контролю:</td></tr>
<tr><td>залік</td></tr>
</table>

Примітка.
Співвідношення кількості годин аудиторних занять до самостійної роботи становить 68:82.

2. Мета та завдання навчальної дисципліни

«Інтелектуальний аналіз даних» – навчальна дисципліна, яка в контексті новітніх технологій обробки даних посилює теоретичну та практичну професійну підготовку бакалаврів з інформаційних технологій.

Мета дисципліни: навчання студентів самостійно будувати програми середньої складності мовою С++ з використанням структурно-модульного та об'єктно-орієнтованого методу програмування.

Задачі дисципліни:

- є ознайомлення з основами об'єктно-орієнтованого програмування,
- отримання навичок використання класів,
- механізмів наслідування,
- інкапсуляції,
- поліморфізму.

Компетентності, якими повинен оволодіти здобувач	Програмні результати навчання
ЗК2. Здатність застосовувати знання у практичних ситуаціях. ЗК3. Знання та розуміння предметної області та професійної діяльності. ЗК5. Навички використання інформаційних і комунікаційних технологій ФК1. Здатність проводити аналіз об'єкту проектування та предметної області ФК3. Здатність до проектування системного, комунікаційного і прикладного програмного забезпечення, технічних засобів та комунікаційних й інформаційних технологій, мереж та систем. ФК4. Здатність розробляти засоби реалізації ІСТ та ІСДС (методичні, інформаційні, алгоритмічні, технічні й програмні). ФК5. Здатність розробляти, налагоджувати та вдосконалювати програмне забезпечення комп'ютерно-інтегрованих систем. ФК6. Здатність використовувати сучасні технології проектування в розробці алгоритмічного та програ-	РН2. Здатність використовувати знання з основних фундаментальних, природничих та загально-інженерних дисциплін, а також системного аналізу, моделювання систем, теорії алгоритмів та дискретної математики при розв'язанні типових задач, проектуванні та використанні ІСТ та ІСДС. РН3. Здатність використовувати: базові знання інформатики й сучасних ІСТ, навички програмування та застосування програмних засобів, безпечної роботи в комп'ютерних мережах, уміння створювати бази даних, використовувати інтернет-ресурси та демонструвати уміння розробляти алгоритми та комп'ютерні програми на мовах високого рівня та технологій об'єктно-орієнтованого програмування для реалізації задач проектування та використання ІСТ та ІСДС. РН4. Здатність проводити системний аналіз об'єктів проектування та обґрунтовувати вибір структури,

много забезпечення ІСТ та ІСДС.	алгоритмів та способів циркулювання інформації в ІСТ та ІСДС. РН6. Здатність демонструвати знання сучасного рівня та новітніх технологій ІСТ та ІСДС з метою їх запровадження у професійної діяльності РН9. Здатність демонструвати знання і практичні навички програмування та використання прикладних і спеціалізованих комп'ютерних систем та середовищ для розв'язання задач проектування

В результаті вивчення навчальної дисципліни студент повинен:

знати:

- передумови і історію виникнення об'єктно-орієнтованого підходу
- поняття класу та об'єкту
- методи об'єктно-орієнтованого аналізу і проектування
- основні синтаксичні конструкції мови С та С++
- найважливіші класи та функції стандартних бібліотек мови С та С++, що включає управління пам'яттю, файлами, консоллю, математичні обчислення
- основні методології розробки програмного забезпечення, особливості їх застосування для об'єктно-орієнтованих програм
- поняття патерну проектування, види паттернів, особливості їх використання;

вміти:

- – складати програми мовами С та С++
- застосувати грамотний стиль програмування, що включає функціональну та об'єктно-орієнтовану декомпозицію
- побудувати структурований алгоритм обробки базових структур даних із застосуванням об'єктно-орієнтованого підходу
- програмно реалізувати поняття у вигляді класу, створити об'єкти даного класу
- виділити загальні методи обробки даних у окремі класи та методи застосувати принципи інкапсуляції, обмеження доступу та поліморфізму для побудови програми середньої складності.

мати навички:

представлення результатів у зручному для користувача вигляді.

