МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

СОГЛАСОВАНО
Заведующий кафедрой
«Информационные технологии и компьютерные системы»

_____ А.А. Брюховецкий
«_20_» _____04_____ 2022_ г.

УТВЕРЖДАЮ
Директор Института
Информационных технологий и управления в технических системах

_____ В.Н. Бондарев
«_20_» _____04_____ 2022_ г.

М.П.

РАБОЧАЯ ПРОГРАММА ПРАКТИКИ

УЧЕБНАЯ

(вид практики)

ТЕХНОЛОГИЧЕСКАЯ (ПРОЕКТНО-ТЕХНОЛОГИЧЕСКАЯ) ПРАКТИКА

(тип практики)

09.03.04 «Программная инженерия»

(код и направление подготовки / специальность)

Системное и прикладное программное обеспечение

(наименование профиля подготовки / специализации)

бакалавриат

(уровень высшего образования)

очная, 2022 г.

(форма обучения, год обучения)

Севастополь
2022 г.

Рабочая программа технологической (проектно-технологической) практики составлена на основе ФГОС 09.03.04 «Программная инженерия», утвержденного приказом Министерства образования и науки Российской Федерации от 19.09.2017 №920;

Рабочая программа технологической (проектно-технологической) практики рассмотрена и одобрена на заседании кафедры «Информационные технологии и компьютерные системы» «20» 04 2022 г., протокол № 6.

Руководитель образовательной программы _____ _____ А.А. Брюховецкий
(подпись) *(И.О. Фамилия)*

Рабочая программа практики
составлена:

доцент, к.т.н. _____ Т.В. Волкова
(подпись) *(И.О. Фамилия)*

Рецензент
Генеральный директор ООО
«ЛАНКОМ»

П.С. Никифоров
(И.О. Фамилия)

Содержание

1. Цели практики

Цель технологической (проектно-технологической) практики – расширить теоретические и практические знания, полученные в процессе обучения по дисциплинам «Языки обработки знаний», «Объектно-ориентированное программирование», получить навыки командной работы при разработке программного обеспечения, получить начальные навыки в разработке бизнес-планов и технических заданий на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием, освоить возможности пакета Microsoft Office по разработке презентаций, оформлению научно-технических отчетов по результатам выполненной работы.

2. Задачи практики

Основными задачами технологической (проектно-технологической) практики являются:

- углубление и закрепление полученных теоретических знаний;
- ознакомление на практике с методологиями управления проектами разработки программного обеспечения, такими как Scrum, Kanban, WaterFall;
- получение навыков управления компьютером с помощью работы в командной строке;
- практическое применение интерактивной среды разработки IDE для улучшения процесса написания программного кода;
- изучение возможностей систем сборки проектов, на примере Gradle;
- получение навыков в использовании системы контроля версии (типа Git) для хранения, отслеживания и внесения изменений в программный проект;
- получение навыков совместной работы над проектом через Github/Gitlab/Bitbucket/Azure (по выбору команды студентов);
- изучение возможностей Pull Requests / Merge Requests, как способов командной работы с 1 кодом;
- получение навыков тестирования программ с помощью unit-тестов (на примере Unit-5 для Java);
- автоматический запуск тестов в CI/CD в выбранной платформе совместной работы;
- получение практических навыков разработки программного проекта;
- получение навыков взаимодействия в командной работе над учебным программным проектом;
- получение навыков в подготовке презентаций и технических отчетов средствами Microsoft Office.

3. Место практики в структуре ООП

Технологическая (проектно-технологическая) практика базируется на учебных следующих дисциплинах:

- Языки обработки знаний
- Объектно-ориентированное программирование.

Изучение этих дисциплин позволяет студентам, в результате успешного усвоения теоретических курсов, иметь знания, умения и компетенции для освоения программы учебной практики.

4. Тип практики, способ и форма ее проведения

Технологическая (проектно-технологическая) практика для студентов 2 курса проводится в компьютерных классах ФГАОУ ВО «Севастопольский государственный университет» в течение двух недель в соответствии с учебным планом подготовки бакалавров направления 09.03.04 «Программная инженерия». Ряд студентов может быть направлен для прохождения практики на IT-предприятия по предварительной договоренности при наличии договора о прохождении практики студентами компьютерных специальностей СевГУ на данном предприятии.

5. Планируемые результаты обучения при прохождении практики, соотнесенные с планируемыми результатами освоения образовательной программы

Код и уровень формируемой компетенции по ООП	Владения	Умения	Знания
ОПК-2. Способен использовать современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности;	Владеть навыками использования современных методологий управления распределенной работой над программным проектом, навыками алгоритмизации вычислительных задач, навыками разработки и запуска программ с помощью	Уметь организовывать процесс командной работы над программным проектом; формулировать требования по оснащению офиса необходимым для разработки проекта аппаратным и программным обеспечением; разрабатывать программы средней сложности на языке Java.	Знать современные методологии управления разработкой программного проекта, виды программного обеспечения, структуру систем разработки программ, особенности трансляции и выполнения Java-программ, состав комплекта разработчика Java Development Kit (JDK), основы объектно-ориентированного программирования на

	инструментальной системы BlueJ (или другой системы разработки программ на языке Java).		языке Java, структуру библиотеки Java, назначение основных библиотечных классов и пакетов, принципы организации и использования систем контроля версий ПО.
ОПК-5. Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем	Владеть: Навыками инсталляции систем разработки программ (BlueJ, IntelliJ IDEA).	Уметь: Создавать репозитории программного кода на GitLab.	Знать: Принципы работы и инсталляции систем разработки программ, системы контроля версий (Git), систем управления репозиториями программного кода для Git (GitLab).

6. Структура и содержание практики

6.1 Структура практики

Общая трудоемкость практики составляет 3 зачетных единицы (108 часов), 2 недели.

№раздела	Наименование раздела практики	Виды учебной нагрузки и их трудоемкость, часы			
		Теоретические занятия зка	Практические занятия	СРС	Всего часов
1	Подготовительный этап.	6	6	10	22
2	Разработка и отладка программ в рамках распределенной работы над программным проектом.		46	15	61
4	Заключительный семинар		4	5	9
3	Подготовка отчета			16	16
ВСЕГО		8	56	46	108

6.2 Содержание практики

Подготовительный этап включает: организационное собрание, заполнение документов на практику, изучение теоретического материала по основам командной работы над программным проектом, разбиение студентов на команды и распределение ролей в команде, получение индивидуального задания.

