Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Дан объект следующей конструкции:

В закрытом доступе имеется массив целого типа и поле его длины. Количество элементов массива четное и больше двух. Объект имеет функциональность:

- Конструктор по умолчанию, вначале работы выдает сообщение;
- Параметризированный конструктор, передается целочисленный параметр. Параметр должен иметь значение больше 2 и быть четным. Вначале работы выдает сообщение;
- Конструктор копии, обеспечивает создание копии объекта в новой области памяти. Вначале работы выдает сообщение;
- Метод деструктор, который в начале работы выдает сообщение;
- Метод который создает целочисленный массив в закрытой области, согласно ранее заданной размерности.
- Метод ввода данных для созданного массива;
- Метод 1, который суммирует значения очередной пары элементов и сумму присваивает первому элементу пары. Например, пусть массив состоит из элементов {1,2,3,4}. В результате суммирования пар получим массив {3,2,7,4};
- Метод 2, который умножает значения очередной пары элементов и результат присваивает первому элементу пары. Например, пусть массив состоит из элементов {1,2,3,4}. В результате умножения пар получим массив {2,2,12,4};
- Метод который, суммирует значения элементов массива и возвращает это значение;
- Метод последовательного вывода содержимого элементов массива, которые

разделены тремя пробелами.

Разработать функцию func, которая имеет один целочисленный параметр, содержащий размерность массива. В функции должен быть реализован алгоритм:

- Создание локального объекта с использованием параметризированного конструктора.
- Возврат созданного локального объекта.

В основной функции реализовать алгоритм:

- Ввод размерности массива.
- Если размерность массива некорректная, вывод сообщения и завершить работу алгоритма.
- Вывод значения размерности массива.
- Создание первого объекта.
- Присвоение первому объекту результата работы функции func с аргументом, содержащим значение размерности массива.
- Для первого объекта вызов метода создания массива.
- Для первого объекта вызов метода ввода данных массива.
- Для первого объекта вызов метода 2.
- Инициализация второго объекта первым объектом.
- Вызов метода 1 для второго объекта.
- Вывод содержимого массива первого объекта.
- Вывод суммы элементов массива первого объекта.
- Вывод содержимого массива второго объекта.
- Вывод суммы элементов массива второго объекта.

1.1 Описание входных данных

Первая строка:

```
«Целое число»
```

Вторая строка:

```
«Целое число» «Целое число» . . .
```

Пример:

```
4
3 5 1 2
```

1.2 Описание выходных данных

Если введенная размерность массива допустима, то в первой строке выводится это значение:

```
«Целое число»
```

Если введенная размерность массива не больше двух или нечетная, то в первой строке выводится некорректное значение и вопросительный знак:

```
«Целое число»?
```

Конструктор по умолчанию в начале работы с новой строки выдает сообщение:

```
Default constructor
```

Параметризированный конструктор в начале работы с новой строки выдает сообщение:

```
Constructor set
```

Конструктор копии в начале работы с новой строки выдает сообщение:

```
Copy constructor
```

Деструктор в начале работы с новой строки выдает сообщение:

```
Destructor
```

Метод последовательного вывода содержимого элементов массива, с новой строки выдает:

```
«Целое число»   «Целое число»   «Целое число»   . . .
```

Пример вывода:

```
4
Default constructor
Constructor set
Destructor
```

```
Copy constructor
15   5   2   2
24
20   5   4   2
31
Destructor
Destructor
```

2 МЕТОД РЕШЕНИЯ

Класс Object:

- Метод make_array:
  - Функционал - создание целочисленного массива в закрытой области, согласно ранее заданной размерности;
  - Возвращаемое значение - void (отсутствует);
  - Модификатор доступа - открытый;
  - Параметры - отсутствуют;
- Метод print:
  - Функционал - последовательный вывод содержимого элементов массива, которые разделены тремя пробелами;
  - Возвращаемое значение - void (отсутствует);
  - Модификатор доступа - открытый;
  - Параметры - отсутствуют.

