Министерство Образования Республики Беларусь
УО Брестский Государственный Технический Университет
Кафедра интеллектуальных информационных технологий

Лабораторная работа №6.
"Наследование и виртуальные функции”

Выполнил:
Студент 2-го курса
Группы АС-53
Вожейко М.В.
Проверил:
Давидюк Ю. И.

Брест, 2020

1. Цель. Получить практические навыки создания иерархии классов и использования статических компонентов класса.

2. Постановка задачи (Вариант 6)

Написать программу, в которой создается иерархия классов. Включить полиморфные объекты в связанный список, используя статические компоненты класса. Показать использование виртуальных функций.

журнал, книга, печатное издание, учебник;;

Классы: Book, Magazine, PrintEd, Textbook
Конструкторы:

- Пустой
- С параметрами
- Копирования

Деструктор.
Виртуальные фукнции:

- Добавления в список
- Вывода информации

3. Определение пользовательских классов с комментариями.

```
	//Базовый класс
class PrintEd  // базовый класс
{
public:
	static  PrintEd* begin;   //указатель на начало списка
	PrintEd* next = NULL;
	static void ShowList()  //список
	{
		PrintEd* p = begin;
		while (p)
		{
			p->show();
			p = p->next;
		}
	}
	PrintEd()  //без параметров
	{
		author = new char[81];
	}
	PrintEd(const char* AUTHOR, int YEAROFPUBL)  //с параметрами
	{
		// выделение памяти для name. размер выделяемой памяти = длина строки NAME
		author = new char[strlen(AUTHOR) + 1];
		strcpy(author, AUTHOR);
		yearOfPubl = YEAROFPUBL;
	}
	~PrintEd()  // деструктор
	{
		cout << "PrintEd object deleted" << endl;
	}
	virtual void show() = 0;  //Чистая виртуальная функция
	virtual void input() = 0;

protected:
	char* author;
	int yearOfPubl;
      };
class Book :public PrintEd   // производный класс
{
public:
	Book() : PrintEd() {}   //без параметров
	Book(const char* AUTHOR, int YEAROFPUBL, int EDITION, string BOOKNAME, bool AddToList = false)
:PrintEd(AUTHOR, YEAROFPUBL) //с параметрами
	{
		if (AddToList)
		{
			PrintEd* p = begin;
			while (p->next)
			{
				p = p->next;
			}
			p->next = this;
		}
		edition = EDITION;
		bookName = BOOKNAME;
	}
	void show()
	{
		cout << "\nКласс: Книга";
		cout << "\nАвтор: " << author;
		cout << "\nГод издания: " << yearOfPubl;
		cout << "\nТираж: " << edition;
```

```
		cout << "\nНазвание книги: " << bookName;
		cout << "\n";
	}
	void input()
	{
		cout << "\nАвтор: ";
		cin >> author;
		cout << "\nГод издания: ";
		cin >> yearOfPubl;
		cout << "\nТираж: ";
		cin >> edition;
		cout << "\nНазвание книги: ";
		cin >> bookName;
		cout << "\n";
	}
	~Book()  // деструктор
	{
		cout << "Book object deleted" << endl;
	}
protected:
	int edition;
	string bookName;
};
class Textbook :public Book // производный класс
{
public:
	Textbook() : Book() {}

	Textbook(const char* AUTHOR, int YEAROFPUBL, int EDITION, string BOOKNAME,string SUBJECT, bool
AddToList = false) :Book(AUTHOR, YEAROFPUBL, EDITION, BOOKNAME)
	{
		if (AddToList)
		{
			PrintEd* p = begin;
			while (p->next)
			{
				p = p->next;
			}
			p->next = this;
		}
		subject = SUBJECT;
	}
	void show()
	{
		cout << "\nКласс: Учебник";
		cout << "\nАвтор: " << author;
		cout << "\nГод издания: " << yearOfPubl;
		cout << "\nТираж: " << edition;
		cout << "\nНазвание книги: " << bookName;
		cout << "\nПредмет: " << subject;
		cout << "\n";
	}
	void input()
	{
		cout << "\nАвтор: ";
		cin >> author;
		cout << "\nГод издания: ";
		cin >> yearOfPubl;
		cout << "\nТираж: ";
		cin >> edition;
		cout << "\nНазвание книги: ";
		cin >> bookName;
		cout << "\nПредмет: ";
		cin >> subject;
		cout << "\n";
```

```
	}
	~Textbook()  // деструктор
	{
		cout << "Textbook object deleted" << endl;
	}
protected:
	string subject;
};
class Magazine :public PrintEd  // производный класс
{
public:
	Magazine() : PrintEd() {}

	Magazine(const char* AUTHOR, int YEAROFPUBL, string PUBLISHER, bool AddToList = false)
:PrintEd(AUTHOR, YEAROFPUBL)
	{
		if (AddToList)
		{
			PrintEd* p = begin;
			while (p->next)
			{
				p = p->next;
			}
			p->next = this;
		}
		publisher = PUBLISHER;
	}
	void show()
	{
		cout << "\nКласс: Журнал";
		cout << "\nАвтор: " << author;
		cout << "\nГод издания: " << yearOfPubl;
		cout << "\nИздатель: " << publisher;
		cout << "\n";
	}
	void input()
	{
		cout << "\nАвтор: ";
		cin >> author;
		cout << "\nГод издания: ";
		cin >> yearOfPubl;
		cout << "\nИздатель: ";
		cin >> publisher;
		cout << "\n";
	}
	~Magazine()  // деструктор
	{
		cout << "Magazine object deleted" << endl;
	}
protected:
	string publisher;
};
```

Реализация конструкторв с параметрами

и
деструктора.