Міждисциплінарні зв'язки: для засвоєння матеріалу використовуються знання, отримані при вивченні курсів «Вища математика», «Дискретна математика», «Теорія ймовірностей та математична статистика», «Алгоритмізація

і структури даних», «Системи та технології організації сховищ даних», «Технології обчислювань та збереження інформації».

3. Програма навчальної дисципліни

Змістовий модуль 1. Об'єктно-орієнтоване програмування.

Тема 1. Основи об'єктно-орієнтованого програмування. Вступ до мови С++. Основні особливості мови, не пов'язані з об'єктно-орієнтованим програмуванням. Поняття об'єкту та його порівняння зі структурами даних та алгоритмів в мові С. Оператори new та delete для керування пам'яттю.

Тема 2. Вбудовані (inline) функції. Перевантаження імен функцій. Аргументи по замовчуванню. Однорядкові коментарі. Глобальні константи. Скорочена форма при використанні означень struct, enum, union.

Змістовий модуль 2. Інкапсуляція та приховання інформації.

Тема 1. Поняття інкапсуляції. Поняття про захист внутрішніх даних об'єкту. Метод як канал доступу до внутрішніх даних. Модель об'єкта як чорного ящика. Клас та екземпляр.

Тема 2. Найпростіший синтаксис означення класу. Специфікатори доступу public, protected та private. Означення методів класу за межами класу

Змістовий модуль 3. Розподіл поведінки та реалізації.

Тема 1. Конструктори, їх роль та призначення. Конструктори по замовчуванню та з параметрами. Особливості ініціалізації членів-даних, винесення перед тілом конструктора. Динамічне виділення пам'яті для об'єкту: виклик конструктора з оператору new. Деструктори.

Змістовий модуль 4. Класи та підкласи.

Тема 1. Відношення клас-підклас та його зв'язок з відношенням абстрактне-конкретне. Сумісність типів знизу вгору.

Тема 2. Дружні функції, окремі класи та класи в цілому, обхід механізмів захисту членів класу. Переваги та недоліки використання механізму дружності в програмах з об'єктно-орієнтованою композицією.

Змістовий модуль 5. Успадкування (перевизначення, динамічне зв'язування)

Тема 1. Наслідування. Механізм наслідування членів-даних та методів.

Тема 2. Перевантаження операторів функціями та методами.

Змістовий модуль 6. Поліморфізм (поліморфізм підтипів і успадкування)

Тема 1. Поліморфізм та віртуальні функції. Поняття оголошеного та фактичного типу. Механізм виклику віртуальної функції.

Змістовий модуль 7. Ієрархія класів

Тема 1. Ієрархія класів. Приклади. Множинне наслідування та його проблеми.

Тема 2. Особливості виклику конструктора базового класу з конструктору надкласу. Порядок виклику конструкторів та деструкторів для об’єктів похідних класів.

Змістовий модуль 8. Класи колекцій і протоколи ітерації

Тема 1. Класи потоків введення-виведення. Управління форматом, модифікатори.

Тема 2. Шаблони функцій та класів.

Тема 3. Класи колекцій стандартної бібліотеки.

Тема 4. Обробка виняткових ситуацій. Поняття про виняткову ситуацію. Обґрунтування вимог до механізму обробки винятків. Оператори try, throw і catch. Порядок генерування та перехоплення виняткової ситуації.

