Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

Разработать систему, которая демонстрирует работу (использование) конструктора копии.

Первоначально, в процессе конструирования, система состоит из одного объекта (исходного) и из одной функции. Далее, ее работа управляется посредством команд. Команды вводятся с стандартного потока данных. По команде создаются копии объектов от исходного или от других копии. Копия так же создается при передаче объекта в функцию по значению.

Спроектировать объект, с свойствами в закрытом доступе:

- строкового типа, для хранения наименования объекта;
- целого типа, для хранения номера копии объекта;
- целого типа, для хранения размерности массива;
- указатель на объект целого типа, для хранения адреса динамически созданного целочисленного массива;
- свойство, для хранения целочисленного значения, который доступен всем копиям (инструкцию static не использовать);
- еще из дополнительных свойств, для реализации требований, заданных в постановке задачи.

С параметризированным конструктором. У конструктора есть параметр строкового типа. Параметр передает (содержит) значение наименования объекта. В реализации метода фиксируется имя объекта. Свойству номера копии присваивается значение нуль. Общедоступному свойству тоже присваивается нуль. Есть фрагмент, в котором вводится значение размерности целочисленного массива, динамический создается целочисленный массив согласно значению размерности и вводятся значения элементов массива.

С конструктором копии. В конструкторе копии из переданному в качестве

параметра объекту в текущем:

- копируется наименование объекта;
- определяется номер новой копии и присваивается соответствующему свойству;
- определяется значение свойства с общим доступом;
- копируется размерность массива:
- динамический создается целочисленный массив согласно значению размерности. Элементам массива присваиваются значения элементов исходного массива увеличенные на значение номера копии текущего объекта;
- выводиться наименование объекта и номер копии.

Имеется функционал (методы) в открытом доступе:

- с одним целочисленным параметром, значением которого редактируется значение свойства с общим доступом;
- возвращает значение номера копии объекта;
- выводит состояние объекта; наименование объекта; номер копии; значение свойства с общим доступом; значения элементов массива;
- деструктор, который сообщает объект с каким именем и с каким номер копии уничтожается, освобождается память, выделенная для целочисленного массива.

В составе системы определена функция с одним параметром. По этому параметру передается объект по значению. В функции для объекта, переданного по параметру, вызывается метод вывода состояния объекта.

Система работает по следующим командам:

```
copy «целое число, номер копии»
```

По данной команде ищется объект с заданным номером копии, создается новый объект посредством конструктора копии, в котором в качестве аргумента

передается найденный объект. Выводиться состояние нового объекта.

```
function «целое число, номер копии»
```

По данной команде ищется объект с заданным номером копии, вызывается функция и в качестве аргумента передается найденный объект.

```
state «целое число, номер копии»
```

По данной команде ищется объект с заданным номером копии и выводится его состояние.

```
shared «целое число, номер копии» «целое число, значение свойства с общим доступом»
```

По данной команде ищется объект с заданным номером копии. Вводиться значение свойства с общим доступом. Для найденного объекта вызывается метод редактирования значения свойства с общим доступом.

```
end
```

По данной команде система завершает работу.

Если не удается найти объект с данным номер копии, то выдается соответствующее сообщение.

Алгоритм конструирования и отработки системы:

1. Объявляются строковые переменные, для хранения наименования объекта и значения команд системы.
2. Объявляются целочисленные переменные для хранения значений: номера копии объекта и значения свойства с общим доступом.
3. Могут быть другие объявления.
4. Вводится значение наименования объекта.
5. Выводится значение наименования объекта.
6. Создается объект посредством параметризированного конструктора.
7. Для созданного объекта вызывается метод вывода состояния объекта.
8. Могут быть другие операторы.

9. Организуется цикл по командам.

1.1 Описание входных данных

Первая строка:

```
«строка, наименование объекта»
```

Вторая строка:

```
«целое число, размерность массива»
```

Третья строка:

```
«целое число» «целое число» . . . «целое число»
```

Значения элементов массива.

Начиная с четвертой строки, построчно, вводятся команды системы.

```
команда «целое число, номер копии» [«целое число, значение свойства с общим доступом»]
```

Пример ввода

```
Document
5
1 2 3 4 5
copy 0
copy 1
shared 8 99
copy 1
shared 2 77
copy 0
function 1
state 5
state 1
end
```

1.2 Описание выходных данных

Каждый вывод производится с новой строки.

Вывод наименования объекта:

```
Object name: «наименование объекта»
```

Вывод состояния объекта занимает две строки. Первая строка:

```
Object name: «наименование объекта»      Copy: «номер копии»      Shared: «значение свойства с общим доступом»
```

Вторая строка:

```
Array:  «целое число»  «целое число» . . . «целое число»
```

Сообщение конструктора копии имеет следующий шаблон:

```
Construcrot copy Object name: «наименование объекта»    Copy: «номер копии»
```

Если объект с заданным номером копии не найден, то выводиться сообщение:

```
A copy of object number «номер копии» was not found.
```

Сообщение деструктора копии имеет следующий шаблон:

```
Destructor Object name: «наименование объекта»    Copy: «номер копии»
```

Пример вывода

```
Object name: Document
Object name: Document    Copy: 0    Shared: 0
Array:  1  2  3  4  5
Construcrot copy Object name: Document    Copy: 1
Object name: Document    Copy: 1    Shared: 0
Array:  2  3  4  5  6
Construcrot copy Object name: Document    Copy: 2
Object name: Document    Copy: 2    Shared: 0
Array:  4  5  6  7  8
A copy of object number 8 was not found.
Construcrot copy Object name: Document    Copy: 3
Object name: Document    Copy: 3    Shared: 0
Array:  5  6  7  8  9
Construcrot copy Object name: Document    Copy: 4
Object name: Document    Copy: 4    Shared: 77
Array:  5  6  7  8  9
Construcrot copy Object name: Document    Copy: 5
Object name: Document    Copy: 5    Shared: 77
Array:  7  8  9  10  11
Destructor Object name: Document    Copy: 5
A copy of object number 5 was not found.
Object name: Document    Copy: 1    Shared: 77
Array:  2  3  4  5  6
```

2 МЕТОД РЕШЕНИЯ

Для решения задачи используется:

- объект original класса Object предназначен для демонстрации работыконструктора копии;
- функция processFunction для вызова метода состояния объекта;
- библиотека iostream;
- библиотека string;
- строковая переменная;
- целочисленная переменная;
- цикл со счетчиком for;
- оператор условия if else;
- объект стандартного потока ввода/вывода данных cin/cout;
- оператор прерывания цикла break;
- оператор выделения памяти new;
- оператор очистки памяти break.

