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 Object класса Child предназначен для ;
- функция main для Главная функция программы;
- Объект потока ввода cin с клавиатуры;
- Объект потока вывода cout на экран;
- Библиотека iostream;
- Пространство имён std.

Класс Parent:

- свойства/поля:
    - поле хранение значения скрытого поля:
        - наименование — dataPrivate;
        - тип — int;
        - модификатор доступа — private;
    - поле хранение значения открытого поля:
        - наименование — dataPublic;
        - тип — int;
        - модификатор доступа — public;
- функционал:
    - метод MultiplyPrivate — установка значения скрытого поля dataPrivate как умножение переданной переменной на 2;
    - метод Parent — параметризированный конструктор, присваивающий открытому полю переданное значение переменной y и вызывающий метод MultiplyPrivate() с передачей ему в качестве параметра переменную x;
    - метод ChangePublic — присваивает открытому полю dataPublic

значение переданной переменной y и вызывает метод MultiplyPrivate с передачей ему значения переменной x;

- метод Output — Вывод значений скрытого и открытого полей.

Класс Child:

- свойства/поля:
  - поле хранение значения скрытого поля:
    - наименование — dataPrivate;
    - тип — int;
    - модификатор доступа — private;
  - поле хранение значения открытого поля:
    - наименование — dataPublic;
    - тип — int;
    - модификатор доступа — public;
- функционал:
  - метод Child — параметризированный конструктор. Присваивает скрытому полю dataPrivate переданное значение переменной x, а открытому полю dataPublic значение переданной переменной y;
  - метод ChangePublic — присваивает скрытому полю dataPrivate переданное значение переменной x, а открытому полю dataPublic значение переданной переменной y;
  - метод Output — Вывод значения полей на экран.

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа при наследовании	Описание	Номер
1	Parent				
		Child	public		2
2	Child				

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 метода MultiplyPrivate класса Parent

Функционал: установка значения скрытого поля dataPrivate как умножение переданной переменной на 2.

Параметры: переменная a целого типа.

Возвращаемое значение: отсутствует.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода MultiplyPrivate класса Parent

№	Предикат	Действия	№ перехода
1		Присвоение скрытому полю dataPrivate значения произведения переданной переменной a на 2	∅

3.2 Алгоритм конструктора класса Parent

Функционал: параметризированный конструктор, присваивающий открытому полю переданное значение переменной y и вызывающий метод MultiplyPrivate() с передачей ему в качестве параметра переменную x.

Параметры: переменные x и y целого типа.

Алгоритм конструктора представлен в таблице 3.

Таблица 3 – Алгоритм конструктора класса Parent

№	Предикат	Действия	№ перехода
1		Присвоение открытому полю dataPublic значения переменной y	2
2		Вызов метода MultiplyPrivate текущего объекта с передачей ему значения переменной x	∅

3.3 Алгоритм метода ChangePublic класса Parent

Функционал: присваивает открытому полю dataPublic значение переданной переменной y и вызывает метод MultiplyPrivate с передачей ему значения переменной x.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода ChangePublic класса Parent

№	Предикат	Действия	№ перехода
1		Присвоение открытому полю dataPublic значения переменной y	2
2		Вызов метода MultiplyPrivate текущего объекта с передачей ему значения переменной x	∅

3.4 Алгоритм метода Output класса Parent

Функционал: Вывод значений скрытого и открытого полей.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода Output класса Parent

№	Предикат	Действия	№ перехода
1		Вывод: <<значение поля dataPrivate>>," ",<<значение поля dataPublic>>	∅

3.5 Алгоритм конструктора класса Child

Функционал: параметризированный конструктор. Присваивает скрытому полю dataPrivate переданное значение переменной x, а открытому полю dataPublic значение переданной переменной y.

Параметры: переменные x и y целого типа.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Child

№	Предикат	Действия	№ перехода
1		Присвоение скрытому полю dataPrivate значения переменной x	2
2		Присвоение открытому полю dataPublic значения переменной y	∅

3.6 Алгоритм метода ChangePublic класса Child

Функционал: присваивает скрытому полю dataPrivate переданное значение переменной x, а открытому полю dataPublic значение переданной переменной y.