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 функции main

Функционал: основной алгоритм программы.

Параметры: .

Возвращаемое значение: целочисленный код завершения работы программы.

Алгоритм функции представлен в таблице 1.

Таблица 1 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Инициализация целочисленной переменной size значением 0	2
2		Ввод size	3
3		Вывод size	4
4	Значение переменной size меньше 2 или остаток от деления значения переменной size на 2 равен 1	Вывод "?"	∅
			5
5		Создание объекта obj1 класса Object с использованием конструктора по умолчанию	6
6		Присваивание объекту obj1 объекта, возвращённого функцией func, вызванной с переменной size в качестве параметра	7

№	Предикат	Действия	№ перехода
7		Вызов метода make_array объекта obj1	8
8		Вызов метода read_array объекта obj1	9
9		Вызов метода method2 объекта obj1	10
10		Создание объекта obj2 с использованием параметризованного конструктора и объектом obj1 в качестве параметра	11
11		Вызов метода method1 объекта obj2	12
12		Вывод "\n" (переход на новую строку)	13
13		Вызов метода print объекта obj1	14
14		Вывод '\n' и значения, возвращённого методом sum объекта obj1	15
15		Вывод "\n" (переход на новую строку)	16
16		Вызов метода print объекта obj2	17
17		Вывод '\n' и значения, возвращённого методом sum объекта obj2	∅

3.2 Алгоритм функции func

Функционал: создание объекта класса Object с использованием параметризованного конструктора с параметром size в качестве аргумента.

Параметры: целочисленный параметр size.

Возвращаемое значение: объект класса Object.

Алгоритм функции представлен в таблице 2.

Таблица 2 – Алгоритм функции func

№	Предикат	Действия	№ перехода
1		Создание объекта obj класса Object с использованием параметризованного конструктора и параметром size в качестве аргумента	2
2		Возвращение объекта obj	∅

3.3 Алгоритм метода make_array класса Object

Функционал: создание целочисленного массива в закрытой области, согласно ранее заданной размерности.

Параметры: .

Возвращаемое значение: void (отсутствует).

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода make_array класса Object

№	Предикат	Действия	№ перехода
1		Присваивание полю array адреса нового целочисленного массива размера size созданного с помощью оператора new	∅

3.4 Алгоритм метода print класса Object

Функционал: последовательный вывод содержимого элементов массива, которые разделены тремя пробелами.

Параметры: .

Возвращаемое значение: void (отсутствует).

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода print класса Object

№	Предикат	Действия	№ перехода
1		Инициализация целочисленной переменной i значением 0	2
2	Значение переменной i меньше значения поля size		3
			∅
3		Вывод элемента массива array с индексом i	4
4	Сумма значения переменной i и 1 меньше значения поля size	Вывод " " (3 пробела)	5
			5
5		Увеличение значения переменной i на 1	2

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-8.