Класс Object:

- свойства/поля:
    - поле имя объекта:
        - наименование — name;
        - тип — int;
        - модификатор доступа — private;
    - поле номер копии:
        - наименование — copyNumber;
        - тип — int;
        - модификатор доступа — private;
    - поле размер массива:

        - наименование — arraySize;
        - тип — int;
        - модификатор доступа — private;
    - поле массив:
        - наименование — intArray;
        - тип — int*;
        - модификатор доступа — private;
    - поле значение свойства с общим доступом:
        - наименование — sharedValue;
        - тип — int&;
        - модификатор доступа — private;
- функционал:
    - метод Object — параметризированный конструктор;
    - метод Object — конструктор копии;
    - метод ~Object — деструктор по умолчанию;
    - метод setSharedValue — установка общедоступного свойства;
    - метод getCopyNumber — возвращает номер копии;
    - метод printState — вывод свойств объекта.

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 Object

Функционал: параметризированный конструктор.

Параметры: строка, objName, имя объекта.

Алгоритм конструктора представлен в таблице 1.

Таблица 1 – Алгоритм конструктора класса Object

№	Предикат	Действия	№ перехода
1		полю name присвоить значение objName	2
2		полю sharedValue присвоить значение globalSharedValue	3
3		полю copyNumber присвоить 0	4
4		вывод "Object name: ", значение name, переход на новую строку	5
5		ввод значения arraySize	6
6		полю intArray присвоить целочисленный массив размера arraySize	7
7		инициализация целочисленной переменной i со значением 0	8
8	i < arraySize ?	ввод значения intArray[i]	9
			∅
9		инкремент i	8

3.2 Алгоритм деструктора класса Object

Функционал: деструктор по умолчанию.

Параметры: нет.

Алгоритм деструктора представлен в таблице 2.

Таблица 2 – Алгоритм деструктора класса Object

№	Предикат	Действия	№ перехода
1		вывод "Destructor Object name: ", значение name, " Copy: ", значение copyNumber, переход на новую строку	2
2		очистка intArray оператором delete	∅

3.3 Алгоритм метода setSharedValue класса Object

Функционал: установка общедоступного свойства.

Параметры: целое, value, новое значение свойства.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода setSharedValue класса Object

№	Предикат	Действия	№ перехода
1		sharedValue присвоить значение value	∅

3.4 Алгоритм метода getCopyNumber класса Object

Функционал: возвращает номер копии.

Параметры: нет.

Возвращаемое значение: целое, номер копии объекта.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода getCopyNumber класса Object

№	Предикат	Действия	№ перехода
1		возвращает значение copyNumber	∅

3.5 Алгоритм метода printState класса Object

Функционал: вывод свойств объекта.

Параметры: нет.

Возвращаемое значение: нет.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода printState класса Object

№	Предикат	Действия	№ перехода
1		вывод "Object name: ", значение name, " Copy: ", значение copyNumber, " Shared: ", значение sharedValue	2
2		вывод "Array:"	3
3		инициализация целочисленной переменной i со значением 0	4
4	i < arraySize	вывод " ", intArray[i]	5
			6
5		инкремент i	4
6		переход на новую строку	∅

3.6 Алгоритм конструктора класса Object

Функционал: конструктор копии.

Параметры: Object, other, копируемый объект.

Алгоритм конструктора представлен в таблице 6.

Таблица 6 – Алгоритм конструктора класса Object

№	Предикат	Действия	№ перехода
1		полю name присвоить значение other.name	2
2		полю copyNumber присвоить значение other.copyNumber + 1	3
3		полю arraySize присвоить значение other.arraySize	4
4		полю sharedValue присвоить значение other.sharedValue	5
5		инкремент globalIndex	6
6	copyNumber < globalIndex?	copyNumber присвоить значение globalIndex	7
			7
7		полю intArray присвоить целочисленный массив размера arraySize	8
8		инициализация целочисленной переменной i со значением 0	9
9	i < arraySize?	intArray[i] присвоить значение other.intArray[i] + copyNumber	10
			11
10		инкремент i	9
11		вывод "Constructor copy Object name: ", значение name, " Copy: ", copyNumber, переход на новую строку	∅

3.7 Алгоритм функции processFunction

Функционал: вызов метода объекта.

Параметры: Object, obj.

Возвращаемое значение: нет.

Алгоритм функции представлен в таблице 7.

Таблица 7 – Алгоритм функции processFunction

№	Предикат	Действия	№ перехода
1		вызов метода printState объекта obj	∅

3.8 Алгоритм функции main

Функционал: главная функция программы.

Параметры: нет.

Возвращаемое значение: целое, идентификтор работоспособности программы.

Алгоритм функции представлен в таблице 8.

