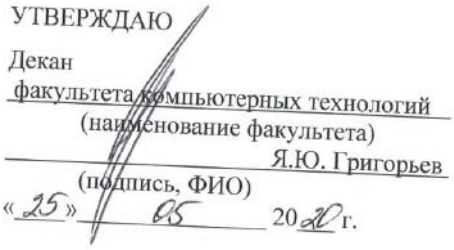
Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»



РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ
Системы интеллектуальной защиты информации

Направление подготовки	*10.05.03 "Информационная безопасность автоматизированных систем"*
Направленность (профиль) образовательной программы	*Обеспечение информационной безопасности распределенных информационных систем*
Квалификация выпускника	*специалист по защите информации*
Год начала подготовки *(по учебному плану)*	*2020*
Форма обучения	*очная*
Технология обучения	*традиционная*

Курс	Семестр	Трудоемкость, з.е.
5	*9*	*3*

Вид промежуточной аттестации	Обеспечивающее подразделение
Зач_с_оц	*Кафедра ИБАС - Информационная безопасность автоматизированных систем*

Комсомольск-на-Амуре 2020

Разработчик рабочей программы:

Доцент ИБАС к.э.н. _____ Власов А.А.
(должность, степень, ученое звание) (подпись) (ФИО)

СОГЛАСОВАНО:

Заведующий кафедрой
ИБАС
(наименование кафедры) _____ А.Ю. Лошманов
(подпись) (ФИО)

1 Общие положения

Рабочая программа и фонд оценочных средств дисциплины «Системы интеллектуальной защиты информации» составлены в соответствии с требованиями федерального государственного образовательного стандарта, утвержденного приказом Министерства образования и науки Российской Федерации № 1509 от 01.12.2016, и основной профессиональной образовательной программы подготовки «Обеспечение информационной безопасности распределенных информационных систем» по специальности 10.05.03 "Информационная безопасность автоматизированных систем".

Задачи дисциплины	-Изучение различных методов организации интеллектуальных систем защиты информации, классификации информации, современных подходов к построению защищенных систем - освоение теоретических основ разработки систем искусственного интеллекта; - овладение основными принципами проектирования экспертных систем; - выработка у студентов умения использовать в практическом программировании основные модели и методы поиска решений в различных пространствах состояний; - развитие навыков построения систем распознавания образов и речи.
Основные разделы / темы дисциплины	Классификация и кластеризация Распознавание образов

2 Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами образовательной программы

Процесс изучения дисциплины «Системы интеллектуальной защиты информации» направлен на формирование следующих компетенций в соответствии с ФГОС ВО и основной образовательной программой (таблица 1):

Таблица 1 – Компетенции и планируемые результаты обучения по дисциплине

Код и наименование компетенции	Планируемые результаты обучения по дисциплине		
	Перечень знаний	Перечень умений	Перечень навыков
Профессиональные			
ОПК-8 способностью к освоению новых образцов программных, технических средств и информационных технологий	**З3(ОПК-8-5)**: архитектуру, назначение и способы реализации компонент систем искусственного интеллекта способствующих освоению новых образцов программных, технических средств и информационных технологий ;	**У3(ОПК-8-5)**: Использовать математические модели и методы для реализации механизма логического вывода в соответствием с требованиями конкретной прикладной задачи;	**Н3(ОПК-8-5)**: Навыками построения приложений с использованием систем искусственного интеллекта при освоении новых образцов программных, технических средств и информационных технологий

3 Место дисциплины (модуля) в структуре образовательной программы

Дисциплина(модуль) «Системы интеллектуальной защиты информации» изучается на 5 курсе в 9 семестре.

Дисциплина является базовой дисциплиной, входит в состав блока 1 «Дисциплины (модули)» и относится к базовой части.

Для освоения дисциплины необходимы знания, умения, навыки и (или) опыт практической деятельности, сформированные в процессе изучения дисциплин / практик: Операционные системы, базы данных, альтернативные операционные системы.

Знания, умения и навыки, сформированные при изучении дисциплины «Системы интеллектуальной защиты информации», будут востребованы при изучении последующих дисциплин и выполнения выпускной квалификационной работы.

Дисциплина ««Системы интеллектуальной защиты информации»» в рамках воспитательной работы направлена на развитие творчества, профессиональных умений, ответственности за выполнение учебно-производственных заданий.

4 Объем дисциплины (модуля) в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Общая трудоемкость (объем) дисциплины составляет 3 зачетных единиц, 108 академических часов.

Распределение объема дисциплины (модуля) по видам учебных занятий представлено в таблице 2.

Таблица 2 – Объем дисциплины (модуля) по видам учебных занятий

Объем дисциплины	**Всего академических часов**
Общая трудоемкость дисциплины	108
Контактная аудиторная работа обучающихся с преподавателем (по видам учебных занятий), всего	48
В том числе:	
занятия лекционного типа (лекции и иные учебные занятия, предусматривающие преимущественную передачу учебной информации педагогическими работниками)	16
занятия семинарского типа (семинары, практические занятия, практикумы, лабораторные работы, коллоквиумы и иные аналогичные занятия)	32
Самостоятельная работа обучающихся и контактная работа, включающая групповые консультации, индивидуальную работу обучающихся с преподавателями (в том числе индивидуальные консультации); взаимодействие в электронной информационно-образовательной среде вуза	60
Промежуточная аттестация обучающихся – Зач_с_оц	

5 Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебной работы

Таблица 3 – Структура и содержание дисциплины (модуля)

