Здесь будет титульник, листай ниже

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель курсовой работы: моделирование работы кофемашины с помощью сигналов и обработчиков.

Для достижения поставленной цели необходимо выполнить следующие задачи.

Задачи:

Освоение принципов объектно - ориентированного программирования

Освоение основ объектно- ориентированного языка программирования C++

Освоение разработки программы как система

Освоение умения проектирования архитектуры программы на базе построения иерархии обектов

Освоение выполнения всех необходимых работ согласно этапам разработки программы и соответствующих программных инструментов

Моделирование работы кофемашины при помощи сигнаов и обработчиков

Построение системы взаимодействия объектов с помощью интерфейса сигналов и обработчиков

Опсиание алгоритма работы программы

Построение кода на языке программирования C++, согласно разработанному алгоритму работы программы

Тестирование работы программы

1 ПОСТАНОВКА ЗАДАЧИ

Надо моделировать работу кофемашины следующей конструкции. Кофе машина состоит из следующих элементов:

• пульт управления;

• устройство приема денег;

• устройство выдачи кофе;

• устройство возврата сдачи;

• экран отображения состояния и информации.

Пульт управления содержит кнопки:

• выбора кофе (множество кнопок);

• возврата денег.

Правила работы с кофе машиной.

Кофе машина готовиться к работе следующим образом:

1. Задается количество сортов кофе (количество кнопок для выбора кофе) их названия и их стоимость, кратное 5 рублям. Загружается кофе. Подразумевается, что объем достаточен для работы.

2. Загружается заданное количество монет для выдачи сдачи с достоинством пять и десять рублей.

3. После этого выводиться сообщение о готовности кофе машины к работе.

После готовности кофе машины выполняются действия:

• ввод денег достоинством 5, 10, 50 или 100 рублей. При вводе денег осуществляется суммирование;

• выборе кофе, если денег достаточно, то выдается кофе и сдача, при наличии. Выводиться сообщение о готовности кофе машины к работе.

• выборе кофе, если денег недостаточно, сообщает о недостаточности средств.

• возврат денег, возвращаются все внесенные средства и выводиться сообщение о готовности кофе машины к работе.

Устройство возврата сдачи может вернуть только монеты с достоинством 5 и 10 рублей.

После ввода купюр достоинством 50 или 100 рублей проверяется возможность возврата внесенной суммы. Если монет с достоинством 5 и 10 рублей недостаточно, то купюры 50 или 100 не принимаются.

Возврат денег или сдача выдается максимальным количеством монет достоинством 10 рублей.

Нажатие на кнопки пульта управления и подача денег моделируется посредством клавиатурного ввода. Ввод делится на команды:

• «натуральное число кратное 5» - ввод денег.

• Coffee «наименование кофе» - нажатье кнопки сорта кофе (выбора кофе).

• Refund money – нажатье кнопки «вернуть деньги».

• Cancel – завершение работы системы.

Отображение текста состояния кофе машины и результата операции моделируется посредством вывода на консоли.

Построить систему, которая использует объекты:

1. Объект «система».

2. Объект для чтения команд и данных. Считывает данные для подготовки и настройки кофе машины. После чтения очередной порции данных для настройки или данных команды, объект выдает сигнал с текстом полученных данных. Все данные настройки и данные команды синтаксический корректны.

3. Объект пульта управления, для отработки нажатия кнопок выбора кофе. Объект после нажатия кнопки анализирует достаточность средств и выдает

соответствующий сигнал.

4. Объект, моделирующий устройства приема денег. После принятия очередной купюры производит суммирование и выдает сигнал, содержащий сумму введенных денег для отображения на экран.

5. Объект, моделирующий устройства возврата денег. Выдает сигнал, содержащий количество возвращаемой суммы. После выводиться сообщение о готовности кофе машины к работе.

6. Объект, моделирующий устройства выдачи кофе. Выдает сигнал, содержащий текст . После выдачи кофе выдает сигнал о готовности кофе машины к работе.

7. Объект для вывода состояния или результата операции кофе машины на консоль.

Написать программу, реализующую следующий алгоритм:

1. Вызов метода объекта «система» build_tree_objects ( ).

1.1. Построение дерева иерархии объектов.

1.2. Установка связей сигналов и обработчиков между объектами.

2. Вызов метода объекта «система» exec_app ( ).

2.1. Приведение всех объектов в состояние готовности.

2.2. Цикл для обработки вводимых данных для настройки и команд.

2.2.1. Выдача сигнала объекту для ввода команды.

2.2.2. Отработка команды.

2.3. После ввода команды «Cancel» завершить работу.

Все приведенные сигналы и соответствующие обработчики должны быть реализованы.

Все сообщения на консоль выводятся с новой строки.

В набор поддерживаемых команд добавить команду «SHOWTREE» и по этой команде вывести дерево иерархии объектов системы с отметкой о готовности и завершить работу программы.

1.1 Описание входных данных

Первая строка.

«натуральное число» «название кофе 1» … «название кофе n»

Задает количество сортов кофе и их наименования в количестве не более 5. Выполняется операция загрузки кофе.

Вторая строка содержит целые числа кратные 5 в количестве сортов кофе. Каждое значение соответствует цене сорта кофе согласно индексу (порядку ввода). Выполняется операция настройки цен.

«натуральное число» «натуральное число» ….

Третья строка содержит исходное количество монет для выдачи сдачи. Выполняется операция первоначальной загрузки монет для сдачи.

«натуральное число» «натуральное число»

Первое число - количество пяти рублевых монет, второе число - количество десяти рублеых монет.

Последующие строки содержат команды (нажатия на кнопки или подача денег).

Подача денег, число 5, 10, 50 или 100

«натуральное число»

Возврат денег

Refund money

Выбор кофе

Coffee «наименование кофе»

Последняя команда присутствует всегда

Cancel

Пример ввода:

3 Espresso Americano Cappuchino

25 50 50

3 5

50

Coffee Cappuchino

10

10

10

Coffee Espresso

5

5

Refund money

5

100

Cancel

1.2 Описание выходных данных

Шаблоны текстов, которые отображаются на консоли:

Готов к работе, отображения в начале работы системы, после завершения загрузки кофе машины. Также отображается после завершения очередной операции и готовности кофе машины для обслуживания нового клиента.

Ready to work

Сумма после ввода очередной монеты или купюры.

The amount: «сумма денег»

Сообщение о готовности кофе

Take the coffee «наименование кофе»

Сообщение для получения сдачи:

Take the change: 10 * «количество десяти рублевых монет» rub., 5 * «количество пяти рублевых монет» rub.

Сообщение о недостаточности средств

There is not enough money

Сообщение для получения введенных денег обратно:

Take the money: 10 * «количество десяти рублевых монет» rub., 5 * «количество пяти рублевых монет» rub.

Сообщение о возврате 50 или 100 рублевой купюры :

Take the money back, no change

Сообщение о завершении работы кофе машины:

Turned off

Пример вывода

Ready to work

The amount: 50

Take the coffee Cappuchino

Ready to work

The amount: 10

The amount: 20

The amount: 30

Take the coffee Espresso

Take the change: 10 * 0 rub., 5 * 1 rub.

Ready to work

The amount: 5

The amount: 10

Take the money: 10 * 1 rub., 5 * 0 rub.

Ready to work

The amount: 5

Take the money back, no change

Turned off

2 МЕТОД РЕШЕНИЯ

Для решения поставленной задачи используются:

Для работы с входными и выходными потоками данных используем библиотеку <iostream>

Библиотека string для работы со троковыми типами данных

Библиотеа vector

Операторы цикла for и while

Класс base:

1. Поля

Вектор ncoffe типа string - хранит названия кофе ;

Вектор pcoffe типа int - хранит цены на кофе ;

Целочисленные summ, fi, te - хранят количество введенных денег, количество монет номиналом 5, количество денег номиналом 10

Класс B:

- Методы

void bild_tree_objects(). Функционал - построение дерева и установка связей между объектами .

void exec_app (). Функционал - приведение объектов в статус готвоности, подача сигнала о подготовке.

Класс base_2:

5. Методы

void signal(string& now). Функционал - сигнал с данными, обработанными классом Read.

void signal_go(string& now). Функционал - сигнал старта, подготовка кофе-машины к работе.

void handler(string now). Функционал - обработка вводимых команд и

данных.

Класс base_3:

3. Методы

void signal(string& now). Функционал - сигнал от пульта управления .

void handler(string now). Функционал - обработка данных и анализ достаточности средств .

Класс base_4:

Методы

void signal(string& now). Функционал - сигнал содержащий сумму введеных денег .

void handler(string now). Функционал - производит суммирование .

Класс base_5:

Методы

void signal(string& now). Функционал -сигнал с количеством возвращаемой суммы .

void handler(string now). Функционал - обработка полученных данных.

Класс base_6:

Методы

void signal(string& now). Функционал -вывод полученных данных.

Класс base_7:

Методы

void signal(string& now). Функционал -сигнал с текстом выдачи кофе .

void handler(string now). Функционал -обработка сигнала с пульта .

Таблица 1 – Иерархия наследования классов

№	Имя класса	Классы-наследники	Модификатор доступа	Описание	Номер	Комментарий

			при наследовании			
1	base			Базовый класс с основными полями и методами		
		B	public		2	
		base_2	public		3	
		base_3	public		4	
		base_4	public		5	
		base_5	public		6	
		base_6	public		7	
		base_7	public		8	
2	B			Класс "система"		
3	base_2			Класс считывающий команды и данные		
4	base_3			Класс пульта управления		
5	base_4			Класс устройства приема		

				денег		
6	base_5			Класс устройства возврата денег		
7	base_6			Класс вывода состояния или результата операции		
8	base_7			Класс устройства выдачи кофе		

3 ОПИСАНИЕ АЛГОРИТМОВ

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

3.0 Алгоритм метода bild_tree_objects класса B

Функционал: Построение дерева и установка связей между объектами.

Параметры: .

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 2.