Таблица 8 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		объявление строковых переменных objectName, cmd	2
2		объявление целочисленной переменной copyNum	3
3		ввод значения objectName	4
4		инициализация переменной original созданием объекта класса Object с параметров objectName	5
5		вызов метода printState объекта original	6
6		объявление массива objects класса Object*	7
7		вызов метода push_back с параметром original для objects	8
8	бесконечный цикл?	ввод значения cmd	9
			∅
9	cmd == "end"?	прерывание цикла оператором break	10
			10
10		ввод значения copyNum	11
11		инициализация логической переменной found	12

№	Предикат	Действия	№ перехода
		значением false	
12		инициализация target класса Object* нулевым указателем	13
13	прохождение по диапазону objects		14
			17
14	значение вызова метода getCopyNumber объекта obj равен значению copyNum	значению target присвоить значение obj	15
			13
15		значению found присвоить true	16
16		прерывание цикла оператором break	17
17	found == false?	вывод "A copy of object number ", значение copyNum, " was not found.", переход на новую строку	18
			19
18		переход на следующую итерацию цикла оператором continue	8
19	cmd == "copy"?	инициализация newCopy объектом класса Object* с параметром target	20
			22
20		вызов метода push_back с параметром newCopy для objects	21
21		вызов метода printState объекта newCopy	8
22	cmd == "function"?	вызов функции processFunction с параметром target	8
			23
23	cmd == "state"?	вызов метода printState объекта	8
			24

№	Предикат	Действия	№ перехода
24	cmd == "shared"?	объявление целочисленной переменной newValue	25
			8
25		ввод значения newValue	26
26		вызов метода setSharedValue с параметром newValue объекта target	8

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

Представим описание алгоритмов в графическом виде на рисунках 1-11.