Наименование разделов, тем и содержание материала	Виды учебной работы, включая самостоятельную работу обучающихся и трудоемкость (в часах)			
	Контактная работа преподавателя с обучающимися			СРС
	Лекции	Семинарские (практические занятия)	Лабораторные занятия	
Классификация. Описание и формализация проблемы. Различие форм описания проблем и его влияние на компьютеризацию. Формы представления информации на различных этапах понимания. Различия в постановке проблемы из-за различия в подходах к объекту. Классификация типов проблем и методы их решения. Область проблем, относящихся к искусственному интеллекту. ЭС, как отдельная область ИИ. Формальные основы ЭС. Архитектура ЭС. Методология построения ЭС. Классификация. Представление знаний в интеллектуальных системах. Основные понятия и определения. Модели представления знаний (повторение). Состав знаний ЭС. Структура и организация знаний в ЭС. Методы приобретения знаний. Системы понимания естественного языка. Морфологический, синтаксический и семантический анализ входных сообщений. Методы анализа входных сообщений. Системы распознавания образов. Тенденции развития систем искусственного интеллекта в задачах информационной безопасности. Аппроксимация функции одной переменной с помощью многослойной нейронной. Кластеризация с помощью нейронных сетей. Аппроксимация функции одной переменной с помощью аппарата нечеткой логики. Построение нечеткой экспертной системы. Кластеризация с помощью алгоритма нечетких центров	8		16	30
Кластеризация Задачи классификация и кластеризация.	8		16	30

Наименование разделов, тем и содержание материала	Виды учебной работы, включая самостоятельную работу обучающихся и трудоемкость (в часах)			
	Контактная работа преподавателя с обучающимися			СРС
	Лекции	Семинарские (практические занятия)	Лабораторные занятия	
нейронные сети. машинное зрение. Обучение нейронных сетей. Реализация на различных языках и платформах.				
ИТОГО по дисциплине	**16**		**32**	**60**

6 Внеаудиторная самостоятельная работа обучающихся по дисциплине (модулю)

При планировании самостоятельной работы студенту рекомендуется руководствоваться следующим распределением часов на самостоятельную работу (таблица 4):

Таблица 4 – Рекомендуемое распределение часов на самостоятельную работу

Компоненты самостоятельной работы	Количество часов
Изучение теоретических разделов дисциплины	5
Подготовка к занятиям семинарского типа	5
Подготовка и оформление РГР.	50
Всего	60

7 Оценочные средства для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю)

Фонд оценочных средств для проведения текущего контроля успеваемости и промежуточной аттестации представлен в Приложении 1.

Полный комплект контрольных заданий или иных материалов, необходимых для оценивания результатов обучения по дисциплине (модулю), практике хранится на кафедре-разработчике в бумажном и электронном виде.

8 Учебно-методическое и информационное обеспечение дисциплины (модуля)

8.1 Основная литература

1 Антамошкин, О. А. Системы интеллектуальной защиты информации. Теория и практика [Электрон-ный ресурс] : учебник / О. А. Антамошкин. - Красноярск: Сиб. Федер. ун-т, 2012. - 247 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

2 Практическая Системы интеллектуальной защиты информации на основе учебного примера: Учебное пособие / Мацяшек Л.А., Лионг Б.Л., - 3-е изд., (эл.) - М.:БИНОМ. Лаб. знаний, 2015. - 959 с // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

3 Объектно-ориентированное программирование с примерами на C#: Учебное

пособие / Хорев П.Б. - М.: Форум, НИЦ ИНФРА-М, 2016. - 200 с // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

8.2 Дополнительная литература

1 Введение в программную инженерию : Учебник / В.А. Антипов, А.А. Бубнов, А.Н. Пылькин, В.К. Столчнев. — М.: КУРС: ИНФРА-М, 2017. — 336 с // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

2 Объектно-ориентированное программирование на Visual Basic в среде Visual Studio .Net/В.Н.Шакин, А.В.Загвоздкина, Г.К.Сосновиков - М.: Форум,ИНФРА-М, 2015. - 400 с // ZNANIUM.COM : электронно-библиотечная система. – Ре- жим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

3 Практикум по объектно-ориентированному программированию / Бабушкина И.А., Окулов С.М., - 4-е изд. - М.:БИНОМ. ЛЗ, 2015. - 369 с. // ZNANIUM.COM : элек-тронно-библиотечная система. – Ре- жим доступа: http://znanium.com/catalog.php#, ограни-ченный. – Загл. с экрана.

4 Мейер Б. Объектно-ориентированное программирование и программная инже-нерия [Электронный ресурс]/ Мейер Б.— Электрон. текстовые данные.— М.: Интернет-Университет Информационных Технологий (ИНТУИТ), Ай Пи Эр Медиа, 2019.— 285 с.— Режим доступа: http://www.iprbookshop.ru/79706.html.— ЭБС «IPRbooks»

8.3 Методические указания для студентов по освоению дисциплины

Обучение дисциплине «Системы интеллектуальной защиты информации» предполагает изучение курса на аудиторных занятиях и в ходе самостоятельной работы. Аудиторные занятия проводятся в форме лекций и лабораторных занятий.

Таблица 7 Методические указания к отдельным видам деятельности

Вид учебного занятия	Организация деятельности студента
Лекция	Написание конспекта лекций: кратко, схематично, последовательно фиксировать основные положения. Выделять ключевые слова, формулы, отмечать на полях уточняющие вопросы по теме занятия
Лабораторные занятие	Работа с автоматизированными рабочими местами.
Самостоятельная работа	Для более глубокого изучения разделов дисциплины предусмотрены отдельные виды самостоятельной работы: подготовка к лабораторным занятиям, изучение теоретических разделов дисциплины, подготовка КР.

Самостоятельная работа является наиболее продуктивной формой образовательной и познавательной деятельности студента в период обучения. СРС направлена на углубление и закрепление знаний студента, развитие практических умений. СРС по дисциплине «Системы интеллектуальной защиты информации» включает следующие виды работ:

– работу с лекционным материалом, поиск и обзор литературы и электронных источников информации по индивидуальному заданию;

– опережающую самостоятельную работу;

– изучение тем, вынесенных на самостоятельную проработку;

– подготовку к практическим занятиям;

– выполнение и оформление контрольной работы.

Контроль самостоятельной работы студентов и качество освоения дисциплины осуществляется посредством:

– представления в указанные контрольные сроки результатов выполнения заданий для текущего контроля;

– выполнения и защиты контрольной работы;

Контрольная работа и отчеты по лабораторным работам должны быть оформлены в соответствии с требованиями внутренних нормативных документов ФГБОУ ВО КнАГУ.