Этап разработки и отладки программ в рамках распределенной работы над программным проектом: выполнение индивидуального задания и мероприятия для осуществления контроля за ходом разработки.

Заключительный семинар посвящен обмену опытом командной работы над проектом с использованием различных методологий.

По результатам практики подготавливается отчет.

7. Формы отчетности по практике

По итогам практики студент представляет отчет.

Защита отчета по ознакомительной практике производится на комиссии кафедры, как правило, в последние два дня практики (не позднее установленного срока). Комиссия, после сообщения студента о результатах практики, вопросов и обсуждения, объявляет оценку.

Отчет по практике (20 - 25 стр.) включает в себя следующие разделы:

Содержание.
Введение.
1. Постановка задачи разработки программной системы.
2. Анализ структуры программной системы.
3. Организация процесса создания проекта.
4. Описание разработанного программного обеспечения.
   4.1. Структура системы (подсистемы).
   4.2. Описание классов, входящих в подсистему.
6. Отладка и тестирование программы.
Заключение.
Библиографический список.
Приложения.

Во введении кратко описываются современные методологии командной работы над программным продуктом.

В разделе «Постановка задачи» приводится постановка задачи, указанная в задании на практику, описывается вариант задания, используемые средства разработки и выполнения программ, вычислительная платформа (требования к аппаратной и программной совместимости).

В разделе «Анализ структуры программной системы» анализируются входные и выходные данные, функции и управляющие воздействия системы, производится процесс декомпозиции системы на подсистемы, исходя из ее функций.

В разделе «Организация процесса создания проекта» обосновывается выбор метода управления разработкой, описывается распределение ролей в команде, распределение

заданий между программистами, правила и инструкции в рамках данного метода управления, организация контроля версий разрабатываемого программного обеспечения.

В разделе «Описание разработанного программного обеспечения» детализируется структура подсистемы, выделенной для разработки конкретному члену команды программистов (пункт «Структура подсистемы»), описываются разработанные классы (переменные экземпляра и методы).

В разделе «Отладка и тестирование программы» описываются средства отладки программы, которые применялись разработчиком (отладчик BlueJ, панель объектов BlueJ, «отладочная печать»), приводится описание наиболее ярких синтаксических ошибок, ошибок процесса выполнения и логических ошибок, возникших в процессе отладки программы.

После этого приводятся результаты проверки выполнения всех функций программы и делается вывод о корректности выполнения этих функций.

Заключение констатирует решение всех задач, описанных в разделе «Постановка задачи», содержит обобщение практических результатов, изложенных в отчете.

Список использованной литературы (отражает источники, которые использовались при выполнении индивидуального задания и написании отчета по практике).

В приложения выносятся промежуточные результаты разработки проекта (таблицы, диаграммы, сканы «доски» и другие материалы, используемые в процессе реализации выбранного метода управления разработкой), текст программы.

Текст отчета по учебной практике набирается в MicrosoftWord в формате А-4: шрифт TimesNewRoman – обычный, размер 14 пт; междустрочный интервал – полуторный; левое, верхнее и нижнее поля – 2,0 см; правое поле – 1,0 см; левое поле – 3см, абзац – 1,25 см. Объем отчета должен быть 20 - 25 страниц.

Страницы отчета нумеруют арабскими цифрами, с соблюдением сквозной нумерации по всему тексту. Номер проставляется в центре нижней части листа (выравнивание от центра) без точки в конце номера. Цифровой материал должен оформляться в виде таблиц.

Студенты при прохождении практики обязаны в конце дня записывать выполненную работу в дневнике-графике.

8. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по практике

8.1. Порядок оценивания результатов прохождения практик обучающимися

Оценка по практике выставляется на основе защиты отчета о практике и представляет дифференцированный зачет (с оценкой). При оценке результатов практики учитывается степень самостоятельности студентов и трудовая дисциплина. В таблице отражена балльно – рейтинговая система оценивания.

Таблица соответствия результатов контроля знаний по разным шкалам и критерии оценивания

Сумма баллов по 100-балльной шкале	Оценка ECTS	Критерии оценивания	Уровень компетентности	Оценка по национальной шкале	
				для экзамена, КП (КР), практики	для зачета
90 – 100	**A**	Индивидуальное задание выполнено полностью. При выполнении задания студент работал творчески, инициативно, самостоятельно. Отчет оформлен грамотно. Доклад по результатам практики сделан четко и ясно.	Высокий (творческий)	отлично	зачтено
82-89	**B**	Индивидуальное задание выполнено полностью. При выполнении задания студент работал творчески, инициативно, самостоятельно. Есть недочеты в оформлении отчета и/или в докладе.	Достаточный	хорошо	
75-81	**C**	Индивидуальное задание выполнено полностью. Руководитель практики влиял на результаты выполнения задания. Есть недочеты в оформлении отчета и/или в докладе.			
69-74	**D**	Задание выполнено частично (70%). Есть недочеты в оформлении отчета и/или в докладе.	Средний	Удовлетворительно	
60-68	**E**	Задание выполнено частично (50%). Есть недочеты в оформлении отчета и/или в докладе.			
35-59	**FX**	Задание не выполнено, отчет не оформлен.	Низкий	не удовлетворительно	не зачтено
1-34	**F**	Задание не выполнено, отчет не оформлен. Студент не являлся на место прохождения практики.			

8.2. Типовые контрольные задания или иные материалы, необходимые для оценки результатов обучения, характеризующих этапы формирования компетенций

8.2.1. Тип 1 🖹 программные проекты повышенной сложности

Задание по практике предполагает разработку программной системы, моделирующей работу простого компьютера. Программная модель состоит из трех частей, которые должны выполняться тремя студентами в команде. Первая часть состоит в разработке программы-ассемблера, которая будет переводить программу в мнемокодах в машинные коды. Вторая часть – создание программы-симулятора компьютера, которая должна выполнять машинную программу, состоящую из команд, входящих в систему команд данного компьютера. Третья часть – разработка графического интерфейса, позволяющего управлять программами, созданными в первой и второй частях, современным способом. Варианты заданий на практику отличаются набором команд (системой команд компьютера), структурой, размером и разрядностью оперативной памяти, структурой процессора (приложение А).