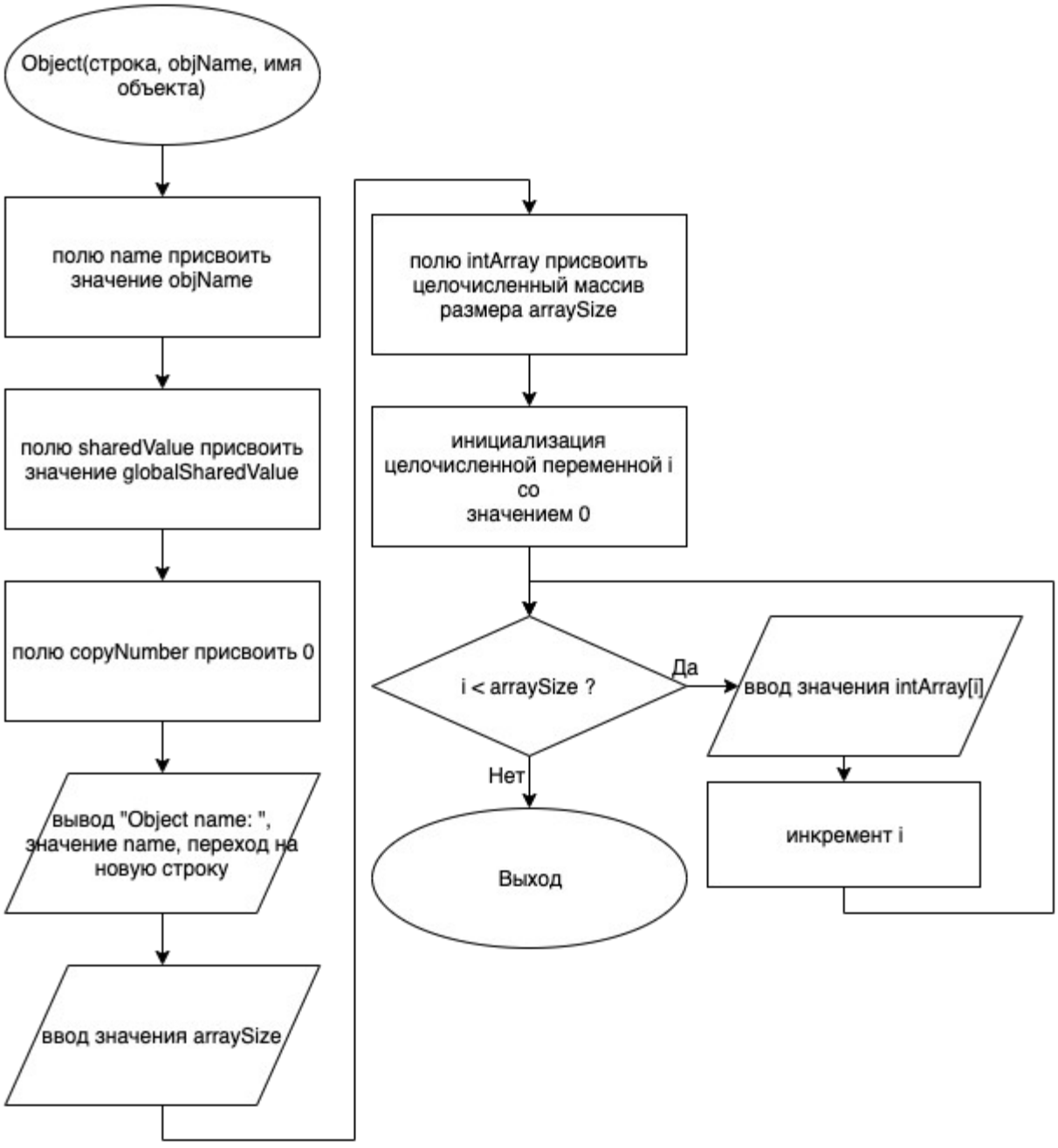

Рисунок 1 – Блок-схема алгоритма

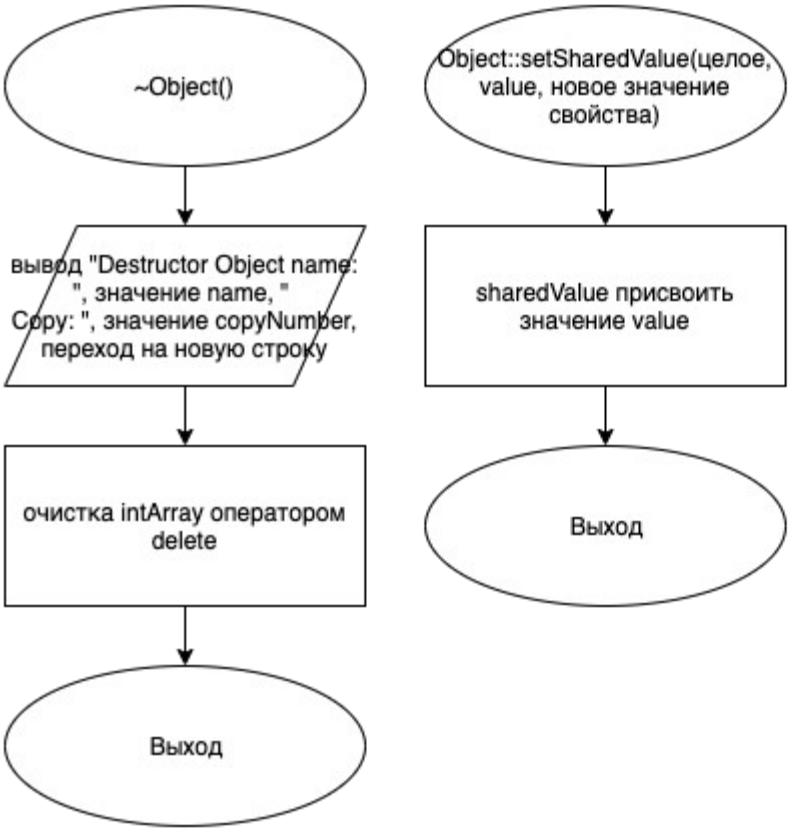

Рисунок 2 – Блок-схема алгоритма

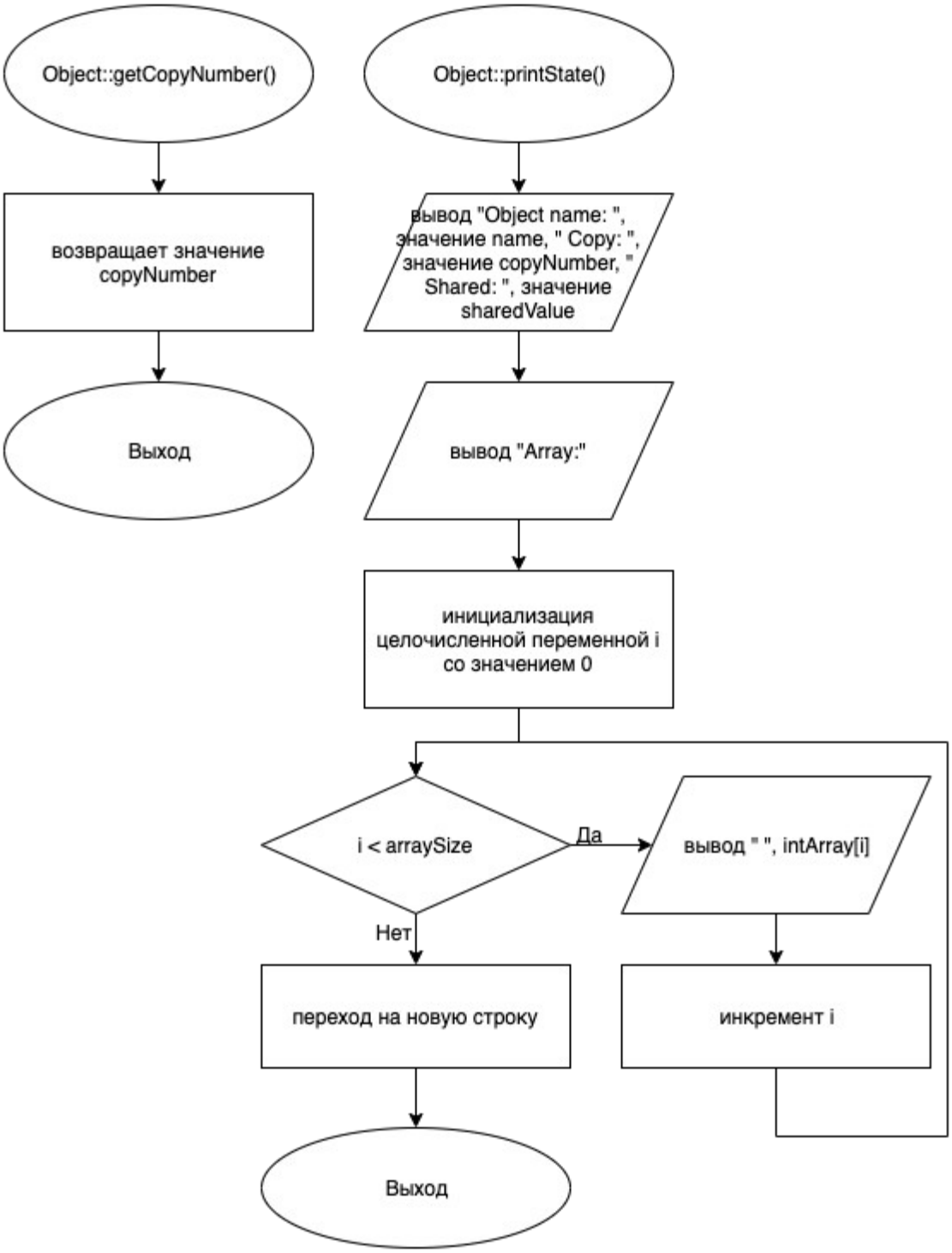

Рисунок 3 – Блок-схема алгоритма

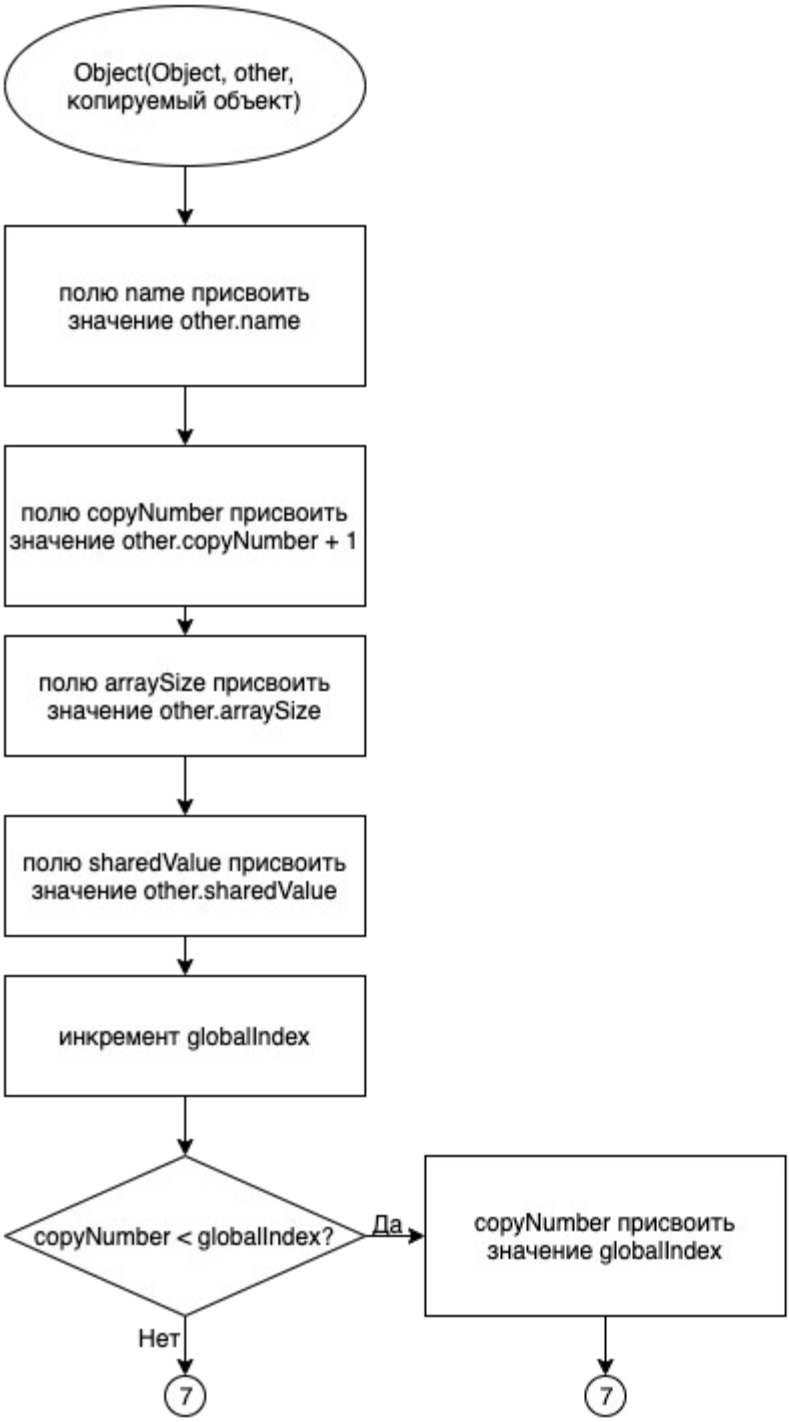

Рисунок 4 – Блок-схема алгоритма

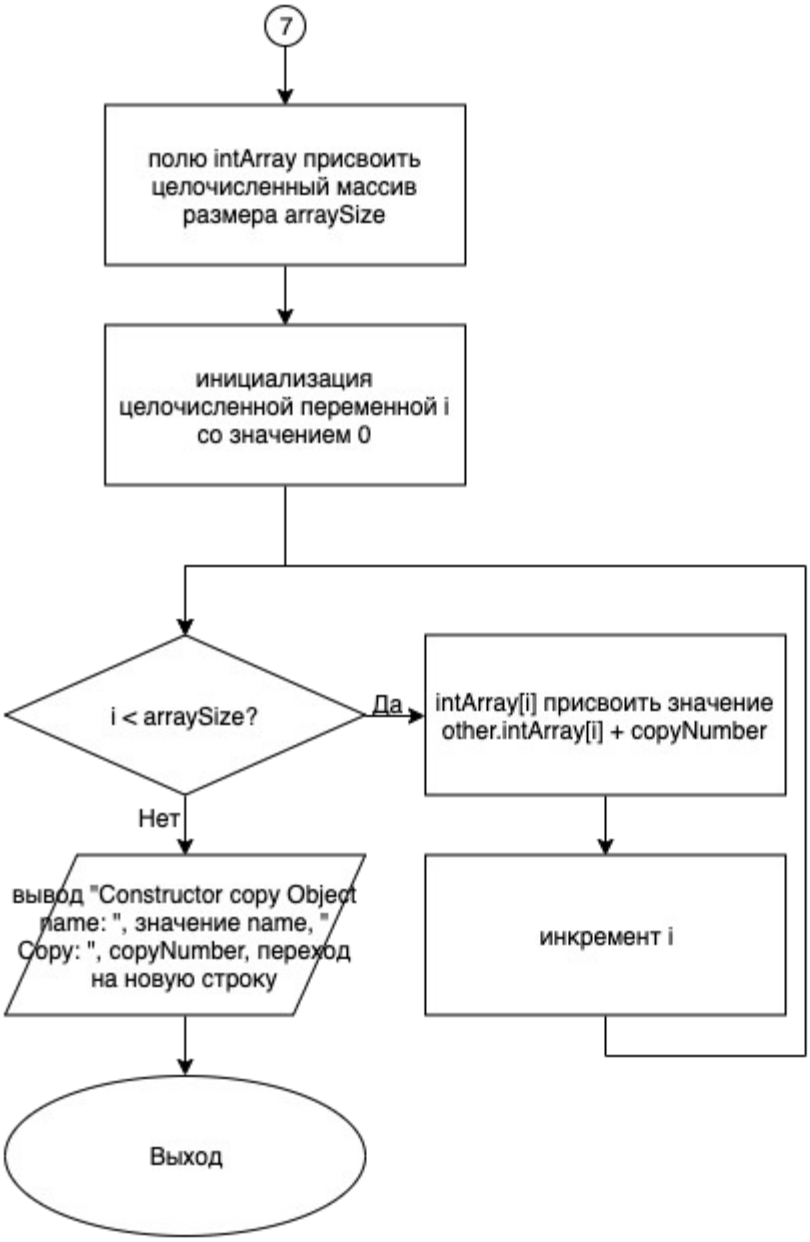

Рисунок 5 – Блок-схема алгоритма

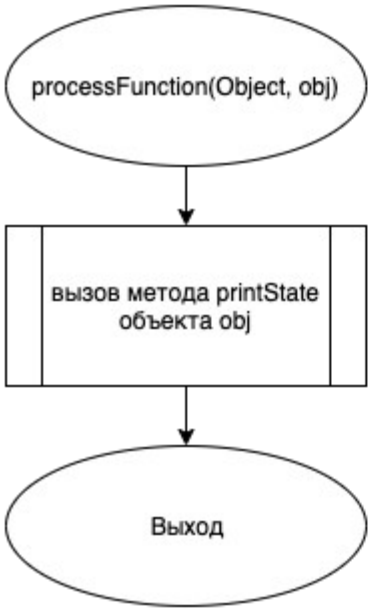

Рисунок 6 – Блок-схема алгоритма

Рисунок 7 – Блок-схема алгоритма

Рисунок 8 – Блок-схема алгоритма

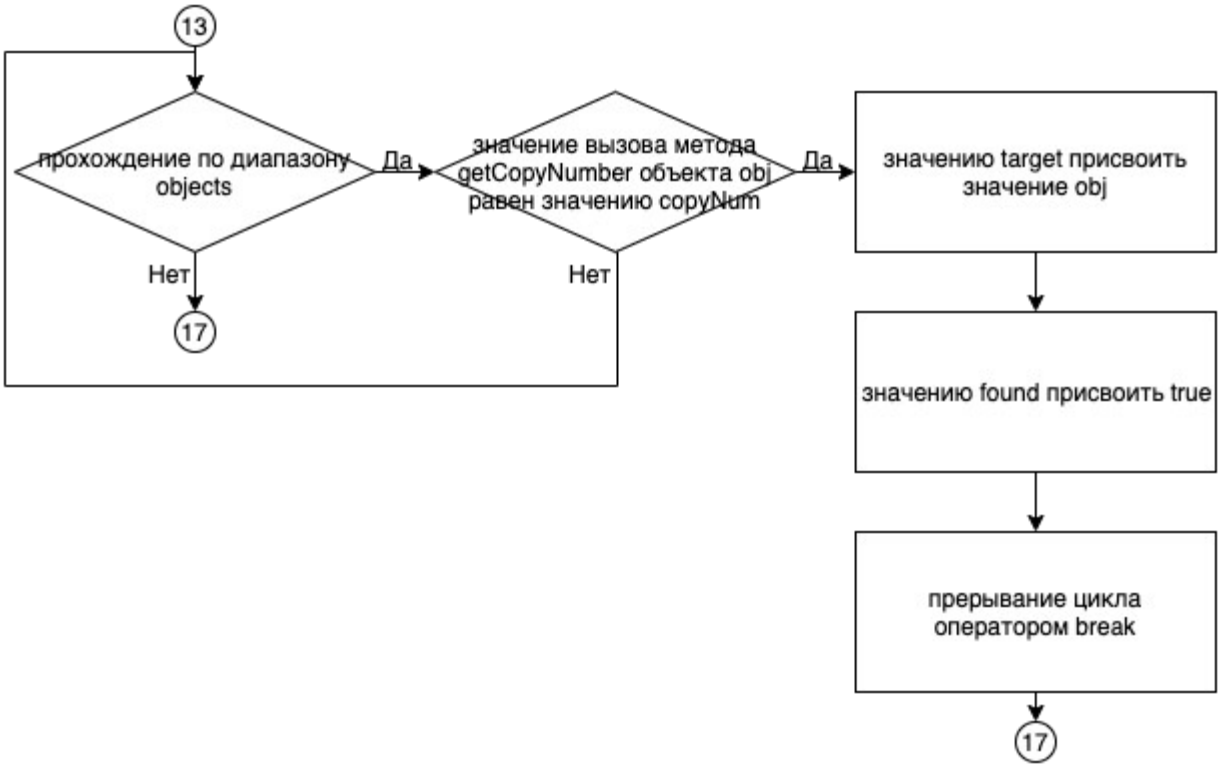

Рисунок 9 – Блок-схема алгоритма

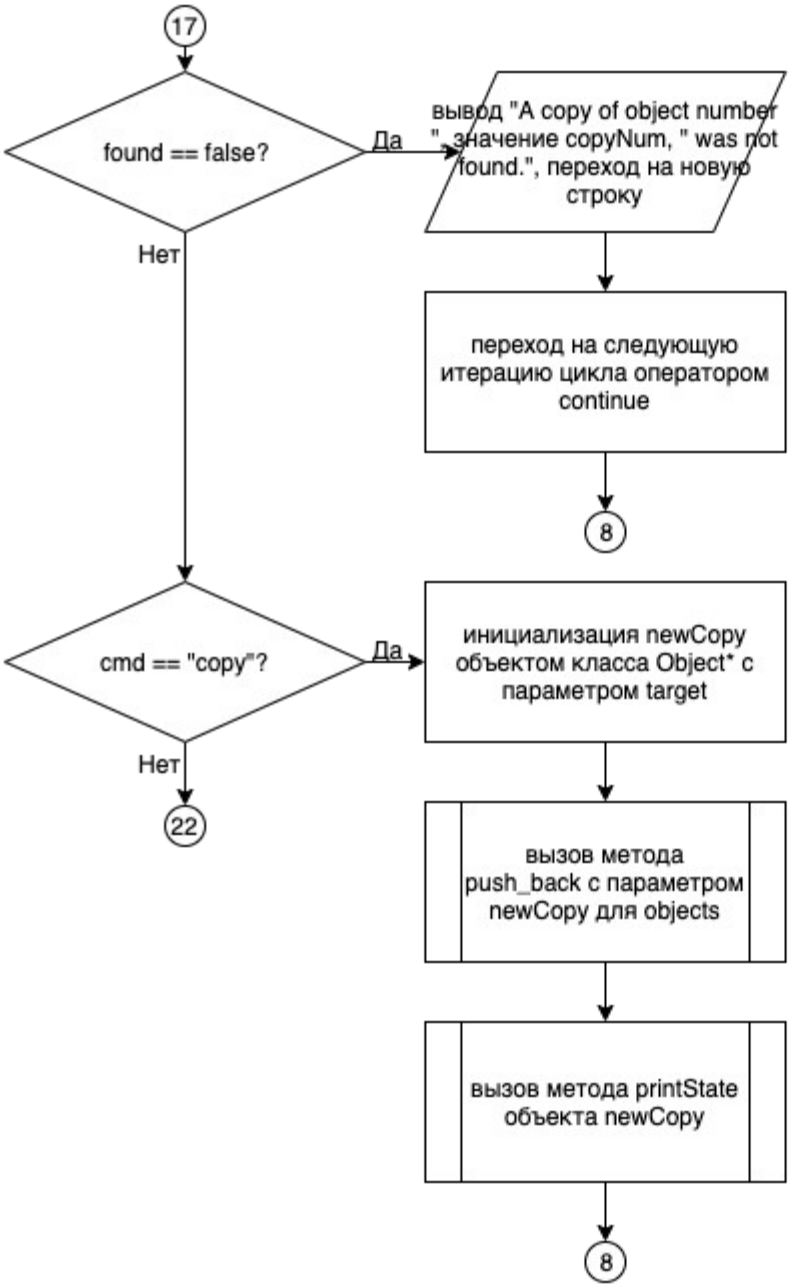

Рисунок 10 – Блок-схема алгоритма

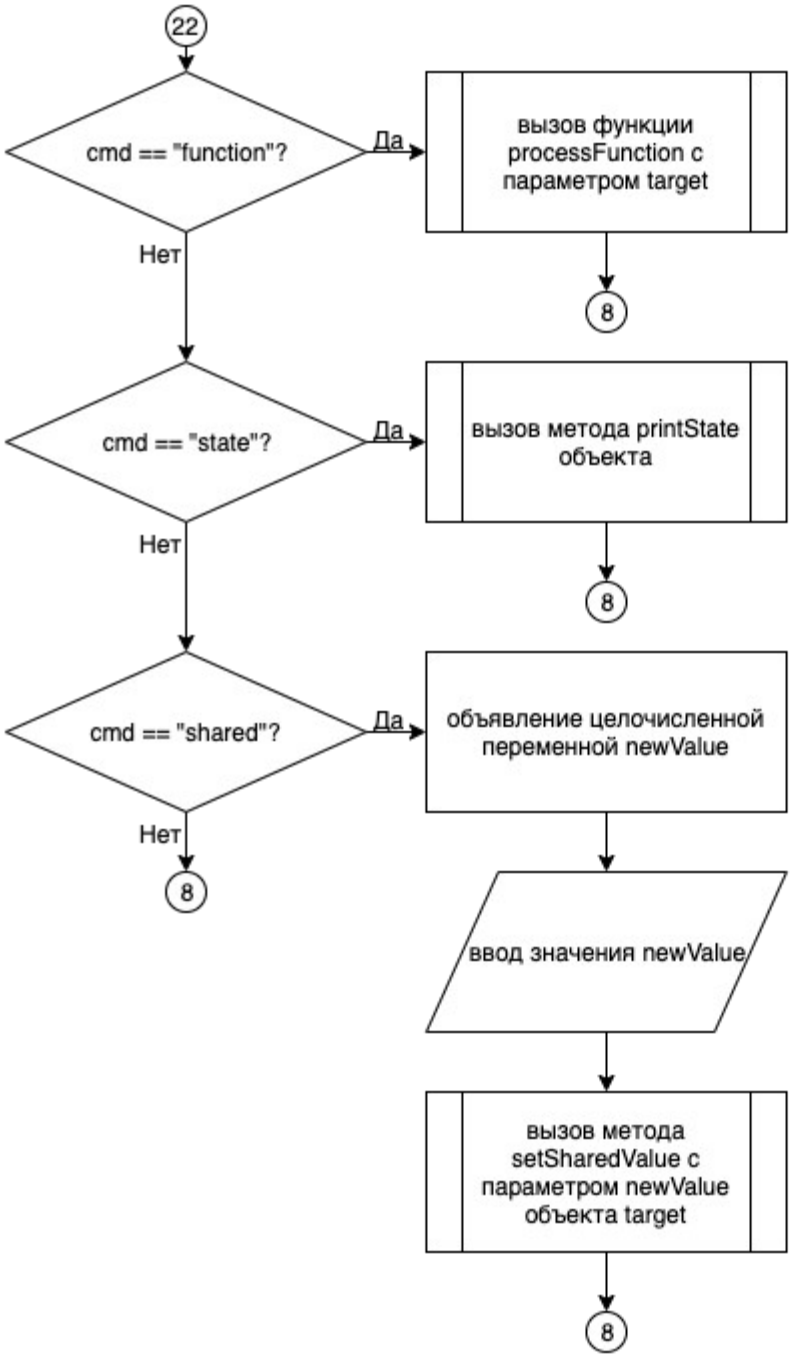

Рисунок 11 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.1 Файл main.cpp

Листинг 1 – main.cpp

```
#include <stdlib.h>
#include <stdio.h>
#include "Object.h"
#include <iostream>
#include <vector>
#include <string>
#include <limits>
using namespace std;

void processFunction(Object obj){
   obj.printState();
}


int main(){

   string objectName,cmd;
   int copyNum;
   cin>>objectName;
   Object* original = new Object(objectName);
   original->printState();

   vector<Object*> objects;
   objects.push_back(original);