Таблица 2 – Алгоритм метода bild_tree_objects класса B

№	Предикат	Действия	№ перехода
1		Создание объекта класса base_2 через параметризированный конструктор с параметрами: строка "Read" и указател на объект this	2
2		Создание объекта класса base_3 через параметризированный конструктор с параметрами: строка "Pult" и указател на объект this	3
3		Создание объекта класса base_4 через параметризированный конструктор с параметрами: строка "Set" и указател на объект this	4
4		Создание объекта класса base_5 через параметризированный конструктор с параметрами: строка "Get" и указател на объект this	5
5		Создание объекта класса base_6 через параметризированный конструктор с параметрами: строка "Print" и указател на объект this	6
6		Создание объекта класса base_7 через параметризированный конструктор с параметрами: строка "CoffeeGet" и указател на объект this	7
7		Вызов метода set_sv объекта "Sys"для установки связи с объектом "Read" через методы signal класса base_2 и handler класса base_2	8

№	Предикат	Действия	№ перехода
8		Вызов метода set_sv объекта "Read"для установки связи с объектом "Set" через методы signal класса base_2 и handler класса base_4	9
9		Вызов метода set_sv объекта "Set"для установки связи с объектом "Print" через методы signal класса base_4 и handler класса base_6	10
10		Вызов метода set_sv объекта "Read"для установки связи с объектом "Print" через методы signal класса base_2 и handler класса base_6	11
11		Вызов метода set_sv объекта "Sys"для установки связи с объектом "Print" через методы signal_go класса base_2 и handler класса base_6	12
12		Вызов метода set_sv объекта "Read"для установки связи с объектом "Get" через методы signal класса base_2 и handler класса base_5	13
13		Вызов метода set_sv объекта "Get"для установки связи с объектом "Print" через методы signal класса base_5 и handler класса base_6	14
14		Вызов метода set_sv объекта "Read"для установки связи с объектом "Pult" через методы signal класса base_2 и handler класса base_3	15
15		Вызов метода set_sv объекта "Pult"для установки связи с объектом "Print" через методы signal класса base_3 и handler класса base_6	16
16		Вызов метода set_sv объекта "Pult"для установки связи с объектом "CoffeeGet" через методы signal класса base_3 и handler класса base_7	17
17		Вызов метода set_sv объекта "CoffeeGet"для установки связи с объектом "Print" через методы signal класса base_7 и handler класса base_6	18
18		Вызов метода set_sv объекта "CoffeeGet"для установки связи с объектом "Get" через методы signal класса base_2 и handler класса base_5	∅

3.1 Алгоритм метода exec_app класса B

Функционал: Приведение объектов в статус готвоности, подача сигнала о подготовке.

Параметры: .

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 3.

Таблица 3 – Алгоритм метода exec_app класса B

№	Предикат	Действия	№ перехода
1		Вызов метода Set_status_all(1) у объекта koren	2
2		Считывание данных с клавиатуры в переменную cmd	3
3		Вызов метода get_sv объекта "Sys" с параметрами указателя на метод signal класса base_2, переменной cmd, указателем на объект "Print"	4
4	cmd не равен "Cancel" и не равен "SHOWTREE"	Считывание данных с клавиатуры в переменую cmd	5
			∅
5		Вызов метода get_sv объекта "Sys" с параметрами указателя на метод signal класса base_2, переменной cmd, указателем на объект "Read"	4

3.2 Алгоритм метода signal класса base_2

Функционал: Сигнал с данными, обработанными классом Read.

Параметры: string& now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 4.

Таблица 4 – Алгоритм метода signal класса base_2

№	Предикат	Действия	№ перехода
1	now равен "money"	В переменную ten присваиваем результат от деления поля summ объекта "Sys" на 10	2
			8
2		Из поля summ объекта "Sys" вычитаем ten*10	3
3		В переменную fife присваиваем результат от деления поля summ объекта "Sys" на 5	4
4		Из поля summ объекта "Sys" вычитаем fife*5	5
5		Из поля fi объекта "Sys" вычитаем fife	6
6		Из поля te объекта "Sys" вычитаем ten	7
7		Переменной now присваиваем значения to_string(ten)+"/"+to_string(fife)	∅
8	now равен"change"	В переменную ten присваиваем результат от деления поля summ объекта "Sys" на 10	9
			15
9		Из поля summ объекта "Sys" вычитаем ten*10	10
10		В переменную fife присваиваем результат от деления поля summ объекта "Sys" на 5	11
11		Из поля summ объекта "Sys" вычитаем fife*5	12
12		Из поля fi объекта "Sys" вычитаем fife	13
13		Из поля te объекта "Sys" вычитаем ten	14
14		Переменной now присваиваем значения to_string(ten)+"$"+to_string(fife)	∅
15	now равен "false"	Переменной now присваиваем "\nTake the money back, no change"	∅

3.3 Алгоритм метода signal_go класса base_2

Функционал: Сигнал старта, подготовка кофе-машины к работе.

Параметры: string &now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 5.

Таблица 5 – Алгоритм метода signal_go класса base_2

№	Предикат	Действия	№ перехода
1		В поле n присваиваем значение atoi(now.c_str())	2
2	i меньше n	Ввод с клавиатуры значение в переменную name	3
			4
3		Вызов метода push_back(name) вектора ncoffee объекта "Sys"	2
4	i меньше n	Ввод с клавиатуры значение в переменную p	5
			6
5		Вызов метода push_back(p) вектора pcoffee объекта "Sys"	4
6		Ввод с клавиатуры данных в поле fi объекта "Sys"	7
7		Ввод с клавиатуры данных в поле te объекта "Sys"	8
8		Переменной now присваиваем Ready to work	∅

3.4 Алгоритм метода handler класса base_2

Функционал: Обработка вводимых команд и данных.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 6.

Таблица 6 – Алгоритм метода handler класса base_2

№	Предикат	Действия	№ перехода
1	now равен "Refund"	Ввод в переменную now данных с клавиатуры	2
			3
2		Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса base_2, переменной now, указателем на объект "Get"	∅
3	now равен "Coffee"	Ввод в переменную now данных с клавиатуры	4
			5
4		Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса base_2, переменной now, указателем на объект "Pult"	∅
5	now равен "Cancel"	Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса base_2, строки "Turned off", указателем на объект "Print"	∅
			6
6	now равен "SHOWTREE"	Вызов метода print_status(0) метода "Sys"	∅
			7
7	stoi(now.c_str()) равен 50		8
			9
8	(fi*5 объекта "Sys")+(te*10 объекта "Sys") меньше 50	Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса base_2, "false", указателем на объект "Print"	∅
			11
9	stoi(now.c_str()) равен 100		10
			11
1	(fi*5 объекта "Sys")+(te*10 объекта "Sys") меньше 100	Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса	∅

№	Предикат	Действия	№ перехода
0		base_2, "false", указателем на объект "Print"	
			11
11		Вызов метода get_sv объекта "Read" с параметрами указателя на метод signal класса base_2, now, указателем на объект "Set"	∅

3.5 Алгоритм метода signal класса base_3

Функционал: Сигнал от пульта управления.

Параметры: string& now .

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 7.

Таблица 7 – Алгоритм метода signal класса base_3

№	Предикат	Действия	№ перехода
1	now равен "false"	now присваиваем "There is not enough money"	∅
			∅

3.6 Алгоритм метода handler класса base_3

Функционал: Обработка данных и анализ достаточности средств.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 8.

Таблица 8 – Алгоритм метода handler класса base_3

№	Предикат	Действия	№ перехода
1		Переменной flag присваиваем true	2
2	i меньше ncoffe.size()		3

№	Предикат	Действия	№ перехода
	объекта "Sys"		
			5
3	now равен ncoffe[i] и pcoffee[i] меньше или равен summ объекта "Sys"	Из summ объекта "Sys" вычитаем pcoffe[i] объекта "Sys"; flag присваиваем false	4
			2
4	summ Объекта "Sys" равен 0	Вызов метода get_sv this с параметрами указателя на метод signal класса base_3, now, указателем на объект "CoffeeGet"	∅
		Вызов метода get_sv this с параметрами указателя на метод signal класса base_3, now+"/"+"change", указателем на объект "CoffeeGet"	∅
5	flag равен true	Вызов метода get_sv this с параметрами указателя на метод signal класса base_3, "false", указателем на объект "Print"	∅
			∅

3.7 Алгоритм метода signal класса base_4

Функционал: Сигнал содержащий сумму введеных денег.

Параметры: string &now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 9.

Таблица 9 – Алгоритм метода signal класса base_4

№	Предикат	Действия	№ перехода
1		now присваиваем "\nThe amount: "+now	∅

3.8 Алгоритм метода handler класса base_4

Функционал: Производит суммирование.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 10.

Таблица 10 – Алгоритм метода handler класса base_4

№	Предикат	Действия	№ перехода
1		Полю Summ прибавляем stoi(now.c_str())	2
2		Вызов метода get_sv this с параметрами указателя на метод signal класса base_4, to_string(koren->GetVater("Sys")->summ), указателем на объект "Print"	∅

3.9 Алгоритм метода signal класса base_5

Функционал: Сигнал с количеством возвращаемой суммы.

Параметры: string& now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 11.

Таблица 11 – Алгоритм метода signal класса base_5

№	Предикат	Действия	№ перехода
1	now.find("/") равен -1	now присваиваем "\nTake the change: 10 * "+now.substr(0,now.find("$"))+" rub., 5 * "+now.substr(now.find("$")+1)+" rub."	∅
		now присваиваем "\nTake the money: 10 * "+now.substr(0,now.find("/"))+" rub., 5 * "+now.substr(now.find("/")+1)+" rub.\nReady to work"	∅

3.10 Алгоритм метода handler класса base_5

Функционал: Обработка полученных данных.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 12.

Таблица 12 – Алгоритм метода handler класса base_5

№	Предикат	Действия	№ перехода
1		Вызов метода get_sv this с параметрами указателя на метод signal класса base_5, now, указателем на объект "Print"	∅

3.11 Алгоритм метода handler класса base_6

Функционал: Вывод полученных данных.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 13.

Таблица 13 – Алгоритм метода handler класса base_6

№	Предикат	Действия	№ перехода
1		Вывод строки now	∅

3.12 Алгоритм метода signal класса base_7

Функционал: Сигнал с текстом выдачи кофе.

Параметры: string& now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 14.

Таблица 14 – Алгоритм метода signal класса base_7

№	Предикат	Действия	№ перехода
1	now не равен "Ready to work"	now присваиваем "Take the coffee"+now	∅
			∅

3.13 Алгоритм метода handler класса base_7

Функционал: Обработка сигнала с пульта.