8.2.2. Тип 2 – программные проекты средней сложности (мини-проекты)

Перечисленные ниже мини-проекты могут выполняться как командой, состоящей из нескольких студентов, так и одним студентом (индивидуально).

Калькулятор. Реализовать простейший калькулятор, который дает возможность пользователю складывать, умножать, вычитать, делить числа, находить их квадратный корень.

Генератор паролей. Создать генератор случайных паролей для своих друзей и семьи, чтобы обезопасить их учётные записи!

Игра-викторина. Приложения-викторины задают пользователям серию вопросов и дают им возможность ответить. Затем викторина даёт пользователю результаты. Реализовать программу-викторину на любую интересную тему.

Игра "Угадай число". В данном мини-проекте необходимо реализовать игру, в которой компьютер пытается угадать целое число от 1 до 1000. Пользователь загадывает число, а программа задает ему вопросы из разряда *Да* или *Нет* в попытках угадать число. Если программа угадает число, она спрашивает у пользователя, то ли число она угадала. Если да - то «урааа!». Если нет, программа показывает грустный смайлик и предлагает новую игру.

Программа-шифровщик. В данном мини-проекте необходимо реализовать программу, которая позволяет как шифровать сообщение, так и дешифровать сообщение. Предполагается, что обладая знанием ключа/ключей, студент может зашифровать сообщение, а другой студент - расшифровать, и наоборот.

Описание вышеперечисленных мини-проектов детализировано в Приложении Б.

Интеллектуальная матрица. Создать программу, реализующую объект «Числовая матрица», способный выполнять заданные математические операции над инкапсулированной матрицей.

Варианты заданий для мини-проекта «Интеллектуальная матрица» приведены в Приложении В.

9. Учебно-методическое и информационное обеспечение практики

9.1 Основная литература

1	Пруцков, А.В. Программирование на языке Java. Введение в курс с примерами и практическими заданиями : учебник / А.В. Пруцков. — М. : КУРС, 2018.
2	Гуськова, О.И. Объектно ориентированное программирование в Java : учебное пособие / О. И. Гуськова. - Москва : МПГУ, 2018. - 240 с. - ISBN 978-5-4263-0648-6. - Режим доступа: http://znanium.com/catalog/product/1020593
3	Васюткина И.А. Технология разработки объектно-ориентированных программ на JAVA / Васюткина И.А. - Новосиб.:НГТУ, 2012. - 152 с.: ISBN 978-5-7782-1973-1 - Режим доступа: http://znanium.com/catalog/product/557111
4	Java™ 2 Platform Standard Edition 5.0 API Specification [Электронный ресурс] — Режим доступа: https://docs.oracle.com/javase/1.5.0/docs/api/index.html

9.2 Дополнительная литература

1	Общие требования к текстовым документам. ГОСТ 2.105—95. ЕСКД. М.: «Стандартинформ», 2005. — 28 с.
2	ГОСТ 18.701−90 (ИСО 5807-85). Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. – Введ. 1992-01-01. – М: Государственный стандарт Союза ССР.
3	Microsoft Office 2016. Новейший самоучитель. – Электронный ресурс. – Режим доступа: http://www.kavserver.ru/library/office2016manual.shtml.

9.3 Методические указания по практике

Методические указания по прохождению технологической (проектно-технологической) практики для студентов 2 курса очной формы обучения по направлениям 09.03.01 «Информатика и вычислительная техника» и 09.03.04 «Программная инженерия» доступны по ссылке:
https://drive.google.com/drive/folders/17CTLdzo7uesx1xrr1yCN5R6nKG8yAwF0?usp=sharin.

9.4 Информационные технологии

Перечень информационных технологий, используемых при проведении практики:
1. BlueJ, Eclipse или IntelliJ IDEA (Community) - бесплатные, платформо-независимые системы разработки Java-программ, включающие менеджер проектов, текстовый редактор, отладчик и др.;

2. Java Development Kit (JDK) — комплект для разработчика приложений на языке Java, который состоит из: стандартных библиотек Java, документации, компилятора Java, различных инструментов разработчика, примеров, а также исполнительной среды Java (интерпретатора).
3. Интернет-браузер для выхода на страницу описания библиотечных классов Java https://docs.oracle.com/javase/1.5.0/docs/api/index.html.
4. Пакет Microsoft Office 2010/2016.

10. Материально-техническое обеспечение практики

Практика проводится в компьютерных классах университета. Компьютеры должны быть оснащены программным обеспечением, приведенным в пункте 9 настоящей программы, и иметь доступ в интернет.

11. Бланки документов на практику

Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ДНЕВНИК ПРАКТИКИ

_____
(вид и тип практики)

обучающегося _____
(фамилия, имя, отчество)

Институт _____

Кафедра _____

Уровень образования _____

Направление подготовки
(специальность) _____

Профиль (специализация) _____

_____ курс, группа _____

Обучающийся _____
(фамилия, имя, отчество)

Прибыл на предприятие (в организацию, учреждение)

М.П. “____” _____ 20___ г.

_____ _____
(подпись) (должность, фамилия и инициалы ответственного лица)

Отметки о проведении инструктажа

Наименование инструктажа	Дата проведения инструктажа	ФИО, должность инструктирующего	Подпись	
			Инструкти-рующего	Инструкти-руемого
Инструктаж по технике безопасности				
Инструктаж по охране труда				
Инструктаж по пожарной безопасности				
Инструктаж по правилам внутреннего распорядка				

Убыл из предприятия (организации, учреждения)

М.П. “____” _____ 20___ г.

_____ _____
(подпись) (должность, фамилия и инициалы ответственного лица)

Индивидуальное задание

№ п/п	Вопросы, подлежащие изучению	Источник информации
	(перечень вопросов заполняет руководитель практики от Университета)	

Руководитель практики от Университета

_____ _____

(подпись) (инициалы и фамилия)

Руководитель практики от профильной организации

_____ _____

(подпись) (инициалы и фамилия)

«_____» _____ 20 ___ г.