4. Структура навчальної дисципліни

Назви змістових модулів та тем	Кількість годин				
	усього	у тому числі			
		лекцій	пз	сем	сам. р.
Змістовий модуль 1. **Об'єктно-орієнтоване програмування**					
Основи об'єктно-орієнтованого програмування	9	2	2		5
Вбудовані (inline) функції	9	2	2		5
Разом за змістовим модулем 1	*18*	*4*	*4*		*10*
Змістовий модуль 2. **Інкапсуляція та приховання інформації**					
Поняття інкапсуляції+10+6+	9	2	2		5
Найпростіший синтаксис означення класу	9	2	2		5
Разом за змістовим модулем 2	*18*	*4*	*4*		*10*
Змістовий модуль 3. **Розподіл поведінки та реалізації**					
Конструктори, їх роль та призначення	10	2	4		6
Разом за змістовим модулем 3	*12*	*2*	*4*		*6*
Змістовий модуль 4. **Класи та підкласи**					
Відношення клас-підклас та його зв'язок з відношенням абстрактне-конкретне	9	1	3		5
Дружні функції, окремі класи та класи в цілому, обхід механізмів захисту членів класу	9	1	3		5
Разом за змістовим модулем 4	*18*	*2*	*6*		*10*
Змістовий модуль 5. **Успадкування (перевизначення, динамічне зв'язування)**					
. Наслідування	*9*	*1*	*3*		*5*
Перевантаження операторів функціями та методами	*11*	*1*	*5*		*5*
Разом за змістовим мод+улем 5	*20*	*2*	*8*		*10*
Змістовий модуль 6. **Поліморфізм (поліморфізм підтипів і успадкування)**					
Поліморфізм та віртуальні функції	*10*	*2*	*2*		*6*
Разом за змістовим модулем 6	*10*	*2*	*2*		*6*
*Змістовий модуль 7***. Ієрархія класів**					
Ієрархія класів	*9*	*1*	*3*		*5*
Особливості виклику конструктора базового класу з конструктору надкласу	*9*	*1*	*3*		*5*
Разом за змістовим модулем 7	*18*	*2*	*6*		*10*
*Змістовий модуль 8***. Класи колекцій і протоколи ітерації**					
Класи потоків введення-	*9*	*1*	*3*		*5*

Назви змістових модулів та тем	Кількість годин				
	усього	у тому числі			
		лекцій	пз	сем	сам. р.
виведення					
Шаблони функцій та класів	*9*	*1*	*3*		*5*
Класи колекцій стандартної бібліотеки	*9*	*1*	*3*		*5*
Обробка виняткових ситуацій	*9*	*1*	*3*		*5*
Разом за змістовим модулем 7	*36*	*4*	*12*		*20*
Усього годин	**150**	**22**	**46**	**-**	**82**

5. Теми семінарських занять

Не передбачено

6. Теми практичних занять

Не передбачено

7. Теми лабораторних занять

№	Тема практичного заняття	Кількість годин
1.	Середовище для розробки програм C++ - середовище для розробки програм на мові C++. Розробка структур як простих класів.	3
2.	Визначення класу. Область дії класу та доступ до елементів класу. Управління доступом до елементів класу. Стандартні класи C++ для введення/виведення даних. Розробка власних класів **(**complex, date_).	3
3.	Створення простих конструкторів для ініціалізації елементів-змінних класів та виділення пам'яті. Використання конструкторів з аргументами за замовченням. Розробка програм з використанням власних класів – реалізація «арифметики» для комплексних чисел	3
4.	Створення простих деструкторів для звільнення пам'яті. Присвоєння за замовченням. Розробка програм з використанням власних класів – реалізація «арифметики» для комплексних чисел	3
5.	Розробка програм з використанням композиції класів – побудова двох класів (date_, student), один з яких використовує інший клас як вкладений.	3
6.	Розробка програм із використанням дружньої функції – розробка функції presentTime як дружньої для класу date_. Використання операцій виділення пам'яті ***new, delete***.	3
7.	Розробка програм із використанням дружнього класу – розробка класу decant як дружнього для класу student.	3
8.	Використання основних принципів перевантаження операцій. Розробка програми, яка дозволяє перевантажувати одномісні операції.	3
9	Розробка програми, яка реалізує «арифметику» комплексних чисел і використовує перевантаження операцій додавання та віднімання.	3

№	Тема практичного заняття	Кількість годин
10	Розробка базового класу person та реалізація функцій для цього класу.	3
11	Розробка похідного класу student від базового класу person та реалізація функцій для класу student.	3
12	Розробка базового класу state та реалізація функцій для цього класу.	3
13	Розробка похідного класу decant від базового класу state та реалізація функцій для цього класу	3
14	Використання віртуальних функцій. Розробка похідного класу teacher від базового класу person. Введення віртуальної функції в базовий клас person.	3
15	Застосування поліморфізму при проектуванні класів. Розробка програми по введенню даних щодо викладачів та студентів.	4
	Разом	***46***