8.4 Современные профессиональные базы данных и информационные справочные системы, используемые при осуществлении образовательного процесса по дисциплине

1. Электронно-библиотечная система ZNANIUM.COM – **Ошибка! Недопустимый объект гиперссылки.**.
2. Консультант+

8.5 Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины (модуля)

1. Научная электронная библиотека Elibrary http://elibrary.ru.

С целью повышения качества ведения образовательной деятельности в университете создана электронная информационно-образовательная среда. Она подразумевает организацию взаимодействия между обучающимися и преподавателями через систему личных кабинетов студентов, расположенных на официальном сайте университета в информационно-телекоммуникационной сети «Интернет» по адресу https://student.knastu.ru. Созданная информационно-образовательная среда позволяет осуществлять взаимодействие между участниками образовательного процесса посредством организации дистанционного консультирования по вопросам выполнения практических заданий.

8.6 Лицензионное программное обеспечение, используемое при осуществлении образовательного процесса по дисциплине

Таблица 5 – Перечень используемого программного обеспечения

Наименование ПО	Реквизиты
Microsoft® Windows Professional 7 Russian	Лицензионный сертификат № 46243844 от 09.12.2009
Open Office или аналог	Свободно-распространяемое
Операционная система Kali Linux или аналог содержащая необходимые модули для анализа защищенности	Свободно-распространяемое
Операционная система Ubuntu или аналог	Свободно-распространяемое
Гипервизор Virtual Box или аналог	Свободно-распространяемое
Обозреватель Google Chrome	Свободно-распространяемое

или аналог	
Виртуальные машины согласно перечню из фондов оценочных средств для дисциплины	Свободно-распространяемое
Parrot OS	Свободно-распространяемое

9 Организационно-педагогические условия

Организация образовательного процесса регламентируется учебным планом и расписанием учебных занятий. Язык обучения (преподавания) — русский. Для всех видов аудиторных занятий академический час устанавливается продолжительностью 45 минут.

При формировании своей индивидуальной образовательной траектории обучающийся имеет право на перезачет соответствующих дисциплин и профессиональных модулей, освоенных в процессе предшествующего обучения, который освобождает обучающегося от необходимости их повторного освоения.

9.1 Образовательные технологии

Учебный процесс при преподавании курса основывается на использовании традиционных, инновационных и информационных образовательных технологий. Традиционные образовательные технологии представлены лекциями и семинарскими (практическими) занятиями. Инновационные образовательные технологии используются в виде широкого применения активных и интерактивных форм проведения занятий. Информационные образовательные технологии реализуются путем активизации самостоятельной работы студентов в информационной образовательной среде.

9.2 Занятия лекционного типа

Лекционный курс предполагает систематизированное изложение основных вопросов учебного плана.

На первой лекции лектор обязан предупредить студентов, применительно к какому базовому учебнику (учебникам, учебным пособиям) будет прочитан курс.

Лекционный курс должен давать наибольший объем информации и обеспечивать более глубокое понимание учебных вопросов при значительно меньшей затрате времени, чем это требуется большинству студентов на самостоятельное изучение материала.

9.3 Занятия семинарского типа

Семинарские занятия представляют собой детализацию лекционного теоретического материала, проводятся в целях закрепления курса и охватывают все основные разделы.

Основной формой проведения семинаров является обсуждение наиболее проблемных и сложных вопросов по отдельным темам, а также разбор примеров и ситуаций в аудиторных условиях. В обязанности преподавателя входят: оказание методической помощи и консультирование студентов по соответствующим темам курса.

Активность на семинарских занятиях оценивается по следующим критериям:

· ответы на вопросы, предлагаемые преподавателем;

· участие в дискуссиях;

· выполнение проектных и иных заданий;

· ассистирование преподавателю в проведении занятий.

Ответ должен быть аргументированным, развернутым, не односложным, содержать ссылки на источники.

Доклады и оппонирование докладов проверяют степень владения теоретическим материалом, а также корректность и строгость рассуждений.

Оценивание заданий, выполненных на семинарском занятии, входит в накопленную оценку.

9.4 Самостоятельная работа обучающихся по дисциплине (модулю)

Самостоятельная работа студентов – это процесс активного, целенаправленного приобретения студентом новых знаний, умений без непосредственного участия преподавателя, характеризующийся предметной направленностью, эффективным контролем и оценкой результатов деятельности обучающегося.

Цели самостоятельной работы:

· систематизация и закрепление полученных теоретических знаний и практических умений студентов;

· углубление и расширение теоретических знаний;

· формирование умений использовать нормативную и справочную документацию, специальную литературу;

· развитие познавательных способностей, активности студентов, ответственности и организованности;

· формирование самостоятельности мышления, творческой инициативы, способностей к саморазвитию, самосовершенствованию и самореализации;

· развитие исследовательских умений и академических навыков.

Самостоятельная работа может осуществляться индивидуально или группами студентов в зависимости от цели, объема, уровня сложности, конкретной тематики.

Технология организации самостоятельной работы студентов включает использование информационных и материально-технических ресурсов университета.

Контроль результатов внеаудиторной самостоятельной работы студентов может проходить в письменной, устной или смешанной форме.

Студенты должны подходить к самостоятельной работе как к наиважнейшему средству закрепления и развития теоретических знаний, выработке единства взглядов на отдельные вопросы курса, приобретения определенных навыков и использования профессиональной литературы.

В данной дисциплине в рамках самостоятельной работы студенты выполняют одну курсовую работу состоящую из двух частей.

9.5 Методические указания для обучающихся по освоению дисциплины

При изучении дисциплины обучающимся целесообразно выполнять следующие рекомендации:

1. Изучение учебной дисциплины должно вестись систематически.

2. После изучения какого-либо раздела по учебнику или конспектным материалам рекомендуется по памяти воспроизвести основные термины, определения, понятия раздела.