Рабочий график (план) прохождения практики

№ п/п	Дата	Наименование работ	Отметка о выполнении
		(заполняет руководитель практики от Университета)	

Руководитель практики от Университета

_____ _____

(подпись) (инициалы и фамилия)

Руководитель практики от профильной организации

_____ _____

(подпись) (инициалы и фамилия)

«_____» _____ 20 ___ г.

Рабочие записи во время практики

Отзыв (характеристика)

_____

(Фамилия, имя, отчество обучающегося)

Руководитель практики от профильной организации

_____ _____

(подпись) (инициалы и фамилия)

«_____» _____ 20 ___ г.

Сведения об уровне освоения обучающимся компетенций

№	Формируемые компетенции	Руководитель от профильной организации	Руководитель от Университета
1.	ОК- (*Наименование компетенций заполняет руководитель практики от института, в соответствии с учебным планом)*	освоена / освоена частично/ не освоена	освоена / освоена частично/ не освоена
2.	ОК-	освоена / освоена частично/ не освоена	освоена / освоена частично/ не освоена
3.	ОПК -	освоена / освоена частично/ не освоена	освоена / освоена частично/ не освоена
4.	ПК -	освоена / освоена частично/ не освоена	освоена / освоена частично/ не освоена
5.	ПК -	освоена / освоена частично/ не освоена	освоена / освоена частично/ не освоена
6.	…		

Руководитель практики от Университета

_____ _____

(подпись) (инициалы и фамилия)

Руководитель практики от профильной организации

_____ _____

(подпись) (инициалы и фамилия)

«_____» _____ 20 ___ г.

Оценка результатов прохождения практики обучающимся

(заполняется руководителем практики от Университета)

_____
(Фамилия, имя, отчество обучающегося)

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Дата сдачи отчета «_____»_____ 20___ г.

Оценка:_____

Руководитель практики от Университета

_____ _____
(подпись) (инициалы и фамилия)

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий и управления в технических системах

Кафедра информационных технологий и компьютерных систем

№	Дата поступления на кафедру	Подпись отв. за регистрацию	Подпись преподавателя

ОТЧЕТ

о __технологической_ (_проектно-технологической_) практике
(вид практики) (тип практики)
в __ФГАОУ ВО «Севастопольский государственный университет»___
(наименование организации)

Выполнил _____
(Фамилия И.О. обучающегося)
_____ПИН/б -19-___- о_____
(шифр группы)
Направление / специальность 09.03.04 «Программная инженерия»_____
(код, наименование)

Руководитель практики от Университета
_____доцент_____
(должность)
_____Волкова Т.В._____
(Фамилия И.О. руководителя)

Севастополь
2022 г.

ПРИЛОЖЕНИЕ А

Варианты задания, описанного в п. 8.2.1 − Описание структуры процессора, системы команд и способов кодирования

Структура 1 (рисунок А.1). Описание и система команд (таблица А.1)

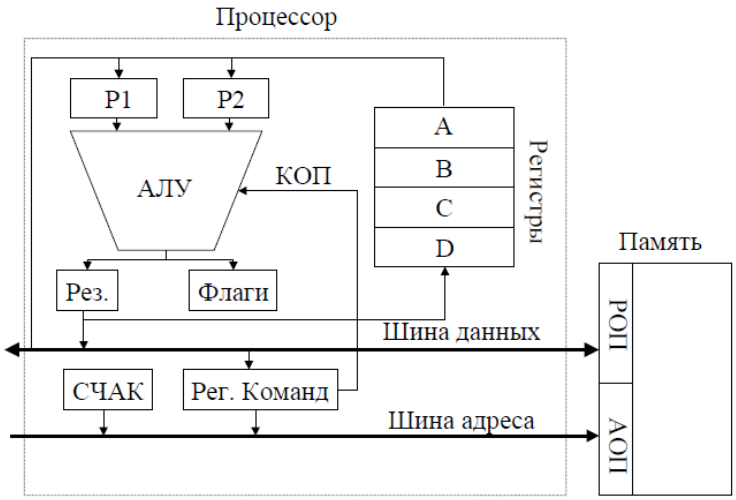

Рисунок А.1 – Структура 1

Объем оперативной памяти − 256 байт.

Длина команд − один или два байта. В первом байте биты 0..3 являются кодом операции, а следующие интерпретируются в зависимости от принадлежности команд к той или иной группе:

- для арифметических и логических команд содержат номера РОН (0-A, 1-B, 2-C, 3-D), над которыми производятся операции;
- для команд пересылки информации − номер регистра, с которым осуществляется операция;
- не используются для команд передачи управления и управления процессом.

Второй байт используется для задания адреса памяти, где находится операнд, адреса перехода для команд передачи управления или константы.

Таблица А.1 – Система команд для структуры 1

Код	Мнемокод и операнды	Описание
0000	ADD регистр1,регистр2	регистр1=регистр1+регистр2
0001	ADD регистр, память	регистр=регистр+память
0010	AND регистр1,регистр2	регистр1=регистр1&регистр2
0011	OR регистр1,регистр2	регистр1=регистр1 | регистр2
0100	NOT регистр	инверсия битов регистра
0101	MOV регистр,константа	занести константу в регистр
0110	LOAD регистр, адрес	загрузка данных в регистр из памяти
0111	STORE регистр, адрес	запись данных из регистра в память
1000	JMP адрес	безусловный переход
1001	JZ адрес	переход, если 0
1010	JO адрес	переход, если переполнение
1011	JL адрес	переход, если меньше
1100	JNZC регистр, адрес	переход, если не ноль, декремент регистра
1101	CLF	установка регистра флагов в 0
1110	CALL адрес	переход к подпрограмме, адрес возврата в регистре D
1111	RET	возврат из подпрограммы

Структура 2 (рисунок А.2). Описание и система команд (таблица А.2)