   while(true){
      cin>>cmd;

      if (cmd == "end"){
         break;
      }

      cin>> copyNum;
      bool found = false;
      Object* target = nullptr;

      for (auto obj:objects){
         if (obj->getCopyNumber() == copyNum){
            target = obj;
```

```
                found = true;
                break;
            }
        }

        if (!(found)){
            cout<<"A copy of object number "<<copyNum<<" was not found."<<endl;
            cin.ignore(numeric_limits<streamsize>::max(),'\n');
            continue;
        }

        if (cmd == "copy"){
            Object* newCopy = new Object(*target);
            objects.push_back(newCopy);
            newCopy->printState();
        } else if(cmd == "function"){
            processFunction(*target);
        } else if (cmd == "state"){
            target->printState();
        } else if (cmd == "shared") {
            int newValue;
            cin>>newValue;
            target->setSharedValue(newValue);
        }
    }
    return(0);
}
```

5.2 Файл Object.cpp

Листинг 2 – Object.cpp

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

using namespace std;

int globalSharedValue = 0;
int globalIndex = 0;

Object::Object(const string &objName):
    name(objName),
    sharedValue (globalSharedValue),
    copyNumber(0){

    cout<<"Object name: "<<name<<endl;
    cin>>arraySize;
    intArray = new int[arraySize];

    for (int i=0; i<arraySize;i++){
```

```
            cin>>intArray[i];
         }
      }

      Object::Object(const Object& other):
         name(other.name),
         copyNumber(other.copyNumber+1),
         arraySize(other.arraySize),
         sharedValue(other.sharedValue){
         globalIndex++;

         if(copyNumber< globalIndex){
            copyNumber = globalIndex;
         }

         intArray = new int[arraySize];
         for (int i=0; i<arraySize;i++){
            intArray[i] = other.intArray[i] + copyNumber;
         }

         cout << "Construcrot copy Object name: " << name << "      Copy: " <<
      copyNumber << endl;
      }

      Object::~Object(){
         cout << "Destructor Object name: " << name << "    Copy: "<< copyNumber <<
      endl;
         delete[] intArray;
      }

      void Object::setSharedValue(int value){
         sharedValue = value;
      }

      int Object::getCopyNumber() const{
         return copyNumber;
      }

      void Object::printState() const{
         cout << "Object name: " << name << "      Copy: " << copyNumber << "