Параметры: переменные x и y целого типа.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода ChangePublic класса Child

№	Предикат	Действия	№ перехода
1		Присвоение скрытому полю dataPrivate значения переменной x	2

№	Предикат	Действия	№ перехода
2		Присвоение открытому полю dataPublic значения переменной y	∅

3.7 Алгоритм метода Output класса Child

Функционал: Вывод значения полей на экран.

Параметры: нет.

Возвращаемое значение: Отсутствует.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода Output класса Child

№	Предикат	Действия	№ перехода
1		Вывод: <<значение поля dataPrivate>>," ",<<значение поля dataPublic>>	∅

3.8 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое, индикация корректности работы программы.

Алгоритм функции представлен в таблице 9.

Таблица 9 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление переменных x и y целого типа	2
2		Ввод значения переменной x	3
3		Ввод значения переменной y	4
4		Объявление объекта Object класса child с передачей в конструктор значений переменных x и y	5

№	Предикат	Действия	№ перехода
5		Вызов метода Output() объекта Object посредством класса Parent	6
6		Переход на новую строку	7
7		Вызов метода Output() объекта Object посредством класса Child	8
8		Переход на новую строку	9
9	Значение переменной x больше нуля	Вызов метода ChangePublic() с передачей значений аргументов (x + 1) и (y + 1) объекта Object посредством класса Child	10
	Значение переменной x не больше нуля	Вызов метода ChangePublic() с передачей значений аргументов (x + 1) и (y + 1) объекта Object посредством класса Parent	14
10		Вызов метода ChangePublic() с передачей значений аргументов (x - 1) и (y - 1) объекта Object посредством класса Parent	11
11		Вызов метода Output() объекта Object посредством класса Child	12
12		Переход на новую строку	13
13		Вызов метода Output() объекта Object посредством класса Parent	14
14		Вызов метода ChangePublic() с передачей значений аргументов (x - 1) и (y - 1) объекта Object посредством класса Child	15
15		Вызов метода Output() объекта Object посредством класса Parent	16
16		Переход на новую строку	17
17		Вызов метода Output() объекта Object посредством класса Child	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