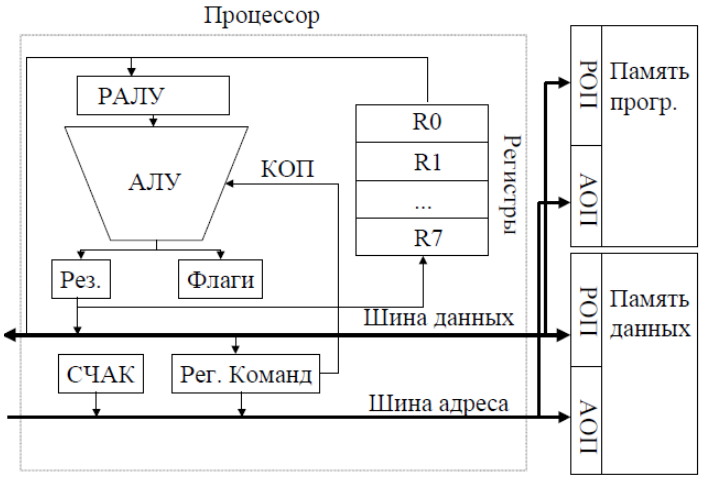

Рисунок А.2 – Структура 2

Объем оперативной памяти данных − 256 байтов, объем оперативной памяти программ − 256 байтов. Данные и программы хранятся раздельно.

Таблица А.2 – Система команд для структуры 2

Код	Мнемокод и операнды	Описание
00000	ADD РОН	рез=РОН+рез
00001	ADD адрес	рез=память[адрес]+рез
00010	ADC РОН	рез=РОН+рез+флаг переполнения
00011	SUB РОН	рез= рез-РОН
00100	SUB адрес	рез=рез-память[адрес]
00101	SBB РОН	рез=рез-РОН-флаг переполнения
00110	AND РОН	рез=РОН&рез
00111	AND адрес	рез=память[адрес]&рез
01000	OR РОН	рез=РОН | рез.
01001	OR адрес	рез=память[адрес] | рез
01010	XOR РОН	рез=РОН⊗рез.
01011	XOR адрес	рез=память[адрес]⊗рез
01100	MUL РОН	рез=рез*РОН
01101	DIV РОН	рез=рез/РОН (деление нацело, остаток – в РОН+1)
01110	LDR РОН	рез=РОН
01111	SVR РОН	РОН=рез
10000	CALL РОН,адрес	переход к подпрограмме; адрес возврата сохраняется в РОНе;
10001	RET РОН	возврат из подпрограммы; адрес возврата находится в РОНе
10010	INC РОН	увеличение РОНа на 1
10011	DEC РОН	уменьшение РОНа на 1
10100	MOV РОН,константа	занести в РОН константу
10101	MOV РОН,адрес	РОН= память[адрес]
10110	MOV память,РОН	память[адрес]=РОН
10111	LOAD РОН	рез=память[адрес=РОН]
11000	SAVE РОН	память[адрес=РОН]=рез
11001	JMP адрес	безусловный переход
11010	JZ адрес	переход, если 0
11011	JNZ адрес	переход, если не 0
11110	HALT	останов
11111	CLR	установка регистра результата в 0

Команды имеют длину один или два байта. Первый байт определяет код команды в соответствии с таблицей команд (бит 0-4), а также номер регистра-операнда (биты 5-7), указывающего на один из РОНов процессора.

Второй байт определяет адрес операнда в памяти, или задает константу.

Арифметические и логические операции выполняются над операндом, находящимся в памяти или регистре, и результатом предыдущей операции, который находится в регистре результатов. Флаг переполнения учитывается как единица младшего разряда, если используется в арифметических командах.

Структура 3 (рисунок А.3). Описание и система команд (таблица А.3)

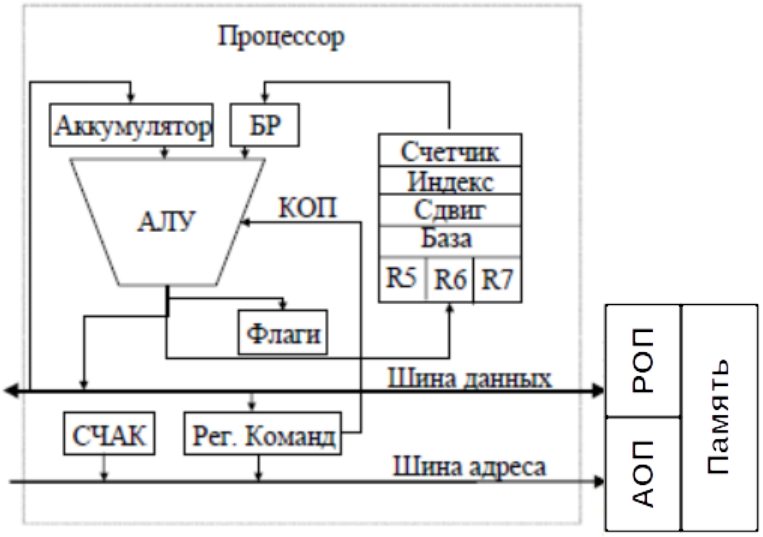

Рисунок А.3 – Структура 3

Объем оперативной памяти − 256 байт. Микропроцессор имеет 8 регистров, 5 из них − специального назначения:

РОН0 (AC). аккумулятор (в него заносятся результаты выполнения всех операций);
РОН 1 (CNT) . счетчик (для операций увеличения (уменьшения) на 1);
РОН 2 (SH) . сдвиговый регистр (для выполнения команд сдвига);
РОН 3 (BASE) . регистр базы (содержит базовый адрес операнда в ОП);
РОН 4 (IND) . регистр индекса (для формирования смещения операнда в ОП).

Таблица А.3 – Система команд для структуры 3

Код	Мнемокод	Описание
0000	ADD регистр	аккумулятор= аккумулятор + регистр
0001	ADC регистр	аккумулятор= аккумулятор + регистр + флаг переполнения
0010	SUB регистр	аккумулятор= аккумулятор - регистр
0011	SBB регистр	аккумулятор= аккумулятор - регистр - флаг переполнения
0100	AND регистр	аккумулятор=аккумулятор & регистр
0101	OR регистр	аккумулятор=аккумулятор | регистр
0110	NOT	аккумулятор=не аккумулятор
0111	MOV регистр, константа	регистр=константа
1000	STOS	память[BASE+IND]= аккумулятор IND=IND+1
1000	LODS*	Аккумулятор =память[BASE+IND], IND=IND+1
1001	OUTR регистр, адрес	память[адрес]= регистр
1001	INR* регистр, адрес	регистр= память[адрес]
1010	LOOP адрес	переход по адресу, если CNT>0, CNT=CNT-1
1011	JNZ адрес	переход, если не 0
1100	JB адрес	переход, если результат меньше 0
1101	SWAP регистр	Обмен данными между аккумулятором и регистром
1110	SHR	Сдвиг вправо регистра SH на число разрядов в регистре CNT
1111	SHL	Сдвиг влево регистра SH на число разрядов в регистре CNT

Коды команд:
Команды имеют длину один или два байта. Биты 0-3 первого байта определяют код команды в соответствии с таблицей команд. Номер регистра хранится в битах 4-6 первого байта. Бит 7 указывает направление перемещения данных: 0 означает перемещение данных *в* оперативную память, в остальных случаях он равен 1. Второй байт определяет константу или адреса памяти.