Параметры: string now.

Возвращаемое значение: Нет.

Алгоритм метода представлен в таблице 15.

Таблица 15 – Алгоритм метода handler класса base_7

№	Предикат	Действия	№ перехода
1	now.find("/") равен -1	Вызов метода get_sv this с параметрами указателя на метод signal класса base_7, now, указателем на объект "Print"	3
		Вызов метода get_sv this с параметрами указателя на метод signal класса base_7,now.substr(0,now.find("/"), указателем на объект "Print"	2
2		Вызов метода get_sv this с параметрами указателя на метод signal класса base_2,now.substr(now.find("/")+1), указателем на объект "Get"	3
3		Вызов метода get_sv this с параметрами указателя на метод signal класса base_7,"Ready to work", указателем на объект "Print"	∅

3.14 Алгоритм функции main

Функционал: Осной алгоритм программы.

Параметры: .

Возвращаемое значение: 0, Целое - индикатор корректсности программы.

Алгоритм функции представлен в таблице 16.

Таблица 16 – Алгоритм функции main

№	Предикат	Действия	№ перехода
1		Создаем объект ob класса B	2
2		Вызов метода bild_tree_objects() объекта ob	3
3		Вызов метода exec_app() объекта ob	∅

4 БЛОК-СХЕМЫ АЛГОРИТМОВ

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

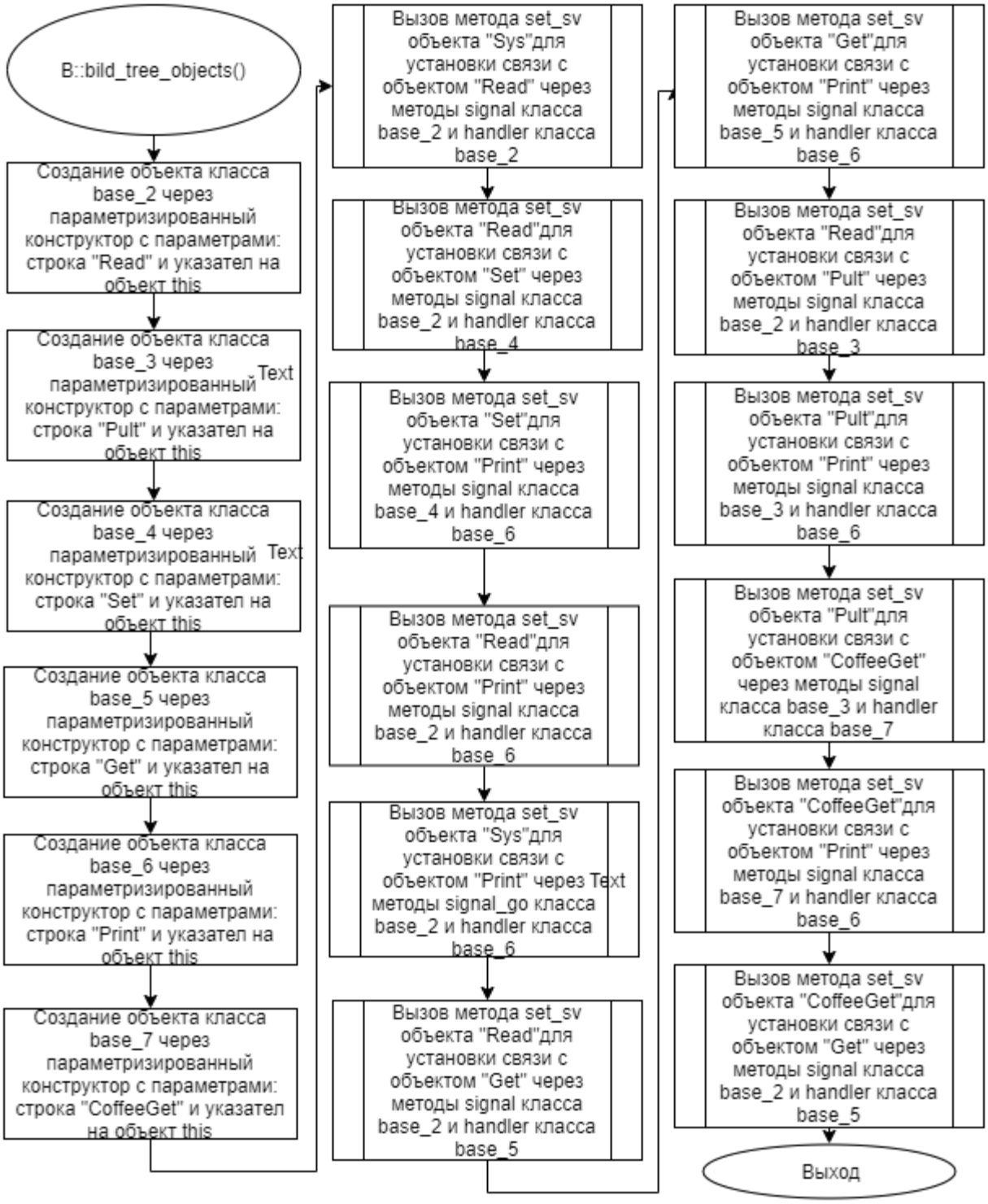

Рисунок 1 – Блок-схема алгоритма

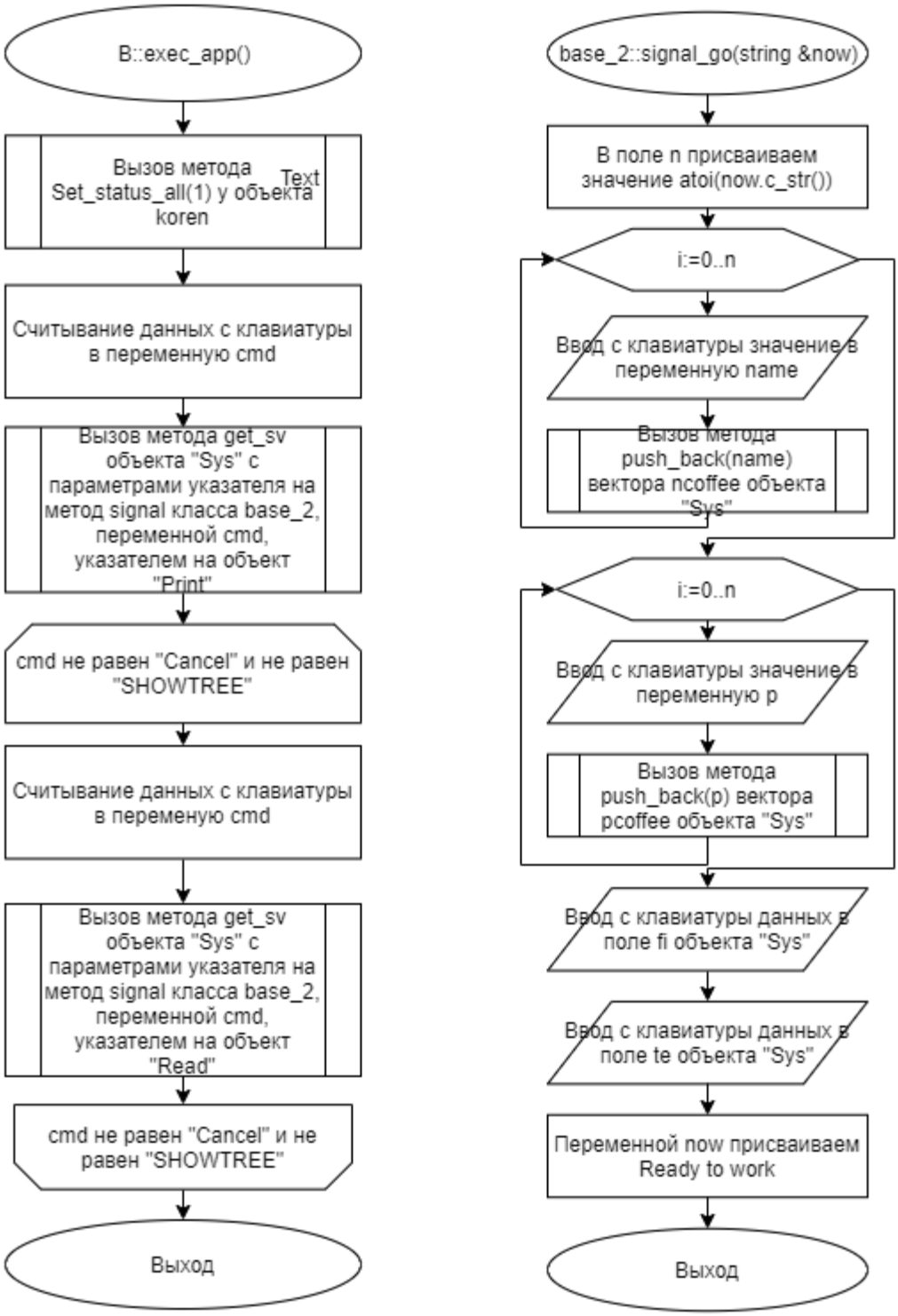

Рисунок 2 – Блок-схема алгоритма

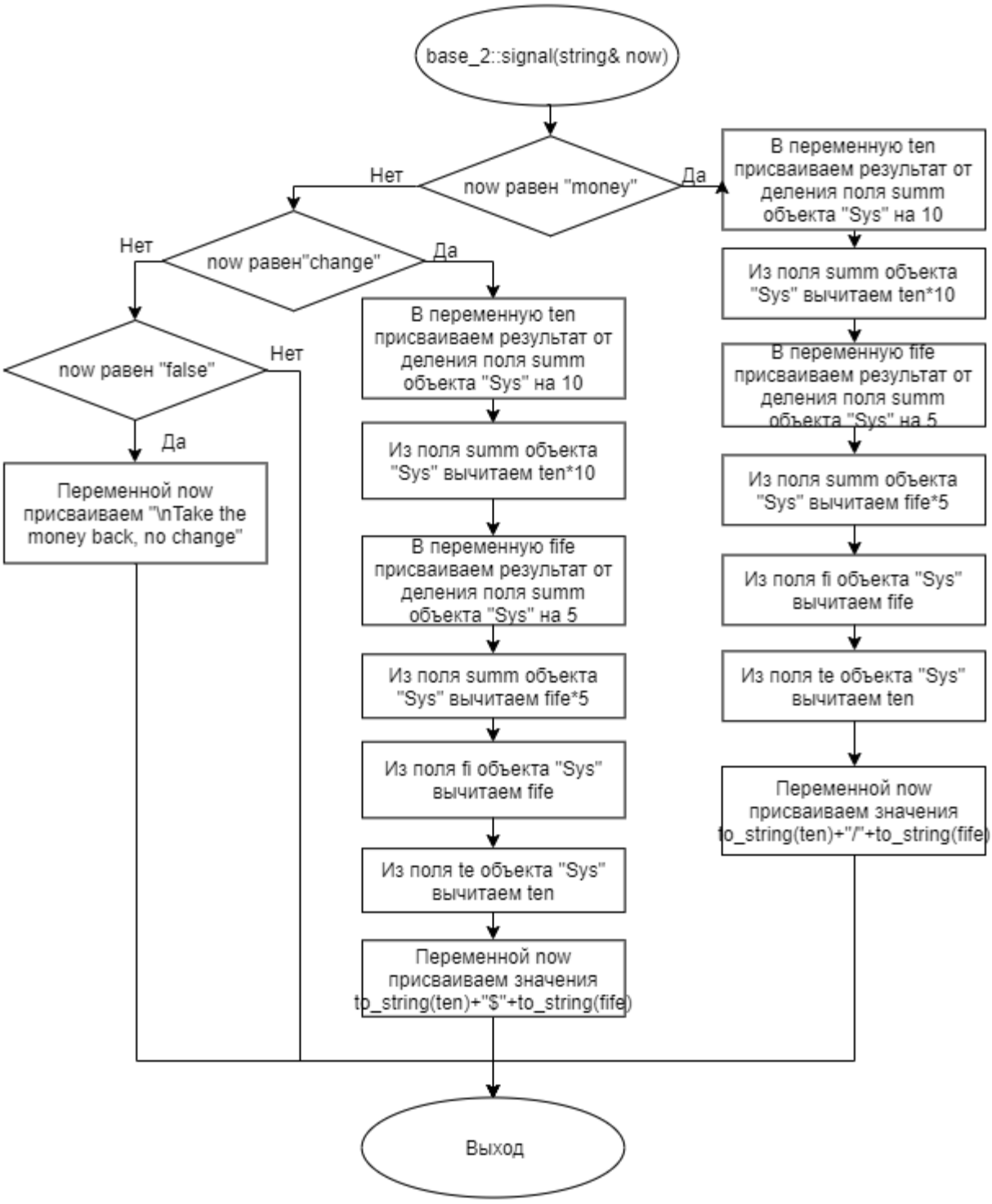

Рисунок 3 – Блок-схема алгоритма

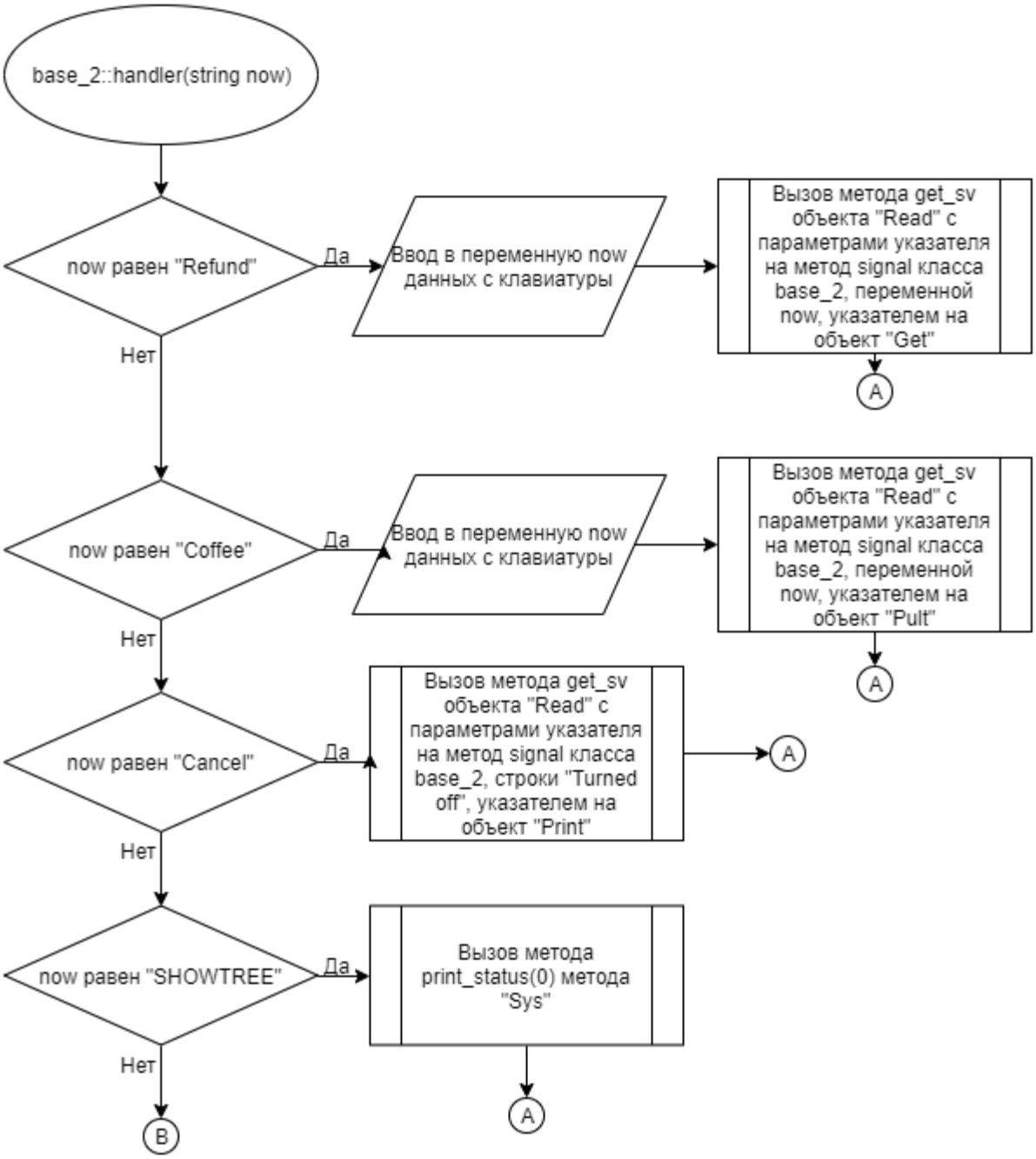

Рисунок 4 – Блок-схема алгоритма

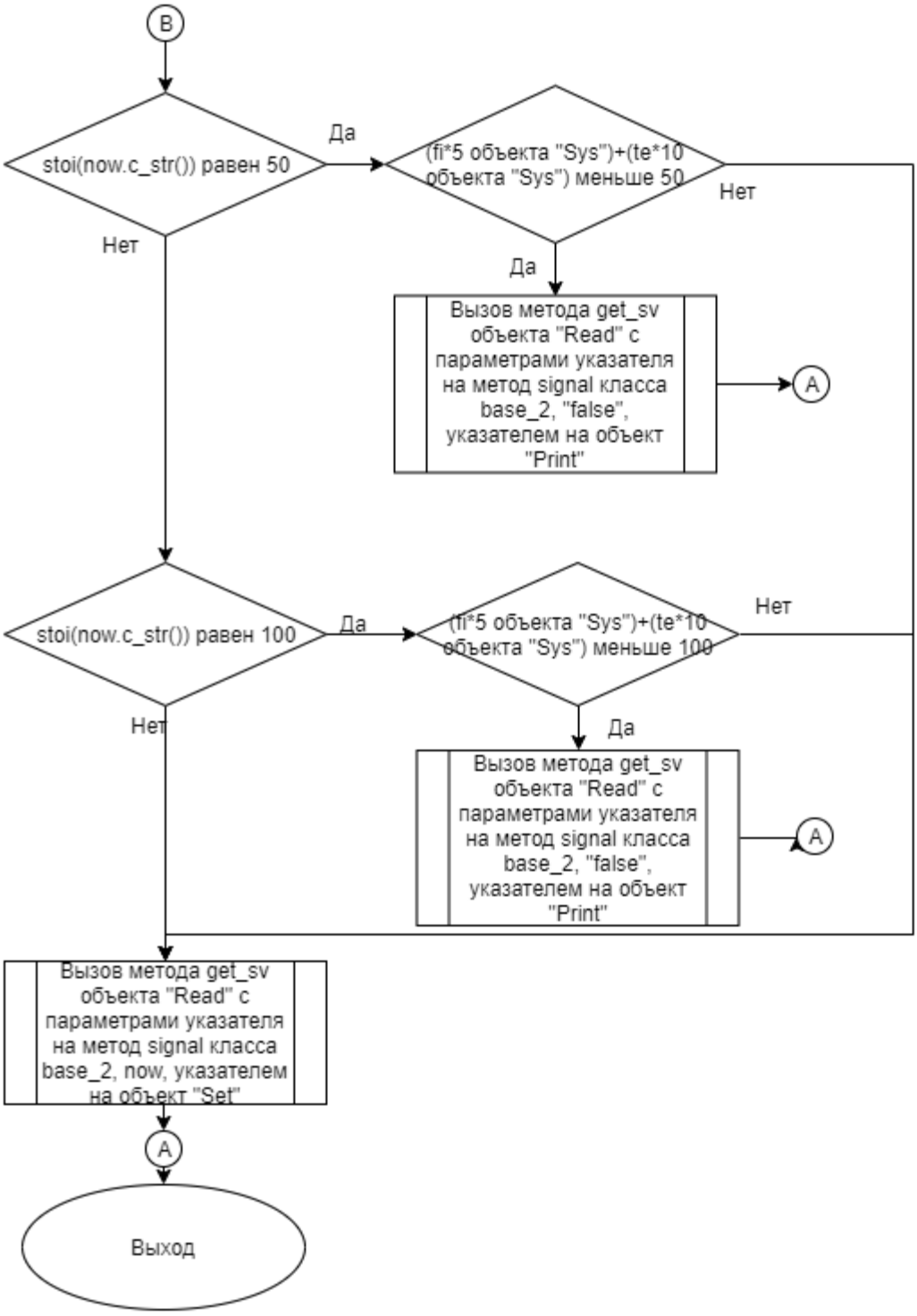

Рисунок 5 – Блок-схема алгоритма

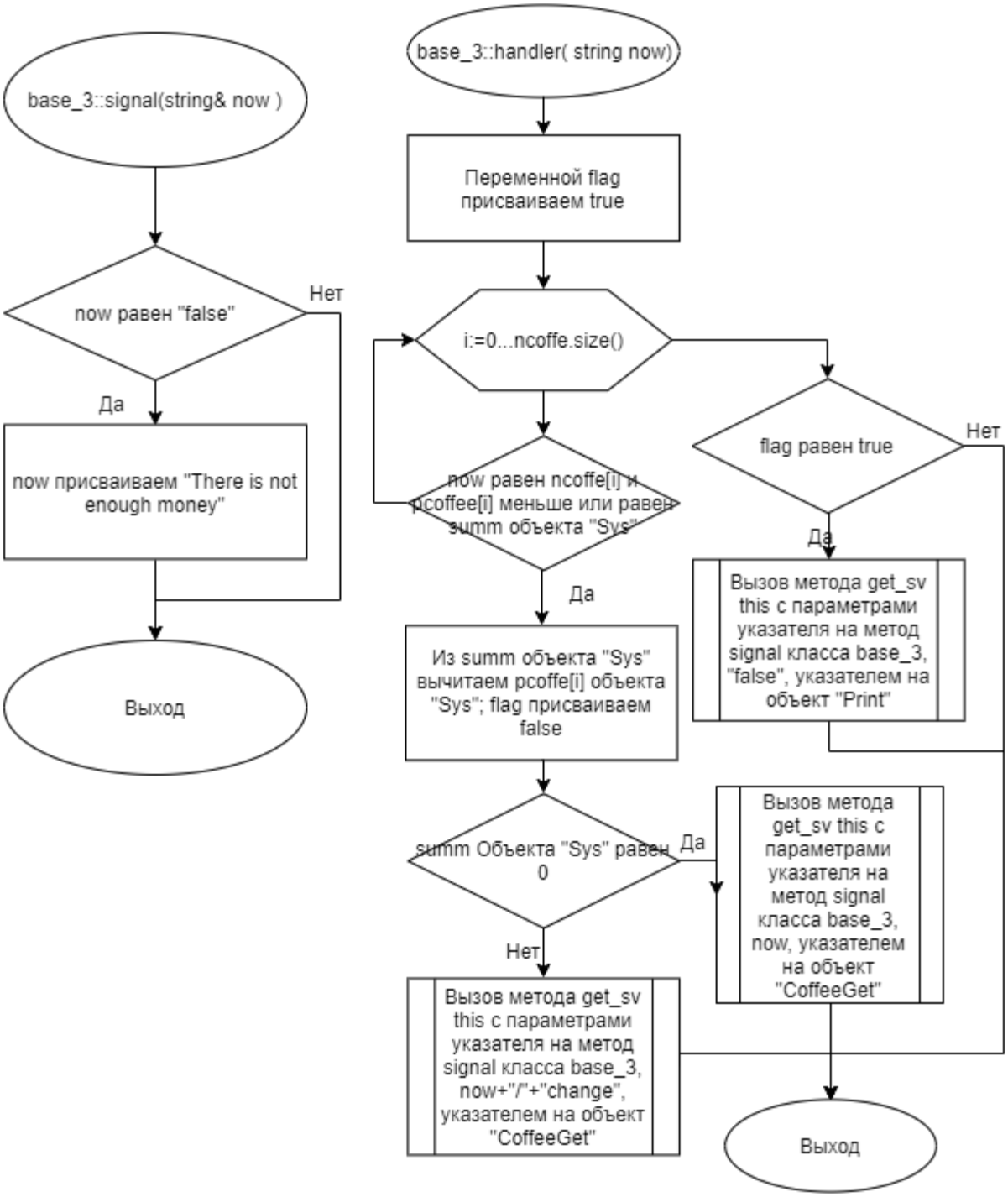

Рисунок 6 – Блок-схема алгоритма

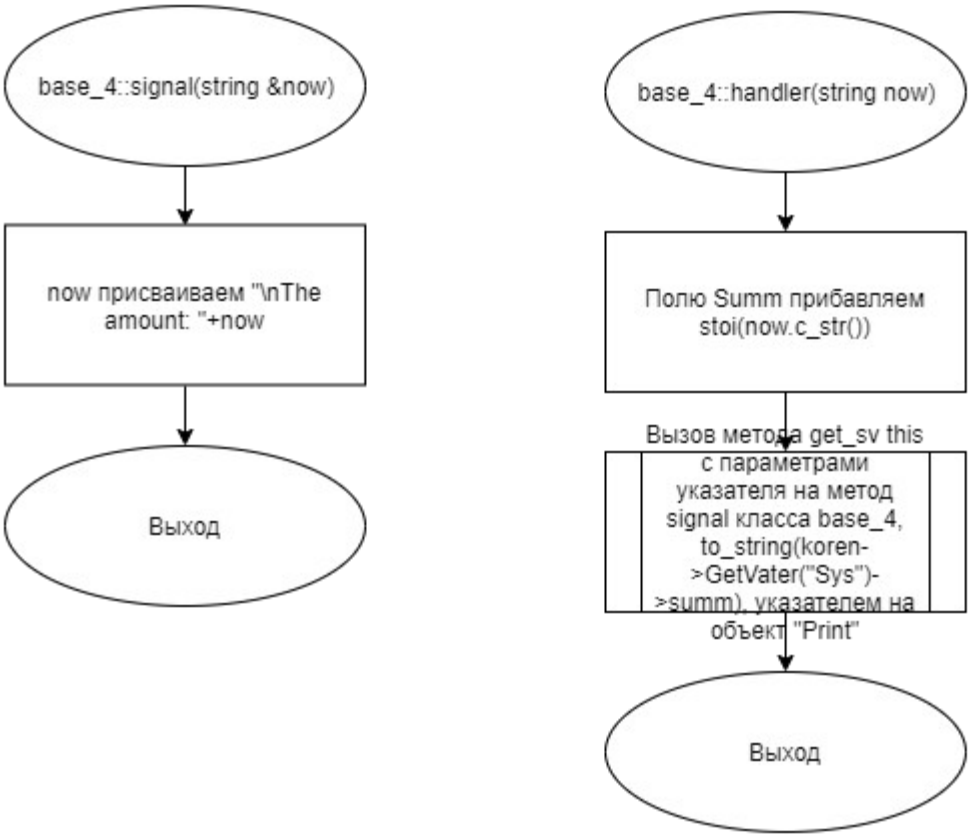

Рисунок 7 – Блок-схема алгоритма