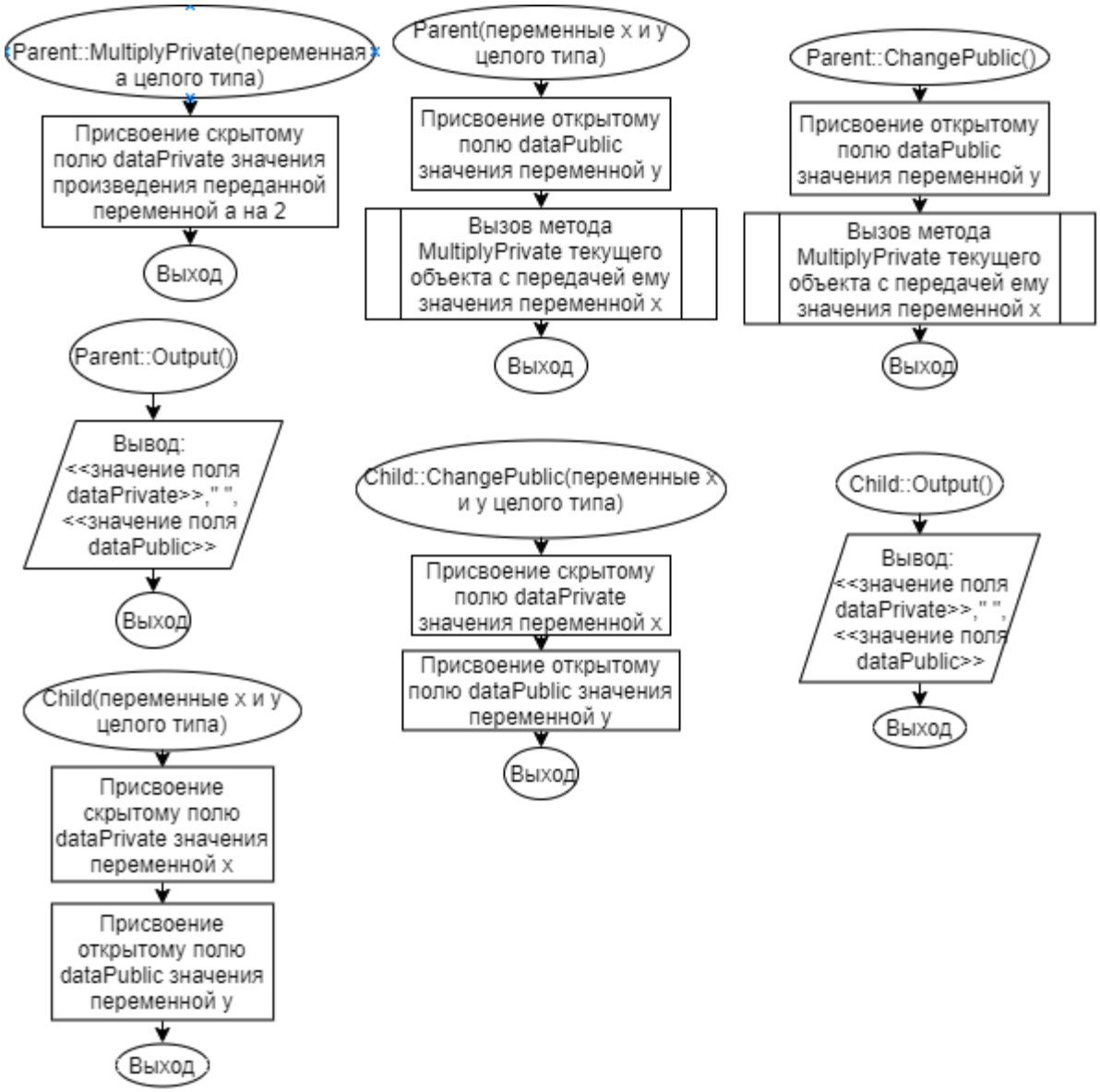

Рисунок 1 – Блок-схема алгоритма

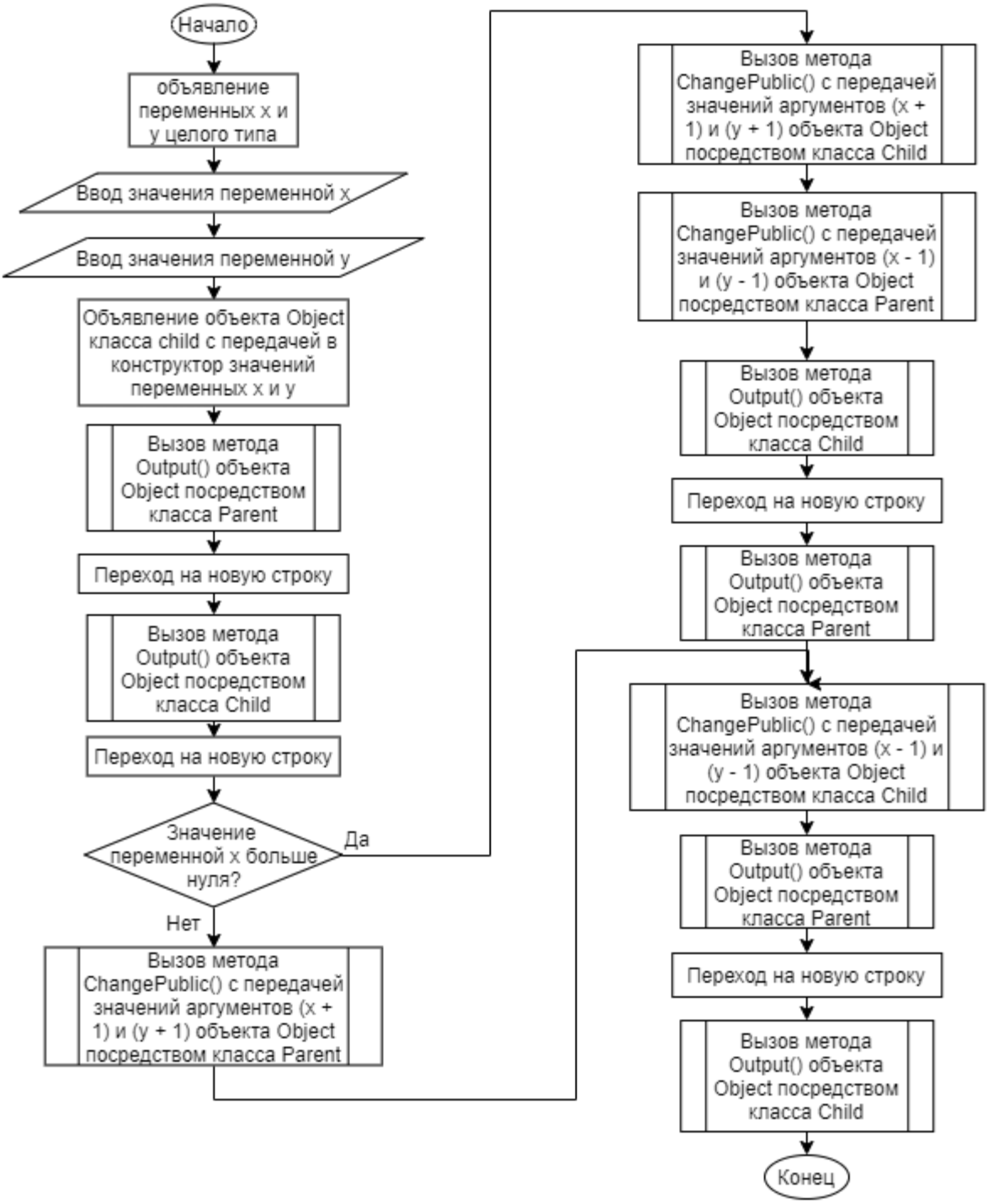

Рисунок 2 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл Child.cpp

Листинг 1 – Child.cpp

```
#include "Child.h"
#include "Parent.h"
#include <iostream>
using namespace std;
Child::Child(int x, int y):Parent::Parent(x, y)
{
   dataPrivate = x;
   dataPublic = y;
}
void Child::ChangePublic(int x, int y)
{
   dataPrivate = x;
   dataPublic = y;
}
void Child::Output()
{
   cout << dataPrivate << "    " << dataPublic;
}
```

5.2 Файл Child.h

Листинг 2 – Child.h

```
#ifndef __CHILD__H
#define __CHILD__H
#include "Parent.h"
using namespace std;
class Child:public Parent
{
   private:
   int dataPrivate;
   public:
   int dataPublic;
   Child(int x, int y);
```

```
   void ChangePublic(int x, int y);
   void Output();
};


#endif
```

5.3 Файл main.cpp

Листинг 3 – main.cpp

```
#include "Child.h"
#include "Parent.h"
#include <iostream>
using namespace std;
int main()
{
   int x, y;
   cin >> x >> y;
   Child Object(x, y);
   Object.Parent::Output();
   cout << "\n";
   Object.Child::Output();
   cout << "\n";
   if (x > 0)
   {
      Object.Child::ChangePublic(x + 1, y + 1);
      Object.Parent::ChangePublic(x - 1, y - 1);