8. Самостійна робота

№ теми	Тема самостійної роботи	Кількість годин
1.	Середовище для розробки програм C++ - середовище для розробки програм на мові C++. Розробка структур як простих класів.	6
2.	Визначення класу. Область дії класу та доступ до елементів класу. Управління доступом до елементів класу. Стандартні класи C++ для введення/виведення даних. Розробка власних класів **(**complex, date_).	5
3.	Створення простих конструкторів для ініціалізації елементів-змінних класів та виділення пам'яті. Використання конструкторів з аргументами за замовченням. Розробка програм з використанням власних класів – реалізація «арифметики» для комплексних чисел	6
4.	Створення простих деструкторів для звільнення пам'яті. Присвоєння за замовченням. Розробка програм з використанням власних класів – реалізація «арифметики» для комплексних чисел	5
5.	Розробка програм з використанням композиції класів – побудова двох класів (date_, student), один з яких використовує інший клас як вкладений.	6
6.	Розробка програм із використанням дружньої функції – розробка функції presentTime як дружньої для класу date_. Використання операцій виділення пам'яті ***new, delete***.	5
7.	Розробка програм із використанням дружнього класу – розробка класу decant як дружнього для класу student.	6
8.	Використання основних принципів перевантаження операцій. Розробка програми, яка дозволяє перевантажувати одномісні операції.	5
9	Розробка програми, яка реалізує «арифметику» комплексних чисел і використовує перевантаження операцій додавання та віднімання.	6
10	Розробка базового класу person та реалізація функцій для цього класу.	5

№ теми	Тема самостійної роботи	Кількість годин
11	Розробка похідного класу student від базового класу person та реалізація функцій для класу student.	6
12	Розробка базового класу state та реалізація функцій для цього класу.	5
13	Розробка похідного класу decant від базового класу state та реалізація функцій для цього класу	6
14	Використання віртуальних функцій. Розробка похідного класу teacher від базового класу person. Введення віртуальної функції в базовий клас person.	5
15	Застосування поліморфізму при проектуванні класів. Розробка програми по введенню даних щодо викладачів та студентів.	5
	Разом	**82**

9. Індивідуальні завдання

Не передбачено

10. Методи навчання

Методи навчання, що використовуються у процесі лекційних занять:

– лекція з елементами пояснення;
– лекція-бесіда;
– лекція-дискусія;
– ілюстрація наочних матеріалів;
– пояснення.

Методи навчання, що використовуються під час практичних занять:

– виконання вправ та завдань;
– застосування комп'ютерної техніки та прикладних програм для вирішення задач;
– розробка власних прикладних програм;
– сумісна робота над проектом;
– самостійна робота.

11. Методи контролю

Підсумковий контроль. Для контролю засвоєння дисципліни навчальним планом передбачений залік. Проведення підсумкової атестації і отримання на ній позитивної оцінки включає:

а) оцінку проміжної атестації (результати модуля),
б) оцінку відвідуваності занять і активність в аудиторії;
в) оцінку виконання усіх завдань самостійної роботи.

Поточний контроль. Для поточного контролю використовуються результати практичних занять.

12. Розподіл балів, які отримують студенти

Поточне тестування та самостійна робота								Сума
Т1	Т2	Т3	Т4	Т5	Т6	Т7	Т8	
12	12	12	12	12	12	12	13	100

Т1, Т2, …. Т12 – теми змістових модулів.

Шкала оцінювання: національна та ECTS

Бали	**Оцінка**		
	Шкала	**Національна шкала**	
		Іспит	**Залік**
90–100	A	Відмінно	Зараховано
82–89	B	Добре	
74–81	C		
67–73	D	Задовільно	
60–66	E		
35–59	FX	Незадовільно з можливістю повторного складання	Не зараховано з можливістю повторного складання
1–34	F	Незадовільно з обов'язковим повторним курсом	Не зараховано з обов'язковим повторним курсом

13. Методичне забезпечення

№ з/п	Найменування методичних матеріалів	Рік видання	наявність в бібл., примірн	Ел. варіант	Код
1. Підручники					
1.				+	