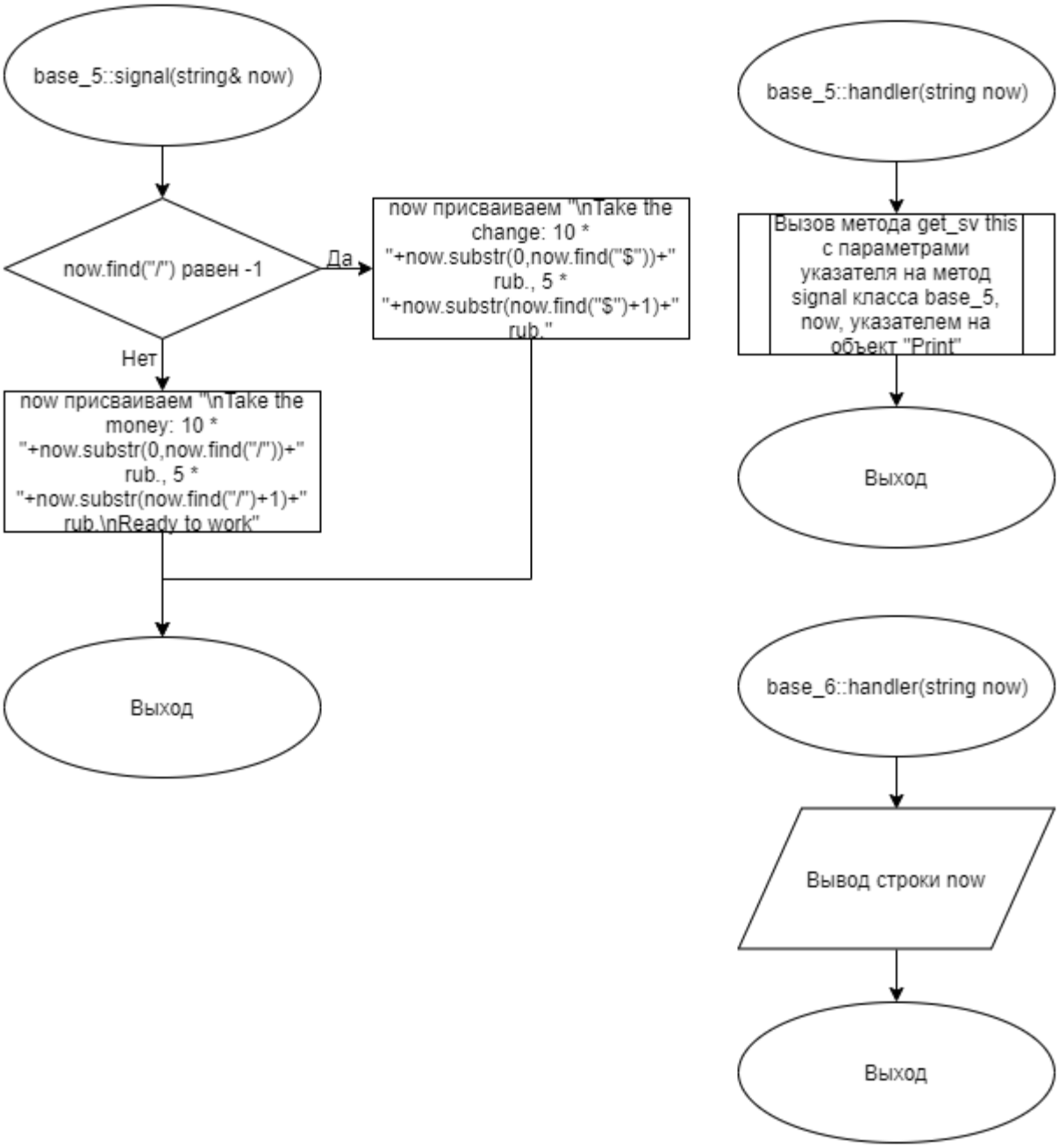

Рисунок 8 – Блок-схема алгоритма

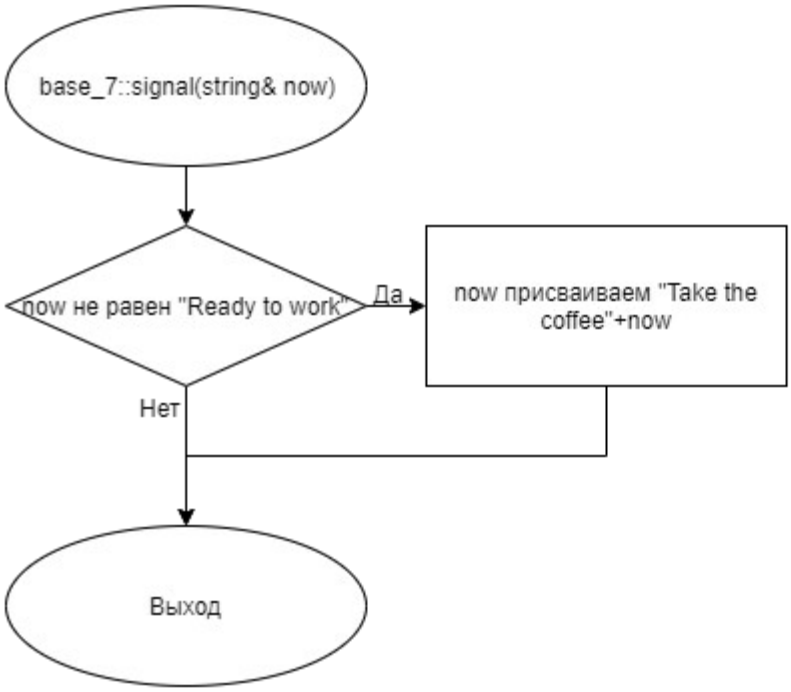

Рисунок 9 – Блок-схема алгоритма

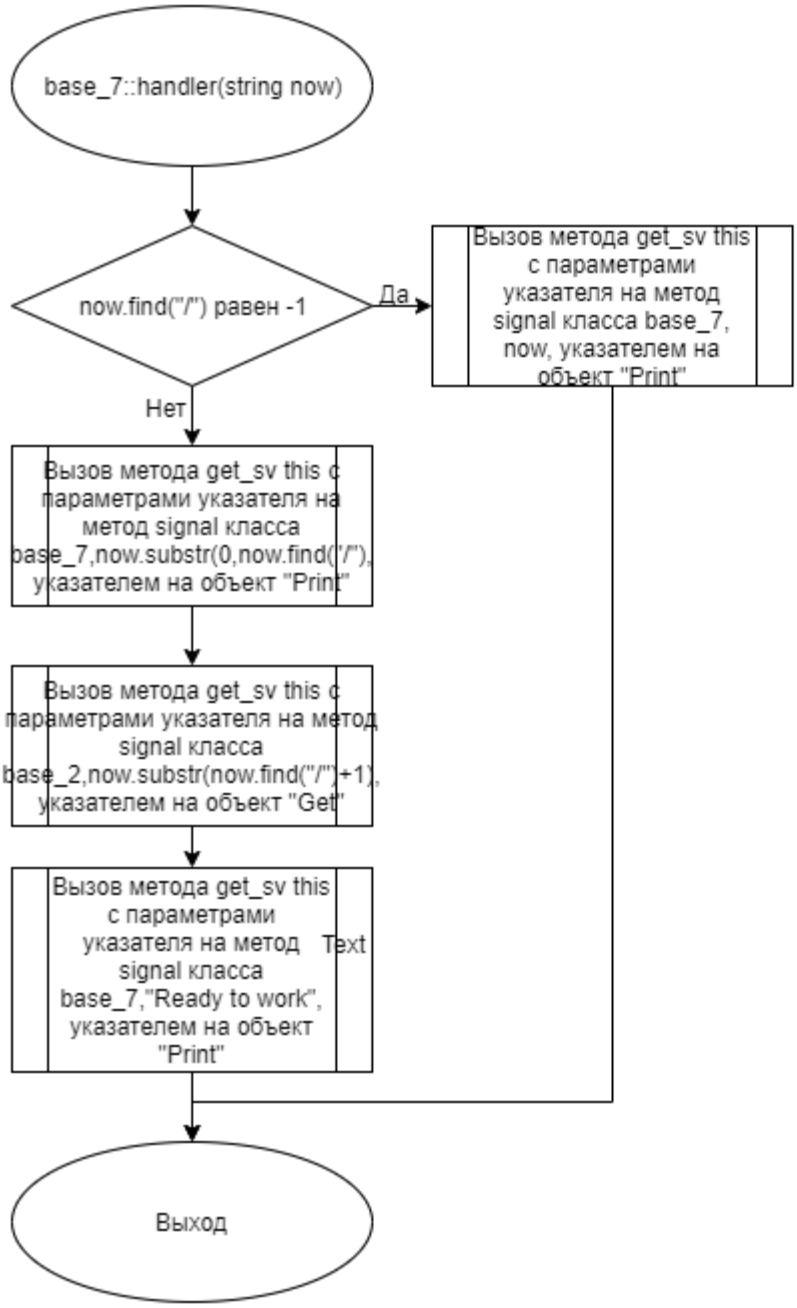

Рисунок 10 – Блок-схема алгоритма

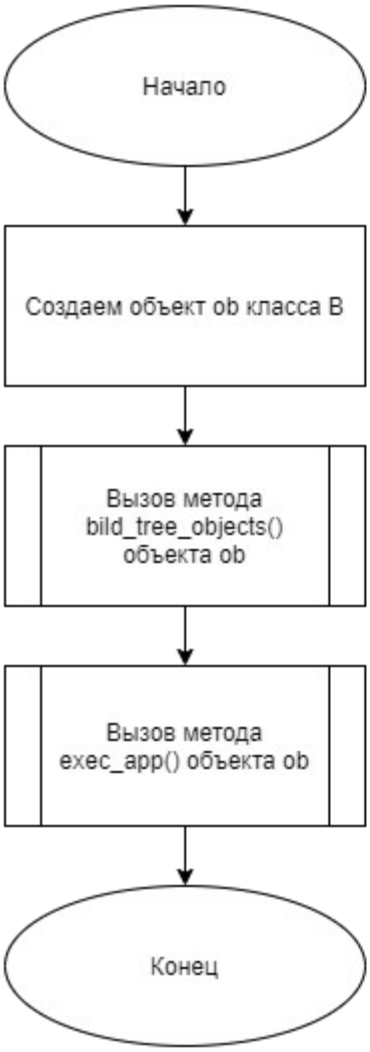

Рисунок 11 – Блок-схема алгоритма

5 КОД ПРОГРАММЫ

Программная реализация алгоритмов для решения задачи представлена ниже.

5.0 Файл Class.cpp

Листинг 1 – Class.cpp

```
#include "Class.h"
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;
base::base(string name, base* pt) {
	this->name = name;
	if (pt != nullptr) {
		this->pt = pt;
		pt->ukas.push_back(this);}
	else {
		this->pt = koren;
		koren->ukas.push_back(this);}
	number=1;
}
string base::GetName() {
	return name;
}
void base::SetName(string name) {
	this->name = name;
}
void base::print(int n) {
	string tab(n, ' ');
	cout <<endl<<tab<< GetName();
	for (int i = 0; i < ukas.size(); i++) {
		ukas[i]->print(n + 4);}}
void base::print_status(int n) {
	string tab(n, ' ');
	cout <<endl << tab << GetName();
	if (status == 0)
		cout << " is not ready";
	else
		cout << " is ready";
	for (int i = 0; i < ukas.size(); i++) {
		ukas[i]->print_status(n + 4);}
}
base::base() {
	pt = nullptr;
	//name="System is ready";
}
```

```
base* base::GetVater(string name) {
	for (int i = 0; i < ukas.size(); i++) {
		if (ukas[i]->GetName() == name) {
			return ukas[i];}}
	for (int i = 0; i < ukas.size(); i++) {
		if (ukas[i]->GetVater(name)->GetName() == name) {
			return ukas[i]->GetVater(name);}
}
	return this;
}


base* base::Get_hd() {
	return pt;
}

void base::Set_hd(base* ptr) {
	if (pt != nullptr) {
		auto help = pt->ukas;
		auto help1 = find(help.begin(), help.end(), this);