      Shared: " << sharedValue << endl;
         cout << "Array:";
         for (int i=0; i<arraySize;i++){
            cout << "  " << intArray[i];
         }
         cout<<endl;
      }
```

5.3 Файл Object.h

Листинг 3 – Object.h

```
#ifndef __OBJECT__H
#define __OBJECT__H

#include <string>
#include <vector>

class Object{
private:
   std::string name;
   int copyNumber;
   int arraySize;
   int* intArray;
   int& sharedValue;
public:
   Object(const std::string &objName);
   Object(const Object& other);
   ~Object();
   void setSharedValue(int value);
   int getCopyNumber() const;
   void printState() const;
};

#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 9.

Таблица 9 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
Document 5 1 2 3 4 5 copy 0 copy 1 shared 8 99 copy 1 shared 2 77 copy 0 function 1 state 5 state 1 end	Object name: Document Object name: Document Copy: 0 Shared: 0 Array: 1 2 3 4 5 Construcrot copy Object name: Document Copy: 1 Object name: Document Copy: 1 Shared: 0 Array: 2 3 4 5 6 Construcrot copy Object name: Document Copy: 2 Object name: Document Copy: 2 Shared: 0 Array: 4 5 6 7 8 A copy of object number 8 was not found. Construcrot copy Object name: Document Copy: 3 Object name: Document Copy: 3 Shared: 0 Array: 