3. Особое внимание следует уделить выполнению отчетов по практическим занятиям и индивидуальным комплексным заданиям на самостоятельную работу.

4. Вся тематика вопросов, изучаемых самостоятельно, задается на лекциях преподавателем. Им же даются источники (в первую очередь вновь изданные в периодической научной литературе) для более детального понимания вопросов, озвученных на лекции.

При самостоятельной проработке курса обучающиеся должны:

· просматривать основные определения и факты;

· повторить законспектированный на лекционном занятии материал и дополнить его с учетом рекомендованной по данной теме литературы;

· изучить рекомендованную литературу, составлять тезисы, аннотации и конспекты наиболее важных моментов;

· самостоятельно выполнять задания, аналогичные предлагаемым на занятиях;

· использовать для самопроверки материалы фонда оценочных средств.

1. Методические указания при работе над конспектом лекции

В ходе лекционных занятий необходимо вести конспектирование учебного материала. Обращать внимание на категории, формулировки, раскрывающие содержание тех или иных явлений и процессов, научные выводы и практические рекомендации, положительный опыт в ораторском искусстве. Желательно оставить в рабочих конспектах поля, на которых делать пометки из рекомендованной литературы, дополняющие материал прослушанной лекции, а также подчеркивающие особую важность тех или иных теоретических положений. Задавать преподавателю уточняющие вопросы с целью уяснения теоретических положений, разрешения спорных ситуаций.

2. Методические указания по самостоятельной работе над изучаемым материалом и при подготовке к лабораторным занятиям

Начинать надо с изучения рекомендованной литературы. Необходимо помнить, что на лекции обычно рассматривается не весь материал, а только его часть. Остальная его часть восполняется в процессе самостоятельной работы. В связи с этим работа с рекомендованной литературой обязательна. Особое внимание при этом необходимо обратить на содержание основных положений и выводов, объяснение явлений и фактов, уяснение практического приложения рассматриваемых теоретических вопросов. В процессе этой работы необходимо стремиться понять и запомнить основные положения рассматриваемого материала, примеры, поясняющие его, а также разобраться в иллюстративном материале. Оформлять отчеты следует руководствуясь внутренними нормативными документами КнАГУ.

3. Методические указания по выполнению расчетно-графической работы

Теоретическая часть расчетно-графической работы выполняется по установленным темам с использованием практических материалов. К каждой теме курсовой работы рекомендуется примерный перечень узловых вопросов, список необходимой литературы. Излагая вопросы темы, следует строго придерживаться плана. Работа не должна представлять пересказ отдельных глав учебника или учебного пособия. Необходимо изложить собственные соображения по существу излагаемых вопросов, внести свои предложения. Общие положения должны быть подкреплены и пояснены конкретными примерами. Излагаемый материал при необходимости следует проиллюстрировать таблицами, схемами, диаграммами.

10 Описание материально-технического обеспечения, необходимого для осуществления образовательного процесса по дисциплине (модулю)

10.1 Учебно-лабораторное оборудование

Таблица 6 – Перечень оборудования лаборатории

Аудитория	Наименование аудитории (лаборатории)	Используемое оборудование
202/5	Лаборатория программно-аппаратных средств защиты информации	СЗИ НСД Secret Net, СЗИ НСД Dallas Lock, СЗИ НСД Страж NT, СЗИ НСД Щит РЖД, СЗИ НСД Аура ,СЗИ НСД Криптон ,СЗИ НСД Аккорд, ФИКС, Ревизор 1,2 как для операционных систем семейства Windows так и для Linux, Ревизор Сети 2.0, Анализатор сетевого траффика Астра,Агент инвентаризации сети,Сканер сетевой безопасности XSpider, Терьер, Secret Net Touch Memory Card, Криптон АМДЗ, Аккорд АМДЗ, КриптоПРО АРМ, ,CryptoPro CSP 3.6, VipNet firewall, Etoken PKI Client, Etoken, Ноутбук с Windows 7+проектор.

		16 ПЭВМ на базе процессоров не ниже Intel Pentium IV

10.2 Технические и электронные средства обучения

Лекционные занятия

Аудитории для лекционных занятий укомплектованы мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории (наборы демонстрационного оборудования (проектор, экран, компьютер/ноутбук), учебно-наглядные пособия, тематические иллюстрации).

Лабораторные занятия

Для лабораторных занятий используется аудитория №_202_, оснащенная оборудованием, указанным в табл. 8:

Самостоятельная работа.

Помещения для самостоятельной работы оснащены компьютерной техникой с возможностью подключения к сети «Интернет» и доступом к электронной информационно-образовательной среде КнАГУ:

- читальный зал НТБ КнАГУ;
- компьютерные классы (ауд. 311 корпус № 5, ауд. 205 корпус № 5, ауд. 313 корпус № 5).

11 Иные сведения

Методические рекомендации по обучению лиц с ограниченными возможностями здоровья и инвалидов

Освоение дисциплины обучающимися с ограниченными возможностями здоровья может быть организовано как совместно с другими обучающимися, так и в отдельных группах. Предполагаются специальные условия для получения образования обучающимися с ограниченными возможностями здоровья.

Профессорско-педагогический состав знакомится с психолого-физиологическими особенностями обучающихся инвалидов и лиц с ограниченными возможностями здоровья, индивидуальными программами реабилитации инвалидов (при наличии). При необходимости осуществляется дополнительная поддержка преподавания тьюторами, психологами, социальными работниками, прошедшими подготовку ассистентами.

В соответствии с методическими рекомендациями Минобрнауки РФ (утв. 8 апреля 2014 г. N АК-44/05вн) в курсе предполагается использовать социально-активные и рефлексивные методы обучения, технологии социокультурной реабилитации с целью оказания помощи в установлении полноценных межличностных отношений с другими студентами, создании комфортного психологического климата в студенческой группе. Подбор и разработка учебных материалов производятся с учетом предоставления материала в различных формах: аудиальной, визуальной, с использованием специальных технических средств и информационных систем.