		help.erase(help1);
		pt->ukas = help;
}
	pt = ptr;
	if (pt != nullptr) {
		pt->ukas.push_back(this);}
}
base* base::koren = new base();

void base::Set_status(int p) {
	if(p==0){
		this->status = 0;
		for (int i = 0; i < ukas.size(); i++)
			ukas[i]->Set_status(0);
}
	else {
		if (pt != nullptr) {
			if (pt->status != 0)
				this->status = p;
		}else
			this->status = p;
	}
}
base* base::jetzt=koren;

base* base:: finde(string adres){
	if(adres=="/")
		return koren;
	else if (adres==".")
		return jetzt;
	else if (adres[0]+adres[1]=='//'){
		if(koren->GetVater(adres.substr(2))->GetName()==adres.substr(2))
			return koren->GetVater(adres.substr(2));
		else
			return nullptr;}
	else	if	(adres.find("/")==0	and	koren-
>GetVater(adres.substr(adres.rfind("/")+1))->Get_adres()==adres){
```

```
            if(koren->GetVater(adres.substr(adres.rfind("/")+1))-
>GetName()==adres.substr(adres.rfind("/")+1)and                                  koren-
>GetVater(adres.substr(adres.rfind("/")+1))->pt!=nullptr)
                  return koren->GetVater(adres.substr(adres.rfind("/")+1));
            else
                  return nullptr;
      }
      else if(adres.find("/")!=0 and adres.find("/")!=1 ){
            if(koren->GetVater(adres.substr(adres.rfind("/")+1))-
>GetName()==adres.substr(adres.rfind("/")+1))
                  return jetzt->GetVater(adres.substr(adres.rfind("/")+1));
            else
                  return nullptr;}
      else
            return nullptr;
}


bool base:: set(string adres){
      if (finde(adres)==nullptr)
            return false;
      else{
            jetzt=finde(adres);
            return true;}
}
int base::GetNumber(){
      return number;}