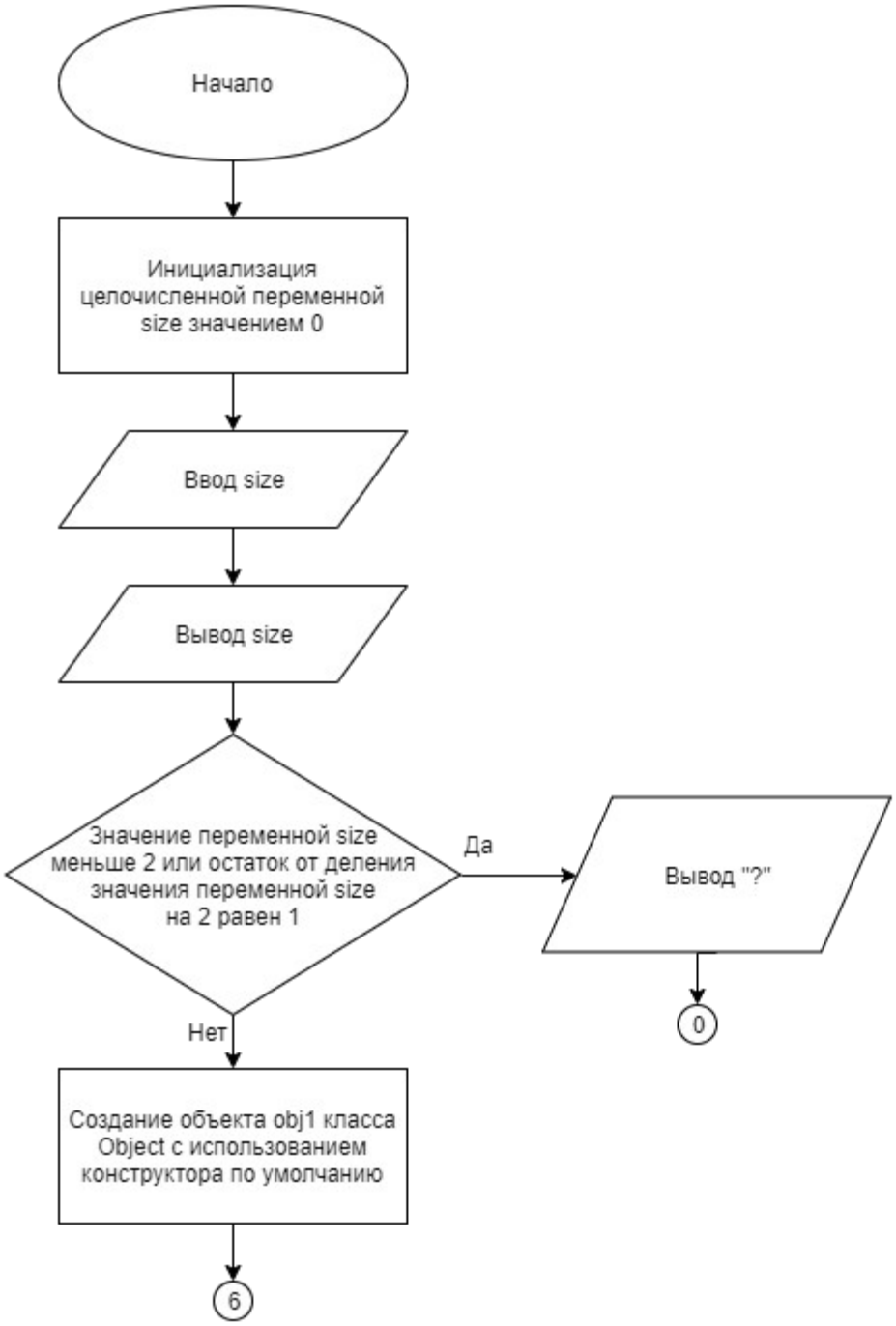

Рисунок 1 – Блок-схема алгоритма

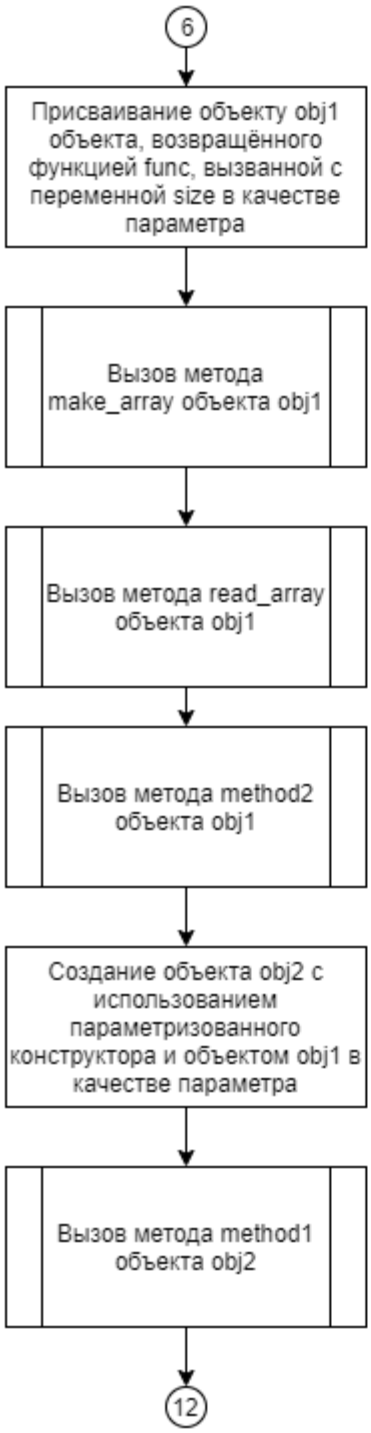

Рисунок 2 – Блок-схема алгоритма

Рисунок 3 – Блок-схема алгоритма

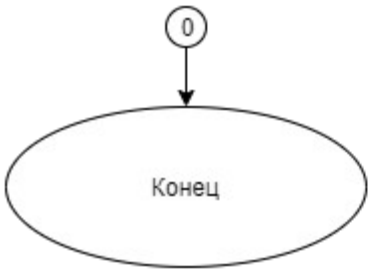

Рисунок 4 – Блок-схема алгоритма

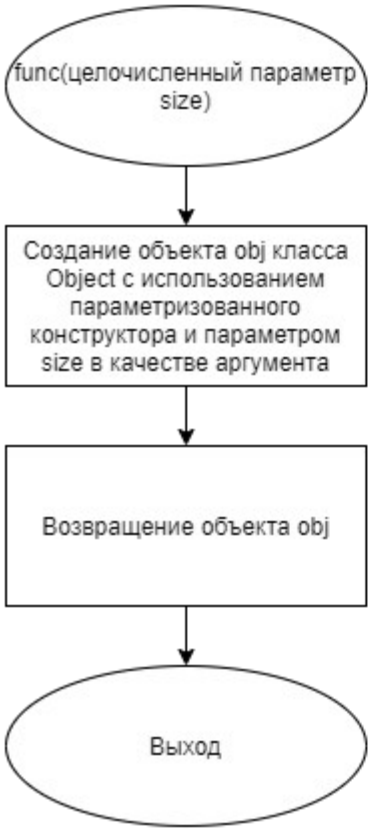

Рисунок 5 – Блок-схема алгоритма

Рисунок 6 – Блок-схема алгоритма

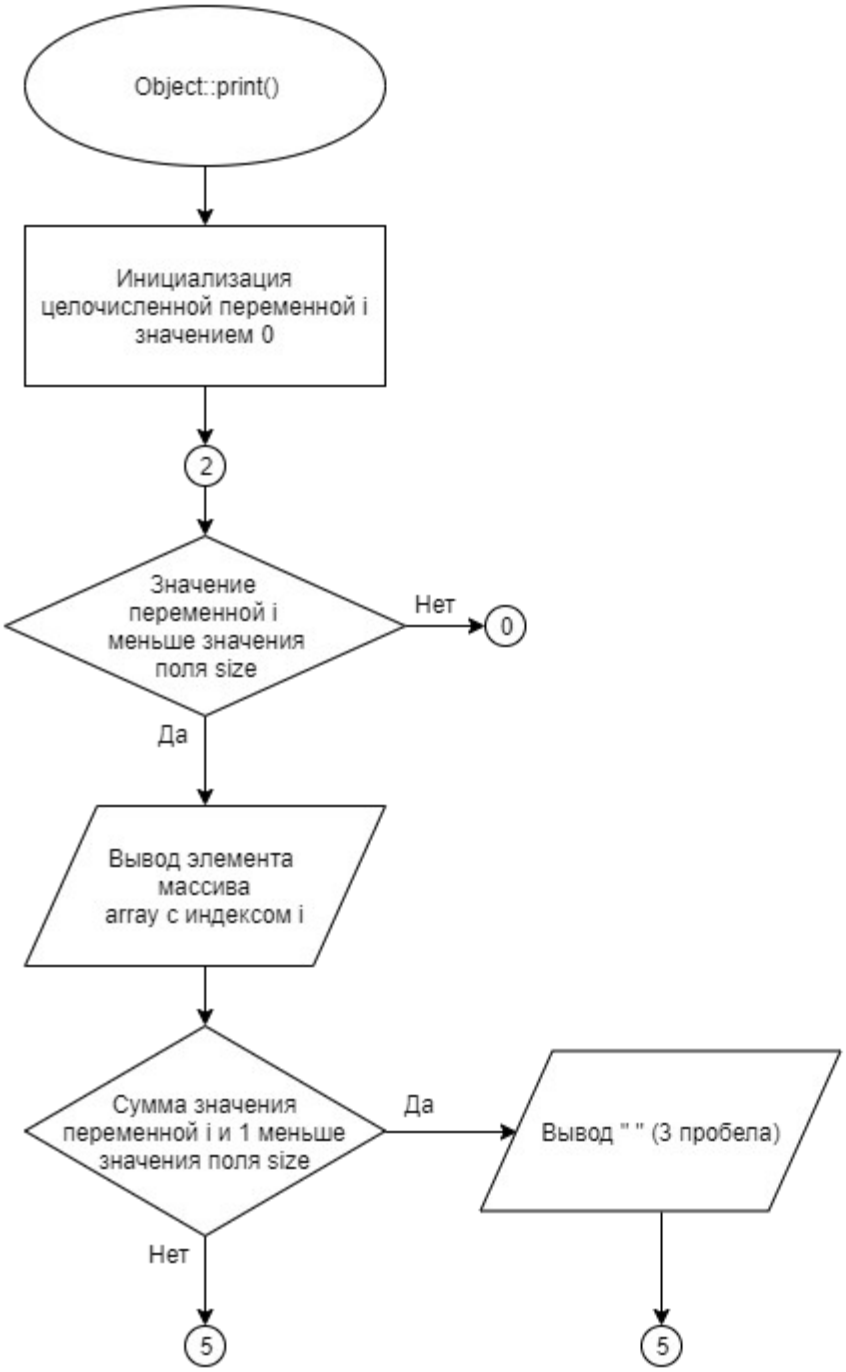