      Object.Child::Output();
      cout << "\n";
      Object.Parent::Output();
   }
   else
   {
      Object.Child::ChangePublic(x - 1, y - 1);
      Object.Parent::ChangePublic(x + 1, y + 1);
      Object.Parent::Output();
      cout << "\n";
      Object.Child::Output();
   }
   // program here
   return(0);
}
```

5.4 Файл Parent.cpp

Листинг 4 – Parent.cpp

```
#include "Parent.h"
#include <iostream>
using namespace std;
void Parent::MultiplyPrivate(int a)
{
   dataPrivate = 2 * a;
}
Parent::Parent(int x, int y)
{
   dataPublic = y;
   MultiplyPrivate(x);
}
void Parent::ChangePublic(int x, int y)
{
   dataPublic = y;
   MultiplyPrivate(x);
}
void Parent::Output()
{
   cout << dataPrivate << "    " << dataPublic;
};
```

5.5 Файл Parent.h

Листинг 5 – Parent.h

```
#ifndef __PARENT__H
#define __PARENT__H
using namespace std;
class Parent
{
   private:
   int dataPrivate;
   void MultiplyPrivate(int a);
   public:
   int dataPublic;
   Parent(int x, int y);
   void ChangePublic(int x, int y);
   void Output();
};

#endif
```

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
-2 7	-4 7 -2 7 -2 8 -3 6	-4 7 -2 7 -2 8 -3 6
46 90	92 90 46 90 47 91 90 89	92 90 46 90 47 91 90 89

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).