string base::Get_adres() {
      vector<string> mas;
      if(pt== nullptr)
            return"/";
      string adres="";
      mas.push_back(this->name);
      base* help;
      help=this->pt;
      while (help!= nullptr){
            mas.push_back(help->name);
            help=help->pt;}
      for(int i=mas.size()-2;i>=0;i--){
            adres+="/";
            adres+=mas[i];}
      return adres;}

void base::set_sv(TYPE_SIGNAL sig, base *p, TYPE_HANDEL hen) {
      for (int i = 0; i < connects.size(); i++) {
            if(connects[i]->sig==sig   and   connects[i]->p==p   and   connects[i]-
>hen==hen)
                  return;}
      connects.push_back(new sv{sig,p,hen});
}

void base::del_sv(TYPE_SIGNAL sig, base *p, TYPE_HANDEL hen) {
      for (int i = 0; i < connects.size(); i++) {
            if(connects[i]->sig==sig   and   connects[i]->p==p   and   connects[i]-
>hen==hen)
                  connects.erase(connects.begin()+i);}}
```

```
void base::get_sv(TYPE_SIGNAL sig, string stroka,base *p) {
      (this->*sig)(stroka);//Получение конкретного метода signal
      for (int i = 0; i < connects.size(); i++) {
            if((connects[i]->sig==sig )and (connects[i]->p==p)){
                  TYPE_HANDEL  hendl=connects[i]->hen;//  Конкретный  метод  отдачи
(класса)
                  if(connects[i]->p->status!=0)
                        (connects[i]->p->*hendl)(stroka);//Объект нужен для вывода
адреса
            }
      }
}









void base::Set_status_all(int a){
      Set_status(a);
      for (int i =0 ;i<ukas.size();i++){
            ukas[i]->Set_status_all(a);};
}

bool base::GetStatus() {
      if (this-> status!=0)
            return true;
      else
            return false;
}
```