- Для класса PrintEd:

```
 PrintEd()  //без параметров
{
	author = new char[81];
}
PrintEd(const char* AUTHOR, int YEAROFPUBL)  //с параметрами
{
	// выделение памяти для name. размер выделяемой памяти = длина строки NAME
	author = new char[strlen(AUTHOR) + 1];
	strcpy(author, AUTHOR);
	yearOfPubl = YEAROFPUBL;    }
```

- Для класса Book:

```
Book() : PrintEd() {}   //без параметров
Book(const char* AUTHOR, int YEAROFPUBL, int EDITION, string BOOKNAME, bool AddToList = false)
:PrintEd(AUTHOR, YEAROFPUBL) //с параметрами
{
if (AddToList)
{
PrintEd* p = begin;
while (p->next)
{
p = p->next;
}
p->next = this;
}
edition = EDITION;
bookName = BOOKNAME;
            }
```

- Для класса Magazine:

```
Magazine() : PrintEd() {}

	Magazine(const char* AUTHOR, int YEAROFPUBL, string PUBLISHER, bool AddToList = false)
:PrintEd(AUTHOR, YEAROFPUBL)
	{
		if (AddToList)
		{
			PrintEd* p = begin;
			while (p->next)
			{
				p = p->next;
			}
			p->next = this;
		}
		publisher = PUBLISHER;
	}
```

- Для класса Textbook :

```
Textbook() : Book() {}

	Textbook(const char* AUTHOR, int YEAROFPUBL, int EDITION, string BOOKNAME,string SUBJECT, bool AddToList = false) :Book(AUTHOR, YEAROFPUBL, EDITION, BOOKNAME)
	{
		if (AddToList)
		{
			PrintEd* p = begin;
			while (p->next)
			{
				p = p->next;
			}
			p->next = this;
		}
		subject = SUBJECT;
	}
```

4. Реализация метода для просмотра списка.

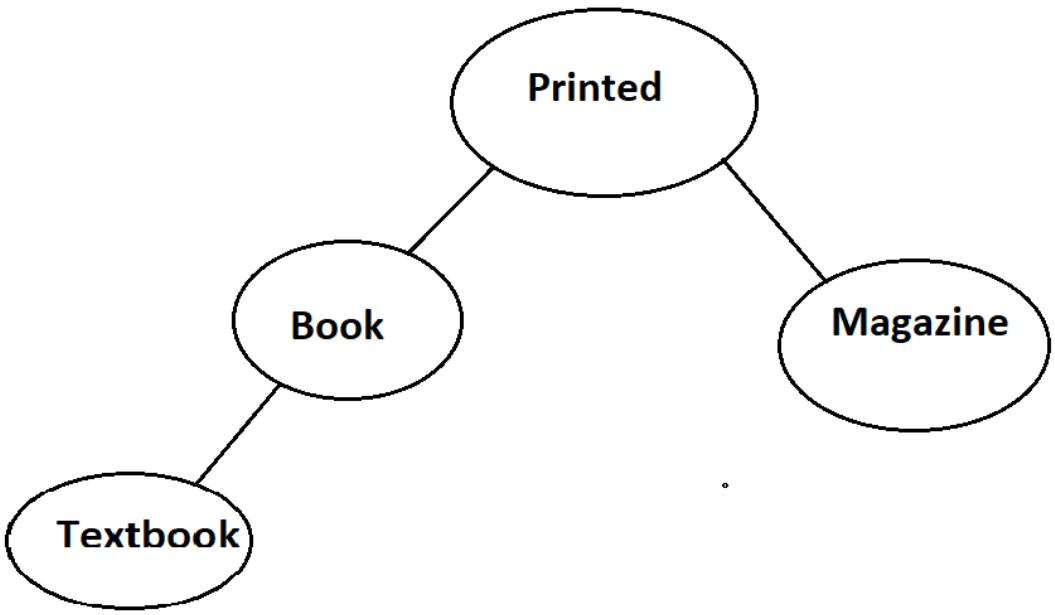
```
void show()
  {
        cout << "\nКласс: Учебник";
        cout << "\nАвтор: " << author;
        cout << "\nГод издания: " << yearOfPubl;
        cout << "\nТираж: " << edition;
        cout << "\nНазвание книги: " << bookName;
        cout << "\nПредмет: " << subject;
        cout << "\n";
  }
```

5. Листинг демонстрационной программы.

```
int main()
{
       SetConsoleCP(1251);
       SetConsoleOutputCP(1251);
       Book* a1 = new Book;
       Magazine* a2 = new Magazine;
       Textbook* a3 = new Textbook;
       a1->input();
       a2->input();
       a3->input();
       cout << "---------------------------------\n";
       PrintEd::begin = a1;
       a1->next = a2;
       a2->next = a3;
       Textbook* x4 = new Textbook("Novik Vadim", 2020, 5000, "how to code in c++", "programming",
true); // Создание объекта класса
       PrintEd::ShowList();
       return 0;

  }
```

8.Диаграмма классов



Объяснение необходимости виртуальных функций. Следует показать, какие результаты будут в случае виртуальных и не виртуальных функций.

При наследовании бывает необходимо, чтобы поведение некоторых методов базового класса и классов-наследников различалось, именно для этого и требуется наличие виртуальных функций `virtual void Show() = 0;virtual void Add() = 0;`

В данном коде, в случае отсутствия виртуальной функции нельзя будет переопределить поведение.

Вывод: Получил практические навыки реализации классов на C++.