5 6 7 8 9 Construcrot copy Object name: Document Copy: 4 Object name: Document Copy: 4 Shared: 77 Array: 5 6 7 8 9 Construcrot copy Object name: Document Copy: 5 Object name:	Object name: Document Object name: Document Copy: 0 Shared: 0 Array: 1 2 3 4 5 Construcrot copy Object name: Document Copy: 1 Object name: Document Copy: 1 Shared: 0 Array: 2 3 4 5 6 Construcrot copy Object name: Document Copy: 2 Object name: Document Copy: 2 Shared: 0 Array: 4 5 6 7 8 A copy of object number 8 was not found. Construcrot copy Object name: Document Copy: 3 Object name: Document Copy: 3 Shared: 0 Array: 5 6 7 8 9 Construcrot copy Object name: Document Copy: 4 Object name: Document Copy: 4 Shared: 77 Array: 5 6 7 8 9 Construcrot copy Object name: Document Copy: 5 Object name:

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
	Document Copy: 5 Shared: 77 Array: 7 8 9 10 11 Destructor Object name: Document Copy: 5 A copy of object number 5 was not found. Object name: Document Copy: 1 Shared: 77 Array: 2 3 4 5 6	Document Copy: 5 Shared: 77 Array: 7 8 9 10 11 Destructor Object name: Document Copy: 5 A copy of object number 5 was not found. Object name: Document Copy: 1 Shared: 77 Array: 2 3 4 5 6

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).