Освоение дисциплины лицами с ОВЗ осуществляется с использованием средств обучения общего и специального назначения (персонального и коллективного использования). Материально-техническое обеспечение предусматривает приспособление аудиторий к нуждам лиц с ОВЗ.

Форма проведения аттестации для студентов-инвалидов устанавливается с учетом индивидуальных психофизических особенностей. Для студентов с ОВЗ предусматривается доступная форма предоставления заданий оценочных средств, а именно:

· в печатной или электронной форме (для лиц с нарушениями опорно-двигательного аппарата);

· в печатной форме или электронной форме с увеличенным шрифтом и контрастностью (для лиц с нарушениями слуха, речи, зрения);

· методом чтения ассистентом задания вслух (для лиц с нарушениями зрения).

Студентам с инвалидностью увеличивается время на подготовку ответов на контрольные вопросы. Для таких студентов предусматривается доступная форма предоставления ответов на задания, а именно:

· письменно на бумаге или набором ответов на компьютере (для лиц с нарушениями слуха, речи);

· выбором ответа из возможных вариантов с использованием услуг ассистента (для лиц с нарушениями опорно-двигательного аппарата);

· устно (для лиц с нарушениями зрения, опорно-двигательного аппарата).

При необходимости для обучающихся с инвалидностью процедура оценивания результатов обучения может проводиться в несколько этапов.

Приложение 1

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ[1]
по дисциплине

Системы интеллектуальной защиты информации

Направление подготовки	*10.05.03 "Информационная безопасность автоматизированных систем"*
Направленность (профиль) образовательной программы	*Обеспечение информационной безопасности распределенных информационных систем*
Квалификация выпускника	*специалист по защите информации*
Год начала подготовки *(по учебному плану)*	*2020*
Форма обучения	*очная*
Технология обучения	*традиционная*

Курс	Семестр	Трудоемкость, з.е.
5	*9*	*3*

Вид промежуточной аттестации	Обеспечивающее подразделение
Зач_с_оц	*Кафедра ИБАС - Информационная безопасность автоматизированных систем*

[1] В данном приложении представлены типовые оценочные средства. Полный комплект оценочных средств, включающий все варианты заданий (тестов, контрольных работ и др.), предлагаемых обучающемуся, хранится на кафедре в бумажном и электронном виде.

1 Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами образовательной программы

Таблица 1 – Компетенции и планируемые результаты обучения по дисциплине

Код и наименование компетенции	Планируемые результаты обучения по дисциплине		
	Перечень знаний	Перечень умений	Перечень навыков
Профессиональные			
ОПК-8 способностью к освоению новых образцов программных, технических средств и информационных технологий	**ЗЗ(ОПК-8-5)**: архитектуру, назначение и способы реализации компонент систем искусственного интеллекта способствующих освоению новых образцов программных, технических средств и информационных технологий ;	**УЗ(ОПК-8-5)**: Использовать математические модели и методы для реализации механизма логического вывода в соответствием с требованиями конкретной прикладной задачи;	**НЗ(ОПК-8-5)**: Навыками построения приложений с использованием систем искусственного интеллекта при освоении новых образцов программных, технических средств и информационных технологий

Таблица 2 – Паспорт фонда оценочных средств

Контролируемые разделы (темы) дисциплины	**Формируемая компетенция**	**Наименование оценочного средства**	**Показатели оценки**
Классификация. Функции, экспертные системы	ОПК-8	Лабораторная работа 1	Умение проводить нечеткую классификацию
Кластеризация. Экспертные системы	ОПК-8	Лабораторная работа 2	Умение проводить нечеткую кластеризацию
Нейронные сети, машинное обучение	ОПК-8	Лабораторная работа 3	Умение проектировать и создавать НС
Нейронные сети, машинное обучение	ОПК-8	Лабораторная работа 4	Умение обучать НС интегрировать их в СЗИ
Все разделы	ОПК-8	Расчетно-графическая работа	Знания, навыки и умения по всем разделам курса

Промежуточная аттестация в седьмом семестре проводится в форме зачета с оценкой.

2 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующие процесс формирования компетенций

Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, представлены в виде технологической карты дисциплины (таблица 3).

Таблица 3 – Технологическая карта

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
_____ 7 семестр Промежуточная аттестация в форме зачета				
1	Лабораторная работа 1 № 1	В течение семестра	25 баллов	25 баллов - студент правильно выполнил задание. Показал отличные знания, навыки и умения в рамках освоенного учебного материала. 20 балла - студент выполнил задание, с небольшими неточностями. Показал хорошие знания, навыки и умения в рамках освоенного учебного материала. 15 балла - студент выполнил задание с существенными неточностями. Показал удовлетворительные знания, навыки и умения в рамках освоенного учебного материала. 8 балла - при выполнении задания студент продемонстрировал недостаточный уровень знаний. 0 баллов – задание не выполнено.
2	Лабораторная работа № 2	В течение семестра	25 баллов	25 баллов - студент правильно выполнил задание. Показал отличные знания, навыки и умения в рамках освоенного учебного материала. 20 балла - студент выполнил задание с небольшими неточностями. Показал хорошие знания, навыки и умения в рамках освоенного учебного материала. 15 балла - студент выполнил задание с существенными неточностями. Показал удовлетворительные знания, навыки и умения в рамках освоенного учебного материала. 8 балла - при выполнении задания студент продемонстрировал