Рисунок 7 – Блок-схема алгоритма

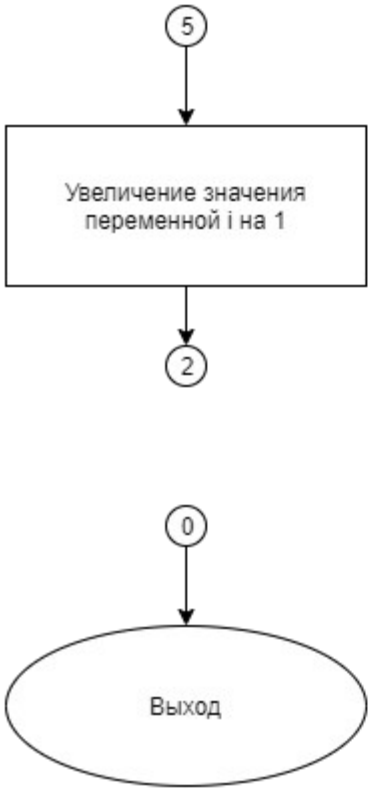

Рисунок 8 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

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

Object func(int size)
{
	// Создание объекта obj
	// с использованием
	// параметризованного
	// конструктора
	Object obj(size);
	// Возвращение
	// объекта obj
	return obj;
}

int main()
{
	// Инициализация целочисленной переменной
	// size значением 0
	int size = 0;
	// Ввод size
	std::cin >> size;
	// Вывод size
	std::cout << size;

	// Если значение size меньше
	// двух или нечётно
	if (size < 2 || size % 2 == 1)
	{
		// Вывод "?"
		std::cout << '?';
	}
	// иначе
	else
	{
		// Создание объекта obj1
		// класса Object
		Object obj1;
		// Присваивание объекту
		// obj1 результата работы
		// функции func
```