5.1 Файл Class.h

Листинг 2 – Class.h

```
#ifndef CLASS_H
#define CLASS_H
#include <iostream>
#include <vector>
#include <string>
class base;
typedef void(base::*TYPE_SIGNAL)(std::string&);
typedef void(base::*TYPE_HANDEL)(std::string);
using namespace std;
class base {
protected:
      int status = 0;
      string name;
      base* pt;
      vector<base*>  ukas;
```

```
      static base* koren;
      static base* jetzt;
      struct sv{
            TYPE_SIGNAL sig;
            base* p;
            TYPE_HANDEL hen;};
      vector<sv*> connects;
      int number;
      int summ=0,fi,te;
      vector <string>ncoffe;
      vector <int>pcoffe;

public:
      void SetName(string name);
      string GetName();
      base(string name, base* pt);
      base();
      void print(int n);
      void print_status(int n);
      base* Get_hd();
      void Set_hd(base* ptr);
      base* GetVater(string name);
      void Set_status(int p);
      base* finde (string adres);
      bool set(string adres);
      string Get_adres();
      int GetNumber();
      void Set_status_all(int a);
      void set_sv(TYPE_SIGNAL sig,base* p,TYPE_HANDEL hen);
      void del_sv(TYPE_SIGNAL sig,base* p,TYPE_HANDEL hen);
      void get_sv(TYPE_SIGNAL sig, string stroka,base *p);
      bool GetStatus();
      friend class base_2;
      friend class base_3;
      friend class base_4;
};
#endif
```

5.2 Файл CoffeeGet.cpp

Листинг 3 – CoffeeGet.cpp

```
#include "CoffeeGet.h"
#include "System.h"
#include "Read.h"
#include <string>
using namespace std;
base_7::base_7(string name, base* pt) :base(name, pt) {number=7;
};

void base_7::signal(string& now){
      if(now!="\nReady to work")
            now="\nTake the coffee "+now;
```

```
}

void base_7::handler(string now){
      if(now.find("/")==-1)
            this->get_sv((TYPE_SIGNAL)(&base_7::signal),now,koren-
>GetVater("Print"));
      else{
            this->get_sv((TYPE_SIGNAL)
(&base_7::signal),now.substr(0,now.find("/")),koren->GetVater("Print"));
            this->get_sv((TYPE_SIGNAL)(&base_2::signal),now.substr(now.find("/")
+1),koren->GetVater("Get"));
      }
      this->get_sv((TYPE_SIGNAL)(&base_7::signal),"\nReady     to     work",koren-
>GetVater("Print"));
}
```

5.3 Файл CoffeeGet.h

Листинг 4 – CoffeeGet.h

```
#ifndef _COFFEEGET_H
#define _COFFEEGET_H

#include "Class.h"
#include <vector>
#include <string>
#include <iostream>
class base_7 : public base {
public:
      base_7(string name, base* pt);
      void signal (string&);
      void handler(string);
};

#endif
```

5.4 Файл Get.cpp

Листинг 5 – Get.cpp

```
#include "Get.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
void base_5::signal(string& now) {
      if(now.find("/")==-1)
            now="\nTake the change: 10 * "+now.substr(0,now.find("$"))+" rub.,  5
* "+now.substr(now.find("$")+1)+" rub.";
```

```
      else
            now="\nTake the money: 10 * "+now.substr(0,now.find("/"))+" rub.,  5 *
"+now.substr(now.find("/")+1)+" rub.\nReady to work";
}
void base_5::handler(string now) {
      this->get_sv((TYPE_SIGNAL)(&base_5::signal),now,koren->GetVater("Print"));
}
```

5.5 Файл Get.h

Листинг 6 – Get.h

```
#ifndef _GET_H
#define _GET_H
#include "Print.h"
#include <vector>
#include <string>
#include <iostream>
class base_5 :public base {
private:
public:
      base_5(string name, base* pt) :base(name, pt) {number=5;};
      void signal (string&);
      void handler(string);
};
#endif
```

5.6 Файл main.cpp

Листинг 7 – main.cpp

```
#include "System.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
int main() {
      B ob;
      ob.bild_tree_objects();
      ob.exec_app();
}
```

5.7 Файл Print.cpp

Листинг 8 – Print.cpp

```
#include "Print.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
void base_6::handler(string now) {
            cout<<now;
}
```

5.8 Файл Print.h

Листинг 9 – Print.h

```
#ifndef _PRINT_H
#define _PRINT_H
#include "Class.h"
#include <vector>
#include <string>
#include <iostream>
class base_6 :public base {
private:
public:
      base_6(string name, base* pt) :base(name, pt) {number=6;};
      void signal (string&);
      void handler(string);
};
#endif
```

5.9 Файл Pult.cpp

Листинг 10 – Pult.cpp

```
#include "Pult.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
void base_3::signal(string& now) {
      if(now=="false"){
            now="\nThere is not enough money";
      }
}
void base_3::handler(string now) {
```

```
      bool flag=true;
      for (int i=0;i<koren->GetVater("Sys")->ncoffe.size();i++){

            if     ((now==koren->GetVater("Sys")->ncoffe[i])     and     ((koren-
>GetVater("Sys")->pcoffe[i])<=(koren->GetVater("Sys")->summ))){
                  koren->GetVater("Sys")->summ-=koren->GetVater("Sys")->pcoffe[i];
                  flag=false;
                  if(koren->GetVater("Sys")->summ==0){
                        this->get_sv((TYPE_SIGNAL)(&base_3::signal),now,koren-
>GetVater("CoffeeGet"));}
                  else
                        this->get_sv((TYPE_SIGNAL)
(&base_3::signal),now+"/"+"change",koren->GetVater("CoffeeGet"));
            }
      }
      if (flag)
            this->get_sv((TYPE_SIGNAL)(&base_3::signal),"false",koren-
>GetVater("Print"));

}
```

5.10 Файл Pult.h

Листинг 11 – Pult.h

```
#ifndef PULT_H
#define PULT_H
#include <string>
#include "Class.h"
class base_3 :public base {
private:
public:
      base_3(string name, base* pt) :base(name, pt) {number=3;};
      void signal (string&);
      void handler(string);
};
#endif
```