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
				недостаточный уровень знаний. 0 баллов – задание не выполнено.
3	Лабораторная работа № 3	В течение семестра	25 баллов	25 баллов - студент правильно выполнил задание. Показал отличные знания, навыки и умения в рамках освоенного учебного материала. 20 балла - студент выполнил задание с небольшими неточностями. Показал хорошие знания, навыки и умения в рамках освоенного учебного материала. 15 балла - студент выполнил задание с существенными неточностями. Показал удовлетворительные знания, навыки и умения в рамках освоенного учебного материала. 8 балла - при выполнении задания студент продемонстрировал недостаточный уровень знаний. 0 баллов – задание не выполнено.
4	Лабораторная работа № 4	В течение семестра	25 баллов	25 баллов - студент правильно выполнил задание. Показал отличные знания, навыки и умения в рамках освоенного учебного материала. 20 балла - студент выполнил задание с небольшими неточностями. Показал хорошие знания, навыки и умения в рамках освоенного учебного материала. 15 балла - студент выполнил задание с существенными неточностями. Показал удовлетворительные знания, навыки и умения в рамках освоенного учебного материала. 8 балла - при выполнении задания студент продемонстрировал недостаточный уровень знаний. 0 баллов – задание не выполнено.
5	Расчетно-графическая работа	В течение семестра	25 баллов	25 баллов - студент правильно выполнил задание. Показал отличные знания, навыки и умения в рамках освоенного

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
				учебного материала. 20 балла - студент выполнил задание с небольшими неточностями. Показал хорошие знания, навыки и умения в рамках освоенного учебного материала. 15 балла - студент выполнил задание с существенными неточностями. Показал удовлетворительные знания, навыки и умения в рамках освоенного учебного материала. 8 балла - при выполнении задания студент продемонстрировал недостаточный уровень знаний. 0 баллов – задание не выполнено.
ИТОГО:		-	125 баллов	-
Критерии оценки результатов обучения по дисциплине: 0 – 64 % от максимально возможной суммы баллов – «неудовлетворительно» (недостаточный уровень для промежуточной аттестации по дисциплине); 65 – 74 % от максимально возможной суммы баллов – «удовлетворительно» (пороговый (минимальный) уровень); 75 – 84 % от максимально возможной суммы баллов – «хорошо» (средний уровень); 85 – 100 % от максимально возможной суммы баллов – «отлично» (высокий (максимальный) уровень)				

3 Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующие процесс формирования компетенций в ходе освоения образовательной программы

3.1 Задания для текущего контроля успеваемости

Лабораторная работа 1

Аппроксимация функции одной переменной с помощью многослойной нейронной.

Цель работы: освоить методики построения нечетких систем. Используя нечеткую систему научится аппроксимировать функцию одной переменной:

Задание: используя среду разработки VisualStudio построить нечеткую систему, необходимую для аппроксимации табличной функции $y_i=f(x_i)$, i-1,10. Варианты задания представлены в табл. 6.1

Таблица 6.1

i	Значение $y_i = y_i(x_i)$									
	№1	№2	№3	№4	№5	№6	№7	№8	№9	№10
1	1.50	2.09	2.02	1.99	2.23	2.07	2.18	-0.10	-0.16	2.09
2	1.26	2.05	1.98	2.03	2.29	2.17	2.43	-0.21	0.01	2.31
3	0.99	2.19	1.67	2.20	2.27	2.21	2.40	0.01	0.10	2.72
4	0.97	2.18	1.65	2.39	2.62	2.31	2.43	0.05	0.16	2.77
5	0.91	2.17	1.57	2.19	2.72	2.10	2.65	-0.13	0.05	2.78
6	0.71	2.27	1.42	2.61	2.82	2.09	2.75	-0.23	0.35	2.97
7	0.43	2.58	1.37	2.35	3.13	2.12	2.67	-0.21	0.19	3.00
8	0.54	2.73	1.07	2.60	3.49	1.63	2.66	-0.43	0.50	3.51
9	0.19	2.82	0.85	2.55	3.82	1.78	2.63	-0.57	0.74	3.43
10	0.01	3.04	0.48	2.49	3.95	1.52	2.75	-0.44	1.03	3.58

Кластеризация с помощью нейронных сетей

Цель работы: освоить основные принципы решения задачи кластеризации с использованием нейронных сетей со слоем Кохена и самоорганизующихся карт.

Задание: используя среду разработки VisualStudio решить выбранную задачу кластеризации.

Аппроксимация функции одной переменной с помощью аппарата нечеткой логики

Цель работы: научится использовать рекуррентные нейронные сети Хопфилда и Хэмминга.

Задание: используя среду разработки VisualStudio с помощью рекуррентных нейронных сетей Хопфилда или Хэмминга решить задачу ассоциативной памяти.

Построение нечеткой экспертной системы

Цель работы: освоить методику построения нечеткой экспертной системы.

Задание: построить элементарную нечеткую экспертную систему используя среду разработки VisualStudio.

Кластеризация с помощью алгоритма нечетких центров

Цель работы: освоить методику нахождения нечетких центров.

Задание: найти центры кластеров, используя алгоритм нечетких центров с помощью среды разработки VisualStudio.

Лабораторная работа 2

1. Используя среду разработки VisualStudio необходимо построить и обучить нейронную сеть для аппроксимации таблично заданной функции $y_i= f(x_i)$ $x_i= i*0.1$, i =1, 2, 3,...,20. Разработать программу, которая реализует нейросетевой алгоритм аппроксимации и выводит результаты аппроксимации в виде графиков. Значение $y_i=y_i(x_i)$ представлено в таблице 1.

Таблица 1 –Исходные данные

i	1	2	3	4	5	6	7	8	9	10
y_i	2.05	1.94	1.92	1.87	1.77	1.88	1.71	1.60	1.56	1.40
i	11	12	13	14	15	16	17	18	19	20
y_i	1.50	1.26	0.99	0.97	0.91	0.71	0.43	0.54	0.19	0.01

2. Используя среду разработки VisualStudio рассмотреть использование рекуррентной нейронной сети Хопфилда на примере решения задачи ассоциативной памяти.

Лабораторная работа 3

Алгоритм kNN или Метод k-ближайших соседей используется для решения задачи классификации. Он относит объекты к классу, которому принадлежит большинство из k его ближайших соседей в многомерном пространстве признаков. Это один из простейших алгоритмов обучения классификационных моделей.

Число k — это количество соседних объектов в пространстве признаков, которые сравниваются с классифицируемым объектом. Иными словами, если k=10, то каждый объект сравнивается с 10-ю соседями. Метод широко применяется в технологиях Data Mining.