```
            obj1 = func(size);
            // Вызов метода make_array
            // объекта obj1
            obj1.make_array();
            // Вызов метода read_array
            // объекта obj1
            obj1.read_array();
            // Вызов метода method2
            // объекта obj1
            obj1.method2();

            // Создание объекта obj2
            // с использованием
            // параметризованного
            // конструктора
            Object obj2(obj1);
            // Вызов метода method1
            // объекта obj2
            obj2.method1();

            // Переход на новую строку
            std::cout << '\n';
            // Вызов метода print
            // объекта obj1
            obj1.print();
            // Вывод '\n' и значения,
            // возвращённого методом sum
            // объекта obj1
            std::cout << '\n' << obj1.sum();

            // Переход на новую строку
            std::cout << '\n';
            // Вызов метода print
            // объекта obj2
            obj2.print();
            // Вывод '\n' и значения,
            // возвращённого методом sum
            // объекта obj2
            std::cout << '\n' << obj2.sum();
      }
      return 0;
}
```

5.2 Файл Object.cpp

Листинг 2 – Object.cpp

```
#include "Object.h"

Object::Object()
{
      std::cout << "\nDefault constructor";
      size = 0;
```

```
	array = nullptr;
}

Object::Object(int size)
{
	std::cout << "\nConstructor set";
	this->size = size;
	array = new int[size];
}

Object::Object(const Object& obj)
{
	std::cout << "\nCopy constructor";
	size = obj.size;
	array = new int[size];
	for (int i = 0; i < size; i++)
	{
		array[i] = obj.array[i];
	}
}

Object::~Object()
{
	std::cout << "\nDestructor";
	delete[] array;
}

void Object::read_array()
{
	for (int i = 0; i < size; i++)
	{
		std::cin >> array[i];
	}
}

int Object::method1()
{
	for (int i = 0; i < size - 1; i += 2)
	{
		array[i] = array[i] + array[i + 1];
	}

	return sum();
}

int Object::method2()
{
	for (int i = 0; i < size - 1; i += 2)
	{
		array[i] = array[i] * array[i + 1];
	}

	return sum();
}

int Object::sum()
{
```

```
	int sum = 0;
	for (int i = 0; i < size; i++)
	{
		sum += array[i];
	}

	return sum;
}

void Object::make_array()
{
	array = new int[size];
}

void Object::print()
{
	for (int i = 0; i < size; i++)
	{
		std::cout << array[i];
		if (i + 1 < size)
		{
			std::cout << "   ";
		}
	}
}
```

5.3 Файл Object.h

Листинг 3 – Object.h

```
#ifndef OBJECT_H
#define OBJECT_H

#include <iostream>

class Object
{
private:
	int* array;
	int size;
public:
	Object();
	Object(int size);
	Object(const Object& obj);
	~Object();
	void read_array();
	int method1();
	int method2();
	int sum();

	void make_array();
	void print();
};
```

```
#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 5.

Таблица 5 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
4 3 5 1 2	4 Default constructor Constructor set Destructor Copy constructor 15 5 2 2 24 20 5 4 2 31 Destructor Destructor	4 Default constructor Constructor set Destructor Copy constructor 15 5 2 2 24 20 5 4 2 31 Destructor Destructor
0	0?	0?
3	3?	3?

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Васильев А.Н. Объектно-ориентированное программирование на C++. Издательство: Наука и Техника. Санкт-Петербург, 2016г. 543 стр.

2. Шилдт Г. C++: базовый курс. 3-е изд. Пер. с англ.. — М.: Вильямс, 2017. — 624 с.

3. Методическое пособие для проведения практических заданий, контрольных и курсовых работ по дисциплине «Объектно-ориентированное программирование» [Электронный ресурс] – URL: https://mirea.aco-avrora.ru/student/files/methodichescoe_posobie_dlya_laboratorny h_rabot_3.pdf (дата обращения 05.05.2021).

4. 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).

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).