Арифметические и логические операции выполняются над операндами, находящимися в аккумуляторе и (или) регистре.

Структура 4 (рисунок А.4). Описание и система команд (таблица А.4)

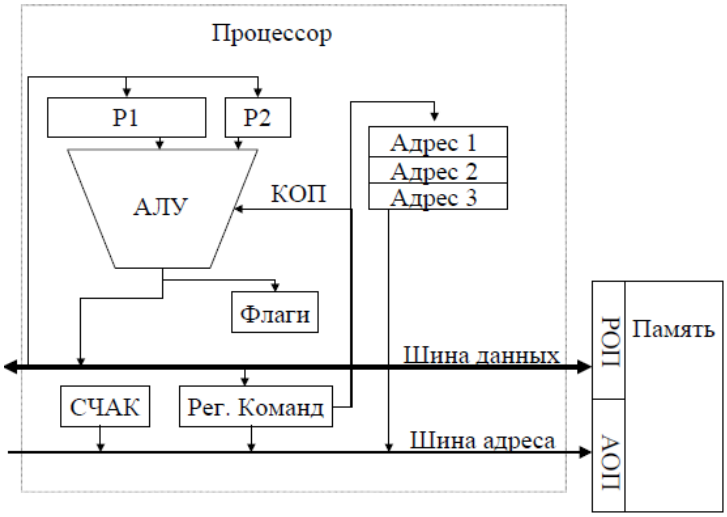

Рисунок А.4 – Структура 4

Объем оперативной памяти . 2048 байт (8 страниц по 256 байт).

Регистр СЧАК имеет 12 битов, но старший бит не используется.

Микропроцессор имеет три адресных регистра. Регистры АДРЕС1 (А1) и АДРЕС2 (А2) содержат адреса операндов команд, регистр АДРЕС3 (А3) . адрес результата операции. Эти регистры имеют по 8 битов и определяют смещение в диапазоне 0-255 внутри страницы. Номер страницы (целое число от 0 до 7) задан в битах 5-7 регистра команд.

Команды имеют длину один или два байта. Биты 0-4 первого байта определяюткод команды в соответствии с таблицей команд. Биты 5-7 первого байта задают номер страницы ОП, в которой располагаются данные. Второй байт определяет константу.

Таблица А.4 – Система команд для структуры 4

Код	Мнемокод	Описание
00000	MOVA1 константа	A1=константа
00001	MOVA2 константа	A2=константа
00010	MOVA3 константа	A3=константа
00110	SUB страница	Память[A3]= Память[A1]- Память[A2]
00111	SBB страница	Память[A3]= Память[A1]- Память[AC2] -флаг переп.
01000	MUL страница	Память[A3]= Память[A1]* Память[A2]
01001	DIV страница	Память[A3]= Память[A1]/ Память[A2]
01010	MOD страница	Память[A3]= Память[A1]% Память[A2]
01011	ABS страница	Память[A3]= | Память[A1] | **
01100	AND страница	Память[A3]=Память[A1]& Память[A2]
01101	OR страница	Память[A3]= Память[A1] | Память[A2]
01110	XOR страница	Память[A3]= Память[A1] ^ Память[A2]
01111	NOT страница	Память[A3]=not Память[A1]
10000	JMP страница	Переход по адресу A3
10001	JB страница	Переход по адресу A3, если результат меньше 0
10010	JNZ страница	Переход по адресу A3, если результат не равен 0
10101	MOVS	Память[A3]=Память[A1], A3+=1, A1+=1
10101	CMPS	Сравнение данных Память[A1] и Память[A3]***

** отрицательные числа представляются в дополнительном коде. Для нахождения модуля необхо-

димо выполнить перевод в прямой код и обнулить старший бит
*** выполняется аналогично операции вычитания, но результат операции не заносится в память

Структура 5 (рисунок А.5). Описание и система команд (таблица А.5)

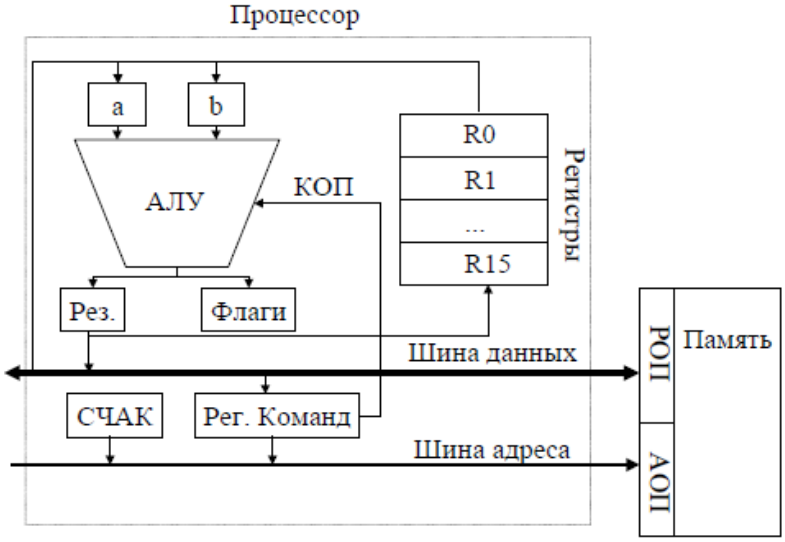

Рисунок А.5 – Структура 5

Объем оперативной памяти . 256 байт. Буферные регистры a и b предназначены дляхранения операндов операций. Рабочий регистр Рез используется как регистр результата, полученного при обработке данных в АЛУ.