В процессе обучения алгоритм просто запоминает все векторы признаков и соответствующие им метки классов. При работе с реальными данными, т.е. наблюдениями, метки класса которых неизвестны, вычисляется расстояние между вектором нового наблюдения и ранее запомненными. Затем выбирается k ближайших к нему векторов, и новый объект относится к классу, которому принадлежит большинство из них.

Выбор параметра k противоречив. С одной стороны, увеличение его значения повышает достоверность классификации, но при этом границы между классами становятся менее четкими. На практике хорошие результаты дают эвристические методы выбора параметра k, например, перекрестная проверка.

Несмотря на свою относительную алгоритмическую простоту, метод показывает хорошие результаты. Главным его недостатком является высокая вычислительная трудоемкость, которая увеличивается квадратично с ростом числа обучающих примеров.

На первом шаге алгоритма задается число k – количество ближайших соседей. Следует задавать значение k > 1, так как при k = 1 алгоритм теряет обобщающую способность. Также не следует задавать значение k слишком большим, потому что в таком случае не будут выявлены многие локальные особенности.

Затем находим k записей с минимальным расстоянием до вектора признаком нового объекта, то есть ищем k соседей.

Для упорядоченных значений атрибутов вычисляется Евклидово расстояние по формуле (1).

$$D_E = \sqrt{\sum_{i}^{n} (x_i - y_i)^2} \quad (1)$$

В данной формуле приняты следующие обозначения:

n – количество параметров;

x_i, y_i … - обозначение параметров объекта.

Далее рассчитывается вероятность принадлежности неизвестной точки к классу путем деления: определяется сколько соседей принадлежит к конкретному классу и делится на k.

При вычислении вероятностей может произойти так, что несколько классов будут иметь равную вероятность. В такой ситуации учитывается также и расстояние до новой записи. Чем меньше расстояние, тем более значимый вклад вносит голос. Голоса за класс находятся по формуле (2).

$$votes(class) = \sum_{i}^{n} \frac{1}{d^2(x_i, y_i)} \quad (2)$$

В формуле (2):

$d^2(x_i, y_i)$ – квадрат расстояния от известной записи из обучающих заданий до неизвестной записи;

Задание:

Пусть система интеллектуальной защиты информации зависит от двух параметров. Реализовать классификацию красных и синих точек (разделение по параметрам)

Для выполнения классификации kNN напишем программу –приложение Windows Forms на языке C#, генерирующее тестовые данные и классифицирующее объект с учетом k ближайших значений. Значение k вводится пользователем

Результатом работы программы является строка определяющая принадлежность объекта к определенному классу.

Рассмотрим принцип работы программы. На рисунке 1 показан внешний вид программы после запуска.

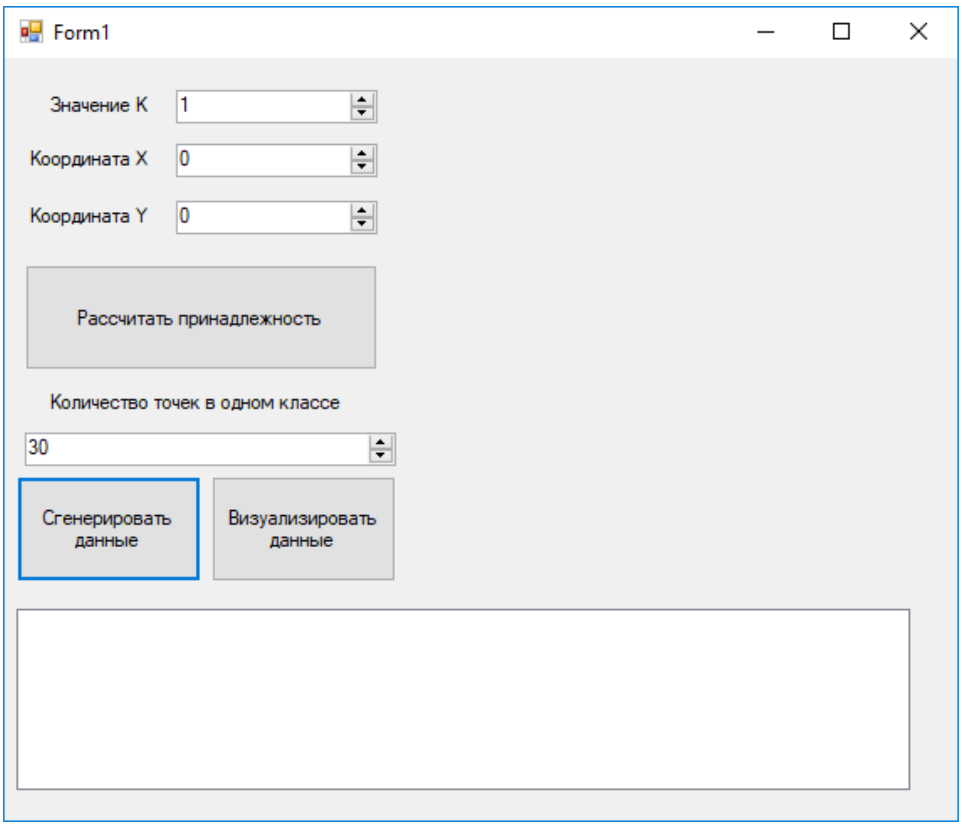


Рисунок 1 – Внешний вид программы

При нажатии на кнопку «Сгенерировать данные» создаются 2 текстовых файла с данными, содержащими координаты точек на плоскости, принадлежащий к первому или второму классу (рисунки 2-3).

