**Харківський національний економічний університет**

**імені Семена Кузнеця**

**ЗВІТ**

**З ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 10**

**за дисципліною: *“*Технології програмування**”

**на тему: “Реалізація концепції ООП”**

**Варіант: 4**

**Виконав: студент факультету Інформаційних технологій**

**2 курсу, спец. Кібербезпека,**

**групи 6.04.125.010.21.2**

**Бойко Вадим Віталійович**

**Перевірив:**

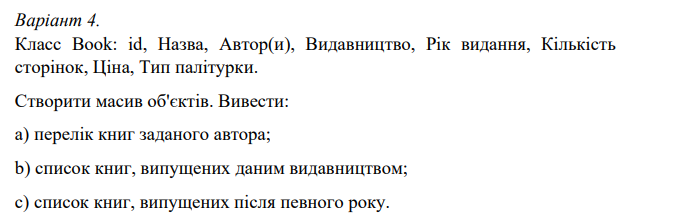
**Венгріна Олена Сергіївна**

**ХНЕУ ім. С. Кузнеця**

**2022**

**Мета:** набуття навичок створення програм на основі технологій об’єктно-орієнтованого програмування: створення класів, робота з об’єктами класів.

**Завданя:** Створити класи опису сутності, специфікації яких наведено нижче. Визначити конструктор, сеттери та геттери, перевизначить метод рядкового представлення. Поле id повинно генеруватися автоматично при створенні об’єктів й представляти собою натуральний ряд чисел. Створити клас, що містить масив об'єктів сутностей, у якому визначити методи, що повертають результат за певним критерієм вибору даних.

****

Я написав наступне:

Клас Книга

*from* random *import* randint  
  
*class* Book:  
 *def \_\_init\_\_*(self, *name*, *authors*, *publisher*, *year\_of\_publishing*, *page\_count*, *price*, *paliturka\_type*):  
 self.\_id = randint(1, 1000000)  
 self.\_name = *name* self.\_authors = *authors* self.\_publisher = *publisher* self.\_year\_of\_publishing = *year\_of\_publishing* self.\_page\_count = *page\_count* self.\_price = *price* self.\_paliturka\_type = *paliturka\_type  
  
 def* getId(self):  
 *return* self.\_id  
  
 *def* getName(self):  
 *return* self.\_name  
  
 *def* getAuthors(self):  
 *if* isinstance(self.\_authors, str):  
 *return* self.\_authors  
 *return* list.copy(self.\_authors)  
  
 *def* getPublisher(self):  
 *return* self.\_publisher  
  
 *def* getYearOfPublishing(self):  
 *return* self.\_year\_of\_publishing  
  
 *def* getPageCount(self):  
 *return* self.\_page\_count  
  
 *def* getPrice(self):  
 *return* self.\_price  
  
 *def* getPaliturkType(self):  
 *return* self.\_paliturka\_type

Клас книжна полиця

*class* BookShelf:  
 *def \_\_init\_\_*(self, *books* = []):  
 self.books = *books  
  
 def* addBook(self, *book*):  
 self.books.append(*book*)  
  
 *def* getBooksNameByAuthorName(self, *author\_name*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if author\_name in* book.getAuthors() *or author\_name* == book.getAuthors():  
 books\_names.append(book.getName())  
 *return* books\_names  
  
 *def* getBooksNameByPublisherName(self, *publisher\_name*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if publisher\_name* == book.getPublisher():  
 books\_names.append(book.getName())  
 *return* books\_names  
  
 *def* getBooksYearlierThanYear(self, *year*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if* book.getYearOfPublishing() >= *year*:  
 books\_names.append(book.getName())  
 *return* books\_names

Створення книг

book1 = Book('Book1', 'Author1', 'Publisher1', 2000, 100, 50, '1')  
book2 = Book('Book2', 'Author2', 'Publisher2', 2002, 102, 52, '2')  
book3 = Book('Book3', 'Author3', 'Publisher3', 2003, 103, 53, '3')  
book4 = Book('Book4', ['Author1'], 'Publisher1', 2004, 104, 54, '1')  
book5 = Book('Book5', ['Author2'], 'Publisher2', 2005, 105, 55, '2')  
book6 = Book('Book6', ['Author3'], 'Publisher3', 2006, 106, 56, '3')  
book7 = Book('Book7', ['Author1', 'Author2'], 'Publisher1', 2007, 107, 57, '4')  
book8 = Book('Book8', ['Author2', 'Author3'], 'Publisher2', 2008, 108, 58, '1')  
book9 = Book('Book9', ['Author3', 'Author1'], 'Publisher3', 2009, 109, 59, '4')

Створення книжної полички

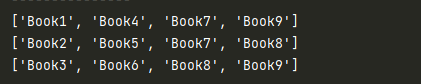
book\_shelf = BookShelf([book1, book2, book3, book4])

додавання книг до книжної поличкиbook\_shelf.addBook(book5)  
book\_shelf.addBook(book6)  
book\_shelf.addBook(book7)  
book\_shelf.addBook(book8)  
book\_shelf.addBook(book9)