Коды команд:

Команды имеют длину один или два байта. Первый байт определяет код команды в соответствии с таблицей команд (биты 0-3). Следующие 4 бита предназначены для указания номера регистра, коммутируемого к буферному регистру a.

Второй байт используется для задания константы или адреса памяти, где находится операнд. Если в качестве второго операнда используется регистр, то его номер хранится в младших 4-х битах второго байта команды, а старшие 4 бита не используются.

Таблица А.5 – Система команд для структуры 5

Код	Мнемокод	Описание
0000	ADNAB регистр,регистр	$\bar{a}+b$
0001	ADANB регистр,регистр	$a+\bar{b}$
0010	ADAB регистр,регистр	$a+b$
0011	ANBONAB регистр,регистр	$a\bar{b} \vee \bar{a}b$
0100	ABONANB регистр,регистр	$ab \vee \overline{a\bar{b}}$
0101	NAB регистр,регистр	$\bar{a}b$
0110	ANB регистр,регистр	$a\bar{b}$
0111	JZ адрес	Переход, если результат равен 0
1001	JL адрес	Переход, если результат меньше 0
1010	JMP адрес	Безусловный переход
1100	MOVC регистр, константа	Регистр:=Константа
1101	MOVR регистр, адрес	Регистр:= Память[адрес]
1110	MOVM регистр, адрес	Память[адрес]:=регистр
1111	CMP регистр, регистр	Сравнение двух регистров. Установка флагов.

Приложение Б

Детализация описания мини-проектов, представленных в п. 8.2.2

1. Калькулятор

Реализовать простейший калькулятор, который дает возможность пользователю складывать, умножать, вычитать, делить числа, находить их квадратный корень.

КАК пользователь
КОГДА я ввожу два числа
ТОГДА я увижу сумму двух чисел

КАК пользователь
КОГДА я ввожу два числа
ТОГДА я могу увидеть разность двух чисел

КАК пользователь
КОГДА я ввожу два числа
ТОГДА я увижу произведение двух чисел

КАК пользователь
КОГДА я ввожу два числа
И первое число является числителем
И второе является знаменателем и *не равно* нулю
ТОГДА я увижу частное двух чисел

КАК пользователь
КОГДА я ввожу два числа
И первое число является числителем
И второе является знаменателем и *равно* нулю
ТОГДА я должен увидеть ошибку

---

2. Генератор паролей

Создайте генератор случайных паролей для своих друзей и семьи, чтобы обезопасить их учётные записи!

Реализовать простейший генератор паролей, который дает возможность пользователю получить пароль, удовлетворяющий его требованиям.

КАК пользователь
КОГДА я открываю программу
И указываю количество обычных символов
ТОГДА я вижу сгенерированный пароль
И этот пароль содержит выбранное количество обычных символов

КАК пользователь
КОГДА я открываю программу

И указываю количество специальных символов
ТОГДА я вижу сгенерированный пароль
И этот пароль содержит выбранное количество специальных символов

КАК пользователь
КОГДА я открываю программу
И указываю количество букв в верхнем регистре
ТОГДА я вижу сгенерированный пароль
И этот пароль содержит выбранное количество букв в верхнем регистре

КАК пользователь
КОГДА я открываю программу
И указываю различные критерии(разное число обычных, специальных, в верхнем регистре символов и цифр)
ТОГДА я вижу сгенерированный пароль
И этот пароль должен удовлетворять *всем* требованиям

---

3. Игра-викторина

Приложения-викторины задают пользователям серию вопросов и дают им возможность ответить. Затем викторина даёт пользователю результаты. Реализовать программу-викторину на любую интересную вам тему. Да, можно про Наруто. Да, можно не про аниме. На любую тему :3

КАК пользователь
КОГДА я открываю программу
ТОГДА программа *предлагает мне сыграть в игру (с)* и ответить на 10 вопросов

КАК пользователь
КОГДА я не соглашаюсь сыграть в игру
ТОГДА программа благодарит за участие и завершается

КАК пользователь
КОГДА я соглашаюсь сыграть в игру
ТОГДА программа задает мне вопрос и дает возможность ответить

КАК пользователь
КОГДА я даю ответ на вопрос
ТОГДА программа задает мне следующий вопрос

КАК пользователь
КОГДА я даю ответ на последний (десятый) вопрос
ТОГДА программа показывает результаты(сколько очков заработано
И программа предлагает сыграть снова

---

4. Игра "Угадай число"

В данном мини-проекте вам необходимо реализовать игру, в которой компьютер пытается угадать целое число от 1 до 1000. Пользователь загадывает число, а программа задает ему вопросы из разряда *Да* или *Нет* в попытках угадать число. Если программа угадает число, она спрашивает у пользователя, это ли число она угадала. Если да - то урааа! Если нет - программа показывает грустный смайлик и предлагает новую игру. *Предполагается, что пользователь играет и отвечает на вопросы честно, а не как всегда*.

КАК пользователь
КОГДА я открываю программу
ТОГДА программа *предлагает мне сыграть в игру (с)*

КАК пользователь
КОГДА я не соглашаюсь сыграть в игру
ТОГДА программа благодарит за участие и завершается

КАК пользователь
КОГДА я соглашаюсь сыграть в игру
ТОГДА программа предлагает мне загадать число от 1 до 1000
И после этого начинает задавать вопросы из-разряда *Да/Нет*

КАК пользователь
КОГДА я даю ответ на вопрос *Да*
ТОГДА программа задает мне следующий вопрос

КАК пользователь
КОГДА я даю ответ на вопрос *Нет*
ТОГДА программа задает мне следующий вопрос

КАК пользователь
КОГДА я даю ответ на вопрос, который не является *Да* или *Нет*
ТОГДА программа просит ввести ответ снова, так как *у нее лапки*

КАК пользователь
КОГДА я даю ответ на вопрос
И программа предполагает что угадала число
ТОГДА программа спрашивает - такое ли число было загадано
И предлагает дать ответ на вопрос *Да* или *Нет*

КАК пользователь
КОГДА я даю ответ на вопрос
И программа угадала число
ТОГДА программа показывает результаты - как быстро она угадала, сколько шагов и какое число было загадано
И программа предлагает сыграть снова

КАК пользователь
КОГДА я даю ответ на вопрос
И программа не число
ТОГДА программа показывает грустный смайлик
И программа предлагает сыграть снова

5. Программа-шифровщик

В данном мини-проекте вам необходимо реализовать программу, которая позволяет как шифровать сообщение, так и дешифровать сообщение. Предполагается, что обладая знанием ключа/ключей, вы можете зашифровать сообщение, а ваш друг - расшифровать, и наоборот.