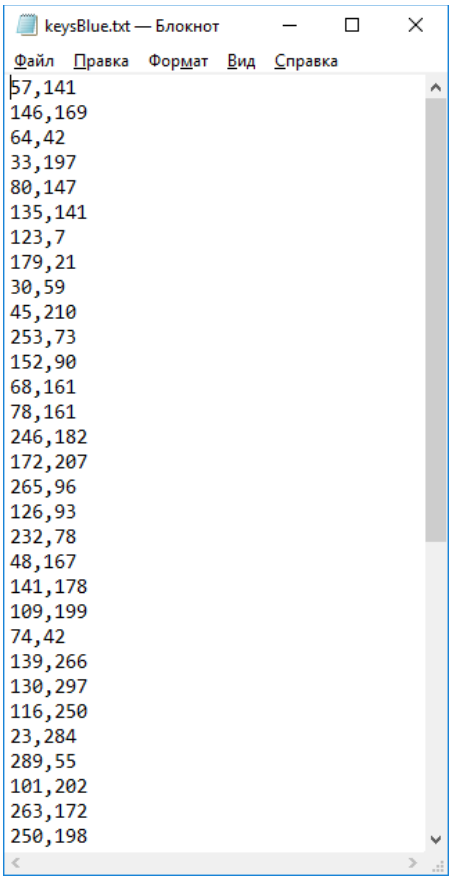


Рисунок 2 – Текстовый файл для синих точек

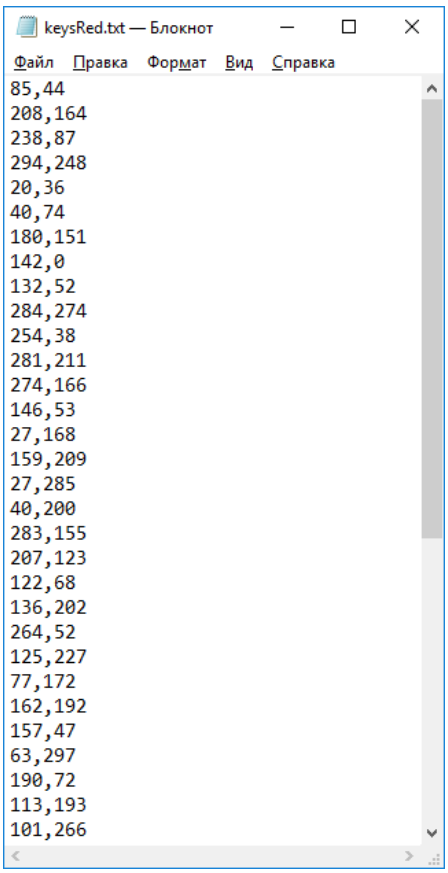


Рисунок 3 – Текстовый файл для красных точек

При нажатии на кнопку «Визуализировать данные» появляется картинка, на которой координаты вышеупомянутых точек окрашены в синий либо красный цвета в зависимости от класса, к которому они принадлежат (рис. 4)

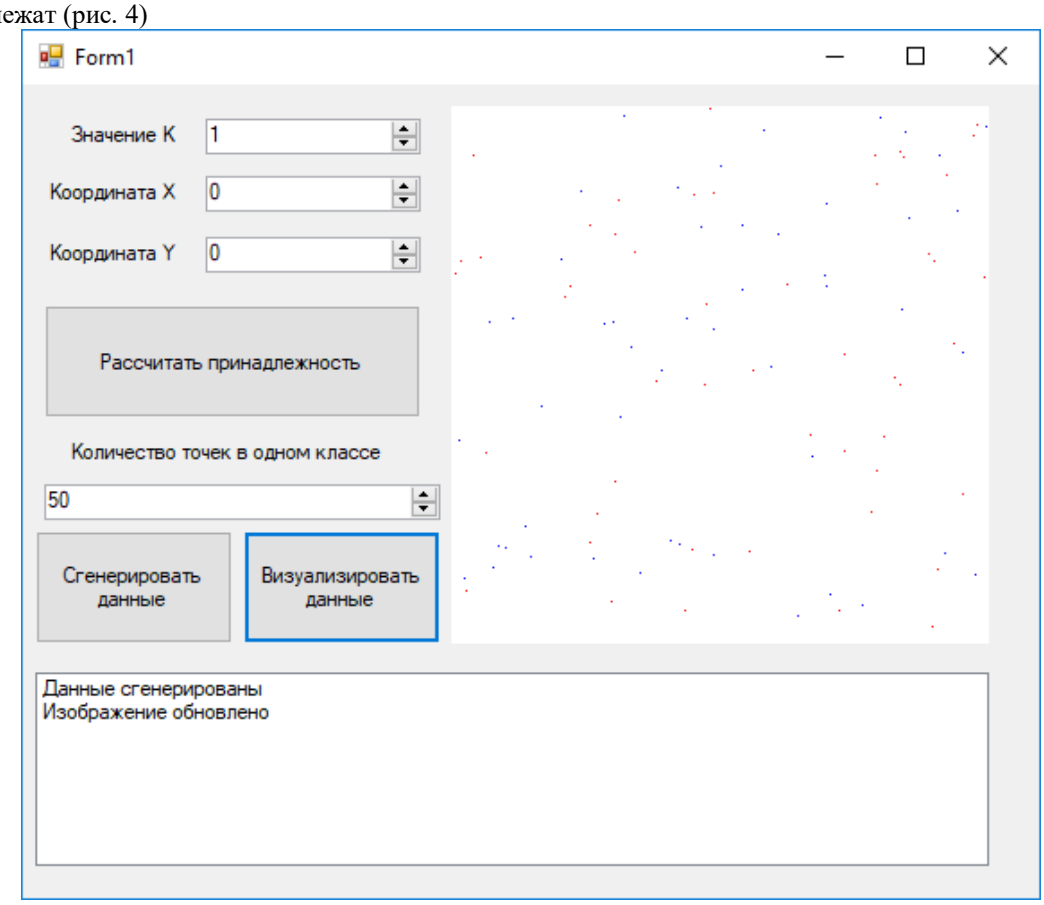


Рисунок 4 – Визуализация данных

Далее пользователь вводит координаты произвольной точки и указывает значение k. При нажатии на кнопку «Рассчитать принадлежность» программа, используя метод kNN, определяет к какому классу цветов относится объект и выдает строку с результатом (рисунок 5). При увеличении значения k возможно изменение принадлежности объекта к определенному классу ввиду более точных расчетов (рисунок 6).