5.11 Файл Read.cpp

Листинг 12 – Read.cpp

```
#include "Read.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
void base_2::signal(string& now){
      if (now=="money"){
```

```
            int fife,ten;
            ten=(koren->GetVater("Sys")->summ)/10;
            koren->GetVater("Sys")->summ-=ten*10;
            fife=koren->GetVater("Sys")->summ/5;
            koren->GetVater("Sys")->summ-=fife*5;
            koren->GetVater("Sys")->fi-=fife;
            koren->GetVater("Sys")->te-=ten;
            now=to_string(ten)+"/"+to_string(fife);
      }
      if(now=="change"){
            int fife,ten;
            ten=(koren->GetVater("Sys")->summ)/10;
            koren->GetVater("Sys")->summ-=ten*10;
            fife=koren->GetVater("Sys")->summ/5;
            koren->GetVater("Sys")->summ-=fife*5;
            koren->GetVater("Sys")->fi-=fife;
            koren->GetVater("Sys")->te-=ten;
            now=to_string(ten)+"$"+to_string(fife);
      }
      if(now=="false")
            now="\nTake the money back, no change";
}

void base_2::handler(string now){
      if (now=="Refund"){
            cin>>now;
            this->get_sv((TYPE_SIGNAL)(&base_2::signal),now,koren-
>GetVater("Get"));}
      else if (now=="Coffee"){
            cin>>now;
            this->get_sv((TYPE_SIGNAL)(&base_2::signal),now,koren-
>GetVater("Pult"));
      }
      else if (now=="Cancel"){
            this->get_sv((TYPE_SIGNAL)(&base_2::signal),"\nTurned      off",koren-
>GetVater("Print"));
      }
      else if (now=="SHOWTREE"){
            koren->GetVater("Sys")->print_status(0);
      }
      else{
            if (stoi(now.c_str())==50){
                  if      ((koren->GetVater("Sys")->fi*5)+(koren->GetVater("Sys")-
>te*10)<50)
                        this->get_sv((TYPE_SIGNAL)(&base_2::signal),"false",koren-
>GetVater("Print"));
                  else
                        this->get_sv((TYPE_SIGNAL)(&base_2::signal),now,koren-
>GetVater("Set"));
            }

            else if (stoi(now.c_str())==100){
                  if      ((koren->GetVater("Sys")->fi*5)+(koren->GetVater("Sys")-
>te*10)<100)
                        this->get_sv((TYPE_SIGNAL)(&base_2::signal),"false",koren-
>GetVater("Print"));
                  else
```

```
                        this->get_sv((TYPE_SIGNAL)(&base_2::signal),now,koren-
>GetVater("Set"));
            }
            else{
                  this->get_sv((TYPE_SIGNAL)(&base_2::signal),now,koren-
>GetVater("Set"));
                  if(now=="5")
                        koren->GetVater("Sys")->fi+=1;
                  if(now=="10")
                        koren->GetVater("Sys")->te+=1;
            }
      }
}

void base_2::signal_go(string& now){
      int n,p;
      n=atoi(now.c_str());
      string name;
      for (int i=0;i<n;i++){
            cin>>name;
            koren->GetVater("Sys")->ncoffe.push_back(name);
      }
      for (int i=0;i<n;i++){
            cin>>p;
            koren->GetVater("Sys")->pcoffe.push_back(p);
      }
      cin>>koren->GetVater("Sys")->fi;
      cin>>koren->GetVater("Sys")->te;
      now="Ready to work";
}
```

5.12 Файл Read.h

Листинг 13 – Read.h

```
#ifndef READ_H
#define READ_H
#include "Class.h"
#include <string>
using namespace std;
class base_2 :public base {
private:
public:
      base_2(string name, base*pt) :base(name, pt) {number=2;};
      void signal (string&);
      void handler(string);
      void signal_go(string&);
};
#endif
```

5.13 Файл Set.cpp

Листинг 14 – Set.cpp

```
#include "Set.h"
#include <iostream>
#include <string>
#include <vector>
using namespace std;
void base_4::signal(string& now) {
      now ="\nThe amount: "+now;
}
void base_4::handler(string now) {
      koren->GetVater("Sys")->summ+=stoi(now.c_str());
      get_sv((TYPE_SIGNAL)(&base_4::signal),to_string(koren->GetVater("Sys")-
>summ),koren->GetVater("Print"));
}
```

5.14 Файл Set.h

Листинг 15 – Set.h

```
#ifndef SET_H
#define SET_H
#include "Class.h"
class base_4 :public base {
private:
      int pl;
public:
      base_4(string name, base* pt) :base(name, pt) {number=4;};
      void signal (string&);
      void handler(string);
};
#endif
```

5.15 Файл System.cpp

Листинг 16 – System.cpp

```
#include "System.h"
#include "Get.h"
#include "Print.h"
#include "CoffeeGet.h"
#include "Read.h"
#include "Pult.h"
#include "Set.h"
#include <iostream>
```

```
#include <string>
#include <vector>
using namespace std;


void B::bild_tree_objects() {
      new base_2("Read",this);
      new base_3("Pult",this);
      new base_4("Set",this);
      new base_5("Get",this);
      new base_6("Print",this);
      new base_7("CoffeeGet",this);
      koren->GetVater("Sys")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Read"),(TYPE_HANDEL)(&base_2::handler));
      koren->GetVater("Read")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Set"),(TYPE_HANDEL)(&base_4::handler));
      koren->GetVater("Set")->set_sv((TYPE_SIGNAL)(&base_4::signal),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("Read")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("Sys")->set_sv((TYPE_SIGNAL)(&base_2::signal_go),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("Read")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Get"),(TYPE_HANDEL)(&base_5::handler));
      koren->GetVater("Get")->set_sv((TYPE_SIGNAL)(&base_5::signal),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("Read")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Pult"),(TYPE_HANDEL)(&base_3::handler));
      koren->GetVater("Pult")->set_sv((TYPE_SIGNAL)(&base_3::signal),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("Pult")->set_sv((TYPE_SIGNAL)(&base_3::signal),koren-
>GetVater("CoffeeGet"),(TYPE_HANDEL)(&base_7::handler));
      koren->GetVater("CoffeeGet")->set_sv((TYPE_SIGNAL)(&base_7::signal),koren-
>GetVater("Print"),(TYPE_HANDEL)(&base_6::handler));
      koren->GetVater("CoffeeGet")->set_sv((TYPE_SIGNAL)(&base_2::signal),koren-
>GetVater("Get"),(TYPE_HANDEL)(&base_5::handler));

}
void B::exec_app () {
      koren->Set_status_all(1);
      string cmd;
      getline(cin,cmd,' ');
      koren->GetVater("Sys")->get_sv((TYPE_SIGNAL)(&base_2::signal_go),cmd,koren-
>GetVater("Print"));
      do{
            cin>>cmd;
            koren->GetVater("Sys")->get_sv((TYPE_SIGNAL)
(&base_2::signal),cmd,koren->GetVater("Read"));
      }while(cmd!="Cancel" and cmd!="SHOWTREE");
}

B::B():base("Sys",koren){number=1;};



void B::signal(string& stroka) {
}
```

```
void B::handler(string stroka) {
};
B::B(string name, base *pt):base(name,pt){number=1;}
```

5.16 Файл System.h

Листинг 17 – System.h

```
#ifndef _SYSTEM_H
#define _SYSTEM_H
#include "Class.h"
#include <vector>
#include <string>
#include <iostream>
class B : public base {
private:
public:
      B();
      B(string name, base *pt);
      void bild_tree_objects();
      void exec_app();
      void signal(string& now);
      void handler(string now);

};


#endif
```

6 ТЕСТИРОВАНИЕ

Результат тестирования программы представлен в таблице 17.

Таблица 17 – Результат тестирования программы

Входные данные	Ожидаемые выходные данные	Фактические выходные данные
3 Espresso Americano Cappuchino 25 50 50 3 5 10 Coffee Cappuchino 50 Coffee Espresso SHOWTREE	Ready to work The amount: 10 There is not enough money The amount: 60 Take the coffee Espresso Take the change: 10 * 3 rub., 5 * 1 rub. Ready to work Sys is ready Read is ready Pult is ready Set is ready Get is ready Print is ready CoffeeGet is ready	Ready to work The amount: 10 There is not enough money The amount: 60 Take the coffee Espresso Take the change: 10 * 3 rub., 5 * 1 rub. Ready to work Sys is ready Read is ready Pult is ready Set is ready Get is ready Print is ready CoffeeGet is ready
3 Espresso Americano Cappuchino 25 50 50 3 5 50 Coffee Cappuchino 10 10 10 Coffee Espresso 5 5 Refund money 5 100 Cancel	Ready to work The amount: 50 Take the coffee Cappuchino Ready to work The amount: 10 The amount: 20 The amount: 30 Take the coffee Espresso Take the change: 10 * 0 rub., 5 * 1 rub. Ready to work The amount: 5 The amount: 10 Take the money: 10 * 1 rub., 5 * 0 rub. Ready to work The amount: 5 Take the money back, no change Turned off	Ready to work The amount: 50 Take the coffee Cappuchino Ready to work The amount: 10 The amount: 20 The amount: 30 Take the coffee Espresso Take the change: 10 * 0 rub., 5 * 1 rub. Ready to work The amount: 5 The amount: 10 Take the money: 10 * 1 rub., 5 * 0 rub. Ready to work The amount: 5 Take the money back, no change Turned off
3 Espresso Americano Cappuchino 25 50 50 3 5 100 50 Refund money 10 10 5 Coffee Espresso	Ready to work Take the money back, no change The amount: 50 Take the money: 10 * 5 rub., 5 * 0 rub. Ready to work The amount: 10 The amount: 20 The amount: 25 Take the coffee Espresso	Ready to work Take the money back, no change The amount: 50 Take the money: 10 * 5 rub., 5 * 0 rub. Ready to work The amount: 10 The amount: 20 The amount: 25 Take the coffee Espresso

Входные данные	Ожидаемые выходные данные	Фактические выходные данные

SHOWTREE	Ready to work
	Sys is ready
	    Read is ready
	    Pult is ready
	    Set is ready
	    Get is ready
	    Print is ready
	    CoffeeGet is ready
		Ready to work
		Sys is ready
		    Read is ready
		    Pult is ready
		    Set is ready
		    Get is ready
		    Print is ready
		    CoffeeGet is ready

3 Espresso Americano Cappuchino 25 50 50 3 5 50 Coffee Espresso Cancel	Ready to work The amount: 50 Take the coffee Espresso Take the change: 10 * 2 rub., 5 * 1 rub. Ready to work Turned off	Ready to work The amount: 50 Take the coffee Espresso Take the change: 10 * 2 rub., 5 * 1 rub. Ready to work Turned off

1 Espresso
25
1 1
SHOWTREE
	Ready to work
	Sys is ready
	    Read is ready
	    Pult is ready
	    Set is ready
	    Get is ready
	    Print is ready
	    CoffeeGet is ready
		Ready to work
		Sys is ready
		    Read is ready
		    Pult is ready
		    Set is ready
		    Get is ready
		    Print is ready
		    CoffeeGet is ready

ЗАКЛЮЧЕНИЕ

При выполнении работы были достигнуты все поставленные задачи:

Освоение принципов объектно - ориентированного программирования

Освоение основ объектно- ориентированного языка программирования C++

Освоение разработки программы как система

Освоение умения проектирования архитектуры программы на базе построения иерархии обектов

Освоение выполнения всех необходимых работ согласно этапам разработки программы и соответствующих программных инструментов

Моделирование работы кофемашины при помощи сигнаов и обработчиков

Построение системы взаимодействия объектов с помощью интерфейса сигналов и обработчиков

Опсиание алгоритма работы программы

Построение кода на языке программирования C++, согласно разработанному алгоритму работы программы

Тестирование работы программы

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).