КАК пользователь
КОГДА я открываю программу
ТОГДА программа предлагает либо зашифровать текст, либо дешифровать текст

КАК пользователь
КОГДА я ввожу текст для шифрования
И я ввожу ключ для шифрования
ТОГДА программа показывает зашифрованный текст

КАК пользователь
КОГДА я ввожу текст для дешифрования
И я ввожу ключ для дешифрования
ТОГДА программа показывает расшифрованный текст

Приложение В

Варианты заданий для мини-проекта «Интеллектуальная матрица», представленного в п. 8.2.2

Вариант 1.
Разработать класс «Интеллектуальная матрица» (IntellectMatr).
Поля: 1) ссылка на прямоугольную (квадратную) матрицу (public), тип элементов матрицы – float.
Методы:
1) конструктор (параметр – ссылка на матрицу);
2) процедура вывода матрицы в окно терминала (без параметров);
3) функция, возвращающая значение суммы элементов главной диагонали матрицы (без параметров);
4) функция, возвращающая значение определителя матрицы (без параметров);
5) функция, возвращающая адрес вектора (одномерного массива), i-ый элемент которого равен количеству отрицательных элементов в i-ой строке матрицы (без параметров);
6) функция, возвращающая адрес транспонированной матрицы (без параметров).

Разработать класс MatrDemo, содержащий статические методы:
1) процедура вывода вектора (одномерного массива) в окно терминала (параметр – ссылка на вектор);
2) метод main, демонстрирующий применение методов класса «Интеллектуальная матрица» для двух матриц различной размерности.

Вариант 2. Разработать класс «Интеллектуальная матрица» (IntellectMatr).
Поля: 1) ссылка на прямоугольную (квадратную) матрицу (public), тип элементов матрицы – double.
Методы:
1) конструктор (параметр – ссылка на матрицу);
2) процедура вывода матрицы в окно терминала (без параметров);
3) функция, возвращающая значение суммы элементов под главной диагональю матрицы (без параметров);
4) функция, возвращающая значение определителя матрицы (без параметров);
5) функция, возвращающая адрес вектора (одномерного массива), i-ый элемент которого равен сумме положительных элементов i-ой строки матрицы (без параметров);
6) функция, возвращающая адрес матрицы-произведения инкапсулированной матрицы на другую матрицу (параметр – ссылка на матрицу).

Разработать класс MatrDemo, содержащий статические методы:
1) процедура вывода вектора (одномерного массива) в окно терминала (параметр – ссылка на вектор);
2) метод main, демонстрирующий применение методов класса «Интеллектуальная матрица» для двух матриц различной размерности.

Вариант 3. Разработать класс «Интеллектуальная матрица» (IntellectMatr).
Поля: 1) ссылка на прямоугольную (квадратную) матрицу (public), тип элементов матрицы – int.
Методы:
1) конструктор (параметр – ссылка на матрицу);
2) процедура вывода матрицы в окно терминала (без параметров);
3) функция, возвращающая значение произведения элементов над главной диагональю матрицы (без параметров);

4) функция, возвращающая значение определителя матрицы (без параметров);
5) функция, возвращающая адрес вектора (одномерного массива), i-ый элемент которого равен произведению положительных элементов i-го столбца матрицы (без параметров);
6) функция, возвращающая адрес транспонированной матрицы (без параметров).

Разработать класс MatrDemo, содержащий статические методы:
1) процедура вывода вектора (одномерного массива) в окно терминала (параметр – ссылка на вектор);
2) метод main, демонстрирующий применение методов класса «Интеллектуальная матрица» для двух матриц различной размерности.

Вариант 4. Разработать класс «Интеллектуальная матрица» (IntellectMatr).
Поля: 1) ссылка на прямоугольную (квадратную) матрицу (public), тип элементов матрицы – double.
Методы:
1) конструктор (параметр – ссылка на матрицу);
2) процедура вывода матрицы в окно терминала (без параметров);
3) функция, возвращающая значение максимального элемента под второй (не главной) диагональю матрицы (без параметров);
4) функция, возвращающая значение определителя матрицы (без параметров);
5) функция, возвращающая адрес вектора (одномерного массива), i-ый элемент которого равен числу положительных элементов в i-ом столбце матрицы (без параметров);
6) функция, возвращающая адрес матрицы-произведения инкапсулированной матрицы на другую матрицу (параметр – ссылка на матрицу).

Разработать класс MatrDemo, содержащий статические методы:
1) процедура вывода вектора (одномерного массива) в окно терминала (параметр – ссылка на вектор);
2) метод main, демонстрирующий применение методов класса «Интеллектуальная матрица» для двух матриц различной размерности.

Вариант 5. Разработать класс «Интеллектуальная матрица» (IntellectMatr).
Поля: 1) ссылка на прямоугольную (квадратную) матрицу (public), тип элементов матрицы – int.
Методы:
1) конструктор (параметр – ссылка на матрицу);
2) процедура вывода матрицы в окно терминала (без параметров);
3) функция, возвращающая значение минимального элемента над второй (не главной) диагональю матрицы (без параметров);
4) функция, возвращающая значение определителя матрицы (без параметров);
5) функция, возвращающая адрес вектора (одномерного массива), i-ый элемент которого равен единице, если в i-ом столбце матрицы есть положительные элементы, и нулю, в противном случае;
6) функция, возвращающая адрес транспонированной матрицы (без параметров).

Разработать класс MatrDemo, содержащий статические методы:
1) процедура вывода вектора (одномерного массива) в окно терминала (параметр – ссылка на вектор);
2) метод main, демонстрирующий применение методов класса «Интеллектуальная матрица» для двух матриц различной размерности.