Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Описать класс cl_parent объекта, в котором следующий состав элементов:

В закрытом разделе:

− одно свойство целого типа;

− метод, с одним целочисленным параметром. который меняет значение свойства в закрытом разделе на удвоенное значение параметра.

В открытом разделе:

− одно свойство целого типа;

− параметризированный конструктор, с двумя целочисленными параметрами, который устанавливает значения свойств в закрытом и открытом разделе. Значение закрытого свойства меняется посредством вызова метода из закрытого раздела;

− метод с двумя целочисленными параметрами, который устанавливает значения свойств в закрытом и открытом разделе. Значение закрытого свойства меняется посредством вызова метода из закрытого раздела;

− метод, который выводит на экран значение обоих свойств. Сперва значение закрытого свойства, потом значение открытого свойства.

Назовем объект данного класса родительским. Соответственно его класс родительским классом.

На базе родительского объекта сконструируем производный объект. Производный объект должен сохранить открытый доступ к открытым элементам родительского класса. Он должен иметь следующие собственные элементы:

В закрытом разделе:

− одно свойство целого типа, наименование которого совпадает с наименование закрытого свойства родительского объекта;

В открытом разделе:

− одно свойство целого типа, наименование которого совпадает с наименование открытого свойства родительского объекта;

− параметризированный конструктор, с двумя целочисленными параметрами, который устанавливает значения свойств в закрытом и открытом разделе;

− метод с двумя целочисленными параметрами, который устанавливает значения свойств в закрытом и открытом разделе. Наименование метода совпадает с наименованием аналогичного метода родительского объекта;

− метод, который выводит на экран значение обоих свойств. Сперва значение закрытого свойства, потом значение открытого свойства. Наименование метода совпадает с наименованием аналогичного метода родительского объекта.

Разработать производный класс используя класс cl_parent в качестве родительского.

В основной функции реализовать алгоритм:

1. Ввод значения двух целочисленных переменных.

2. Создать объект производного класса используя целочисленных переменных в конструкторе в качестве аргументов в последовательности, как им были присвоены значения. Первый аргумент содержит значение для свойства закрытого раздела/,/ второй для свойства открытого раздела.

3. Вывод значений свойств родительского объекта.

4. Вывод значений свойств производного объекта.

5. Если исходное значение закрытого свойства больше нуля, то:

5.1. Переопределить значения свойств производного объекта, увеличив на единицу введенные исходные значения.

5.2. Переопределить значения свойств родительского объекта, уменьшив на единицу введенные исходные значения.

5.3. Вывод значений свойств производного объекта.

5.4. Вывод значений свойств родительского объекта.

6. Иначе:

6.1. Переопределить значения свойств родительского объекта, увеличив на единицу введенные исходные значения.

6.2. Переопределить значения свойств производного объекта, уменьшив на единицу введенные исходные значения.

6.3. Вывод значений свойств родительского объекта.

6.4. Вывод значений свойств производного объекта.

1.1 Описание входных данных

В первой строке:

«Целое число» «Целое число»

Пример ввода:

```
8 5
```

1.2 Описание выходных данных

Начиная с первой строки:

«Целое число» «Целое число»
«Целое число» «Целое число»
«Целое число» «Целое число»
«Целое число» «Целое число»