Перелік книг по автору

print(book\_shelf.getBooksNameByAuthorName('Author1'))  
print(book\_shelf.getBooksNameByAuthorName('Author2'))  
print(book\_shelf.getBooksNameByAuthorName('Author3'))

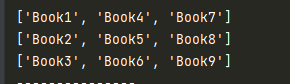
Результат



Перелік книг по видавництву

print(book\_shelf.getBooksNameByPublisherName('Publisher1'))  
print(book\_shelf.getBooksNameByPublisherName('Publisher2'))  
print(book\_shelf.getBooksNameByPublisherName('Publisher3'))

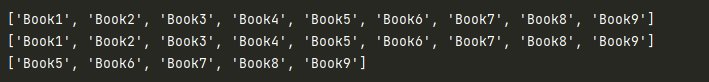
Результат



Перелік книг випущених після певного року

print(book\_shelf.getBooksYearlierThanYear(1990))  
print(book\_shelf.getBooksYearlierThanYear(2000))  
print(book\_shelf.getBooksYearlierThanYear(2005))

Результат



Код всієї програми

*from* random *import* randint  
  
*class* Book:  
 *def \_\_init\_\_*(self, *name*, *authors*, *publisher*, *year\_of\_publishing*, *page\_count*, *price*, *paliturka\_type*):  
 self.\_id = randint(1, 1000000)  
 self.\_name = *name* self.\_authors = *authors* self.\_publisher = *publisher* self.\_year\_of\_publishing = *year\_of\_publishing* self.\_page\_count = *page\_count* self.\_price = *price* self.\_paliturka\_type = *paliturka\_type  
  
 def* getId(self):  
 *return* self.\_id  
  
 *def* getName(self):  
 *return* self.\_name  
  
 *def* getAuthors(self):  
 *if* isinstance(self.\_authors, str):  
 *return* self.\_authors  
 *return* list.copy(self.\_authors)  
  
 *def* getPublisher(self):  
 *return* self.\_publisher  
  
 *def* getYearOfPublishing(self):  
 *return* self.\_year\_of\_publishing  
  
 *def* getPageCount(self):  
 *return* self.\_page\_count  
  
 *def* getPrice(self):  
 *return* self.\_price  
  
 *def* getPaliturkType(self):  
 *return* self.\_paliturka\_type  
  
  
*class* BookShelf:  
 *def \_\_init\_\_*(self, *books* = []):  
 self.books = *books  
  
 def* addBook(self, *book*):  
 self.books.append(*book*)  
  
 *def* getBooksNameByAuthorName(self, *author\_name*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if author\_name in* book.getAuthors() *or author\_name* == book.getAuthors():  
 books\_names.append(book.getName())  
 *return* books\_names  
  
 *def* getBooksNameByPublisherName(self, *publisher\_name*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if publisher\_name* == book.getPublisher():  
 books\_names.append(book.getName())  
 *return* books\_names  
  
 *def* getBooksYearlierThanYear(self, *year*):  
 books\_names = []  
 *for* book *in* self.books:  
 *if* book.getYearOfPublishing() >= *year*:  
 books\_names.append(book.getName())  
 *return* books\_names  
  
book1 = Book('Book1', 'Author1', 'Publisher1', 2000, 100, 50, '1')  
book2 = Book('Book2', 'Author2', 'Publisher2', 2002, 102, 52, '2')  
book3 = Book('Book3', 'Author3', 'Publisher3', 2003, 103, 53, '3')  
book4 = Book('Book4', ['Author1'], 'Publisher1', 2004, 104, 54, '1')  
book5 = Book('Book5', ['Author2'], 'Publisher2', 2005, 105, 55, '2')  
book6 = Book('Book6', ['Author3'], 'Publisher3', 2006, 106, 56, '3')  
book7 = Book('Book7', ['Author1', 'Author2'], 'Publisher1', 2007, 107, 57, '4')  
book8 = Book('Book8', ['Author2', 'Author3'], 'Publisher2', 2008, 108, 58, '1')  
book9 = Book('Book9', ['Author3', 'Author1'], 'Publisher3', 2009, 109, 59, '4')  
  
book\_shelf = BookShelf([book1, book2, book3, book4])  
  
book\_shelf.addBook(book5)  
book\_shelf.addBook(book6)  
book\_shelf.addBook(book7)  
book\_shelf.addBook(book8)  
book\_shelf.addBook(book9)  
  
print('-' \* 15)  
  
print(book\_shelf.getBooksNameByAuthorName('Author1'))  
print(book\_shelf.getBooksNameByAuthorName('Author2'))  
print(book\_shelf.getBooksNameByAuthorName('Author3'))  
  
print('-' \* 15)  
  
print(book\_shelf.getBooksNameByPublisherName('Publisher1'))  
print(book\_shelf.getBooksNameByPublisherName('Publisher2'))  
print(book\_shelf.getBooksNameByPublisherName('Publisher3'))  
  
print('-' \* 15)  
  
print(book\_shelf.getBooksYearlierThanYear(1990))  
print(book\_shelf.getBooksYearlierThanYear(2000))  
print(book\_shelf.getBooksYearlierThanYear(2005))