Пример вывода:

```
16   5
8   5
9   6
14   4
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект obj класса Cl_child предназначен для демонстрация реализации наследования;
- cin - объект стандартного потока ввода;
- cout - объект стандартного потока вывода.

Класс Cl_parent:

- свойства/поля:
  - поле поле хранения значения скрытого свойства:
    - наименование — PrivateData;
    - тип — int;
    - модификатор доступа — private;
  - поле поле хранения значения открытого свойства:
    - наименование — PublicData;
    - тип — int;
    - модификатор доступа — public;

Класс Cl_child:

- свойства/поля:
  - поле поле хранения скрытого значения:
    - наименование — PrivateData;
    - тип — int;
    - модификатор доступа — private;
  - поле поле хранения открытого значения:
    - наименование — PublicData;
    - тип — int;
    - модификатор доступа — public;

- функционал:
    - метод Cl_child — конструктор, присвоение скрытому свойству значение переменной x, открытому y;
    - метод PublicChange — присвоение скрытому свойству значение переменной x, открытому - y;
    - метод Print — вывод значений скрытого и открытого свойств.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	Cl_parent			Родительский класс. Содержит основные поля и методы	
2	Cl_child			Дочерний класс класса Cl_parent	

3 ОПИСАНИЕ АЛГОРИТМОВ

Согласно этапам разработки, после определения необходимого инструментария в разделе «Метод», составляются подробные описания алгоритмов для методов классов и функций.

3.1 Алгоритм конструктора класса Cl_child

Функционал: присвоение скрытому свойству значение переменной x, открытому y.

Параметры: int x, int y.

Алгоритм конструктора представлен в таблице 2.

Таблица 2 – Алгоритм конструктора класса Cl_child

№	Предикат	Действия	№ перехода
1		присвоение значению поля PrivateData значения переменной x	2
2		Присвоение значению поля PublicData значения переменной y	∅

3.2 Алгоритм метода PublicChange класса Cl_child

Функционал: присвоение скрытому свойству значение переменной x, открытому - y.

Параметры: int x, int y.

Возвращаемое значение: -.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода PublicChange класса Cl_child

№	Предикат	Действия	№ перехода
1		присвоение значению поля PrivateData значения переменной x	2

№	Предикат	Действия	№ перехода
2		Присвоение значению поля PublicData значения переменной y	∅

3.3 Алгоритм метода Print класса Cl_child

Функционал: вывод значений скрытого и открытого свойств.

Параметры: none.

Возвращаемое значение: -.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода Print класса Cl_child

№	Предикат	Действия	№ перехода
1		вывод значения PrivateData и PublicData	∅

3.4 Алгоритм конструктора класса Cl_parent

Функционал: открытому свойству присваевается значение переменной y, для закрытого свойства вызывается метод PrivateChange(int x).

Параметры: int x, int y.

Алгоритм конструктора представлен в таблице 5.

Таблица 5 – Алгоритм конструктора класса Cl_parent

№	Предикат	Действия	№ перехода
1		присвоение знаечнию поля PublicData значения переменной y	2
2		вызов метода PrivateChange(x) текущего объекта	∅

3.5 Алгоритм метода PrivateChange класса Cl_parent

Функционал: устанавливает значение скрытого свойства, равное 2*n.

Параметры: int n.

Возвращаемое значение: -.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода PrivateChange класса Cl_parent

№	Предикат	Действия	№ перехода
1		присвоение значению поля PrivateData значения n*2	∅

3.6 Алгоритм метода PublicChange класса Cl_parent

Функционал: открытому свойству присваевается значение переменной y, для закрытого свойства вызывается метод PrivateChange(int x).

Параметры: int x, int y.

Возвращаемое значение: -.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода PublicChange класса Cl_parent

№	Предикат	Действия	№ перехода
1		присвоение значению поля PublicData значения переменной y	2
2		вызов метода PrivateChange(x) текущего объекта	∅

3.7 Алгоритм метода Print класса Cl_parent

Функционал: вывод значений скрытого и открытого свойств.

Параметры: none.

Возвращаемое значение: -.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода Print класса Cl_parent

№	Предикат	Действия	№ перехода
1		вывод значения PrivateData и PublicData	∅

3.8 Алгоритм функции main

Функционал: запуск программы.

Параметры: none.

Возвращаемое значение: код ошибки (int).

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление целочисленных переменных a, b	2
2		ввод значения переменной a	3
3		ввод значения переменной b	4
4		объявление объекта obj класса Cl_child с передачей в конструктор значений переменных a, b	5
5		вызов метода Print() объекта obj через класс Cl_parent	6
6		вывод переноса на новую строку	7
7		вызов метода Print() объекта obj через класс Cl_child	8
8		вывод переноса на новую строку	9
9	a>0	вызов метода PublicChange(a+1, b+1) объекта obj через класс Cl_child	10
		Вызов метода PublicChange(a+1, b+1) объекта obj через класс Cl_parent	14
10		вызов метода PublicChange(a-1, b-1) объекта obj через класс Cl_parent	11
11		вызов метода Print() объекта obj через класс Cl_child	12
12		вывод переноса на новую строку	13

№	Предикат	Действия	№ перехода
13		вызов метода Print() объекта obj через класс Cl_parent	14
14		вызов метода PublicChange(a-1, b-1) объекта obj через класс Cl_child	15
15		вызов метода Print() объекта obj через класс Cl_parent	16
16		вывод переноса на новую строку	17
17		вызов метода Print() объекта obj через класс Cl_child	18
18		вывод значения 0	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-2.

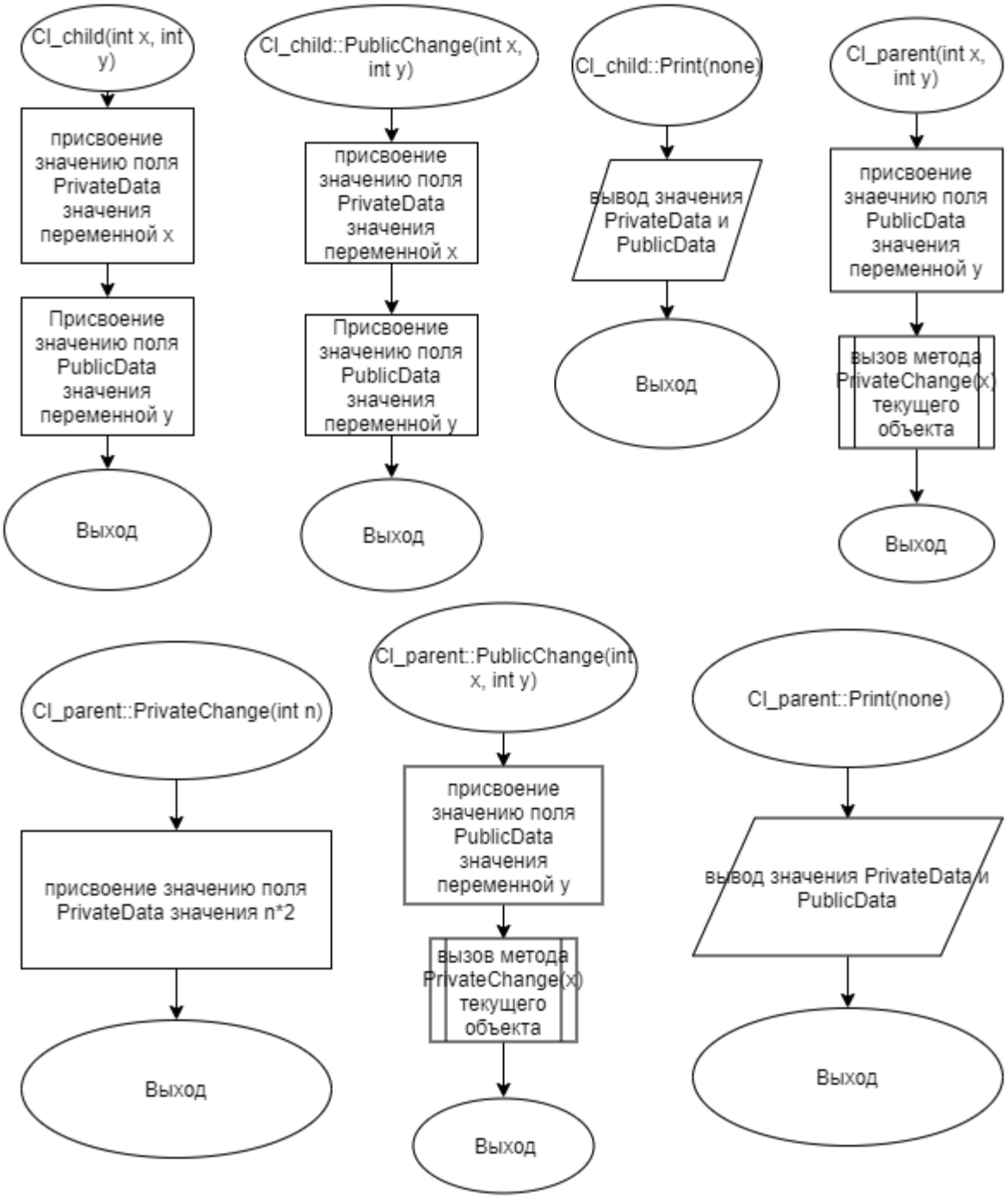

Рисунок 1 – Блок-схема алгоритма

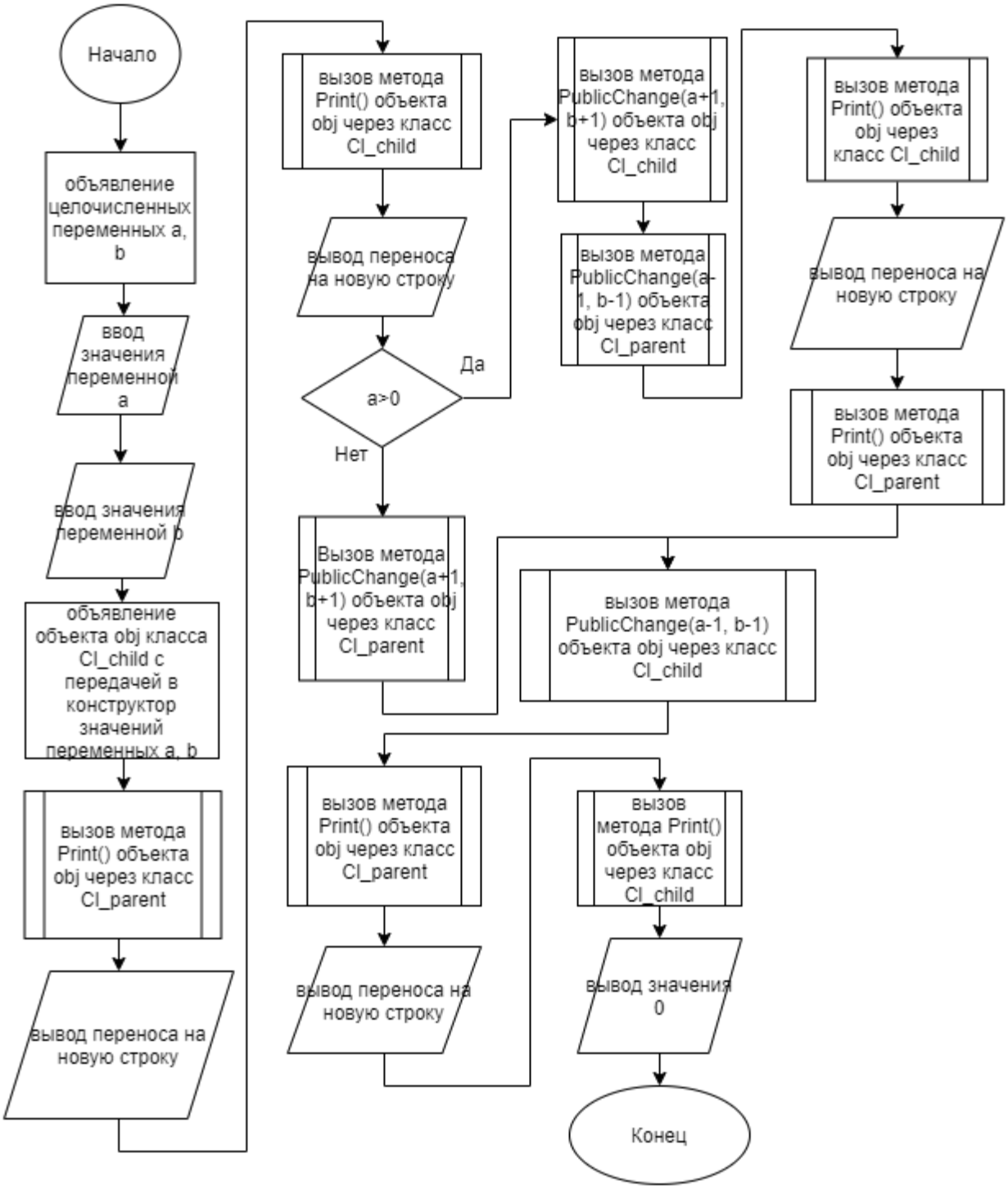

Рисунок 2 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Cl_child.cpp

Листинг 1 – Cl_child.cpp

```
#include "Cl_child.h"
#include "Cl_parent.h"
#include <iostream>

using namespace std;

Cl_child::Cl_child(int x, int y):Cl_parent::Cl_parent(x, y)
{
   PrivateData=x;
   PublicData=y;
}
void Cl_child::PublicChange(int x, int y)
{
   PrivateData=x;
   PublicData=y;
}
void Cl_child::Print()
{
   cout<<PrivateData<<"    "<<PublicData;
}
```

5.2 Файл Cl_child.h

Листинг 2 – Cl_child.h

```
#ifndef __CL_CHILD__H
#define __CL_CHILD__H
#include "Cl_parent.h"

class Cl_child:public Cl_parent
{
   private:
      int PrivateData;
   public:
```

```
        int PublicData;
        Cl_child(int x, int y);
        void PublicChange(int x, int y);
        void Print();
};

#endif
```

5.3 Файл Cl_parent.cpp

Листинг 3 – Cl_parent.cpp

```
#include "Cl_parent.h"
#include <iostream>

using namespace std;

void Cl_parent::PrivateChange(int n)
{
   PrivateData=n*2;
}
Cl_parent::Cl_parent(int x, int y)
{
   PublicData=y;
   PrivateChange(x);
}
void Cl_parent::PublicChange(int x, int y)
{
   PublicData=y;
   PrivateChange(x);
}
void Cl_parent::Print()
{
   cout<<PrivateData<<"    "<<PublicData;
}
```

5.4 Файл Cl_parent.h

Листинг 4 – Cl_parent.h

```
#ifndef __CL_PARENT__H
#define __CL_PARENT__H

class Cl_parent
{
```

```
   private:
      int PrivateData;
      void PrivateChange(int n);
   public:
      int PublicData;
      Cl_parent(int x, int y);
      void PublicChange(int x, int y);
      void Print();
};

#endif
```

5.5 Файл main.cpp

Листинг 5 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include <iostream>
#include "Cl_parent.h"
#include "Cl_child.h"

using namespace std;

int main()
{
   int a, b;
   cin>>a>>b;
   Cl_child obj(a, b);
   obj.Cl_parent::Print();
   cout<<endl;
   obj.Cl_child::Print();
   cout<<endl;
   if (a>0)
   {
      obj.Cl_child::PublicChange(a+1, b+1);
      obj.Cl_parent::PublicChange(a-1, b-1);
      obj.Cl_child::Print();
      cout<<endl;
      obj.Cl_parent::Print();
   }
   else
   {
      obj.Cl_parent::PublicChange(a+1, b+1);
      obj.Cl_child::PublicChange(a-1, b-1);
      obj.Cl_parent::Print();
      cout<<endl;
      obj.Cl_child::Print();
   }
   return(0);
```

```
    }
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 10.

Таблица 10 – Результат тестирования программы

Входные данные	**Ожидаемые выходные данные**	**Фактические выходные данные**
8 5	16 5 8 5 9 6 14 4	16 5 8 5 9 6 14 4
6 9	12 9 6 9 7 10 10 8	12 9 6 9 7 10 10 8

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 19 Единая система программной документации.

2. Методическое пособие студента для выполнения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratornyh_rabot_3.pdf (дата обращения 05.05.2021).

3. Приложение к методическому пособию студента по выполнению заданий в рамках курса «Объектно-ориентированное программирование» [Электронный ресурс]. URL: https://mirea.aco-avrora.ru/student/files/Prilozheniye_k_methodichke.pdf (дата обращения 05.05.2021).

4. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2019. — 624 с.

5. Видео лекции по курсу «Объектно-ориентированное программирование» [Электронный ресурс]. АСО «Аврора».

6. Антик М.И. Дискретная математика [Электронный ресурс]: Учебное пособие /Антик М.И., Казанцева Л.В. — М.: МИРЭА — Российский технологический университет, 2018 — 1 электрон. опт. диск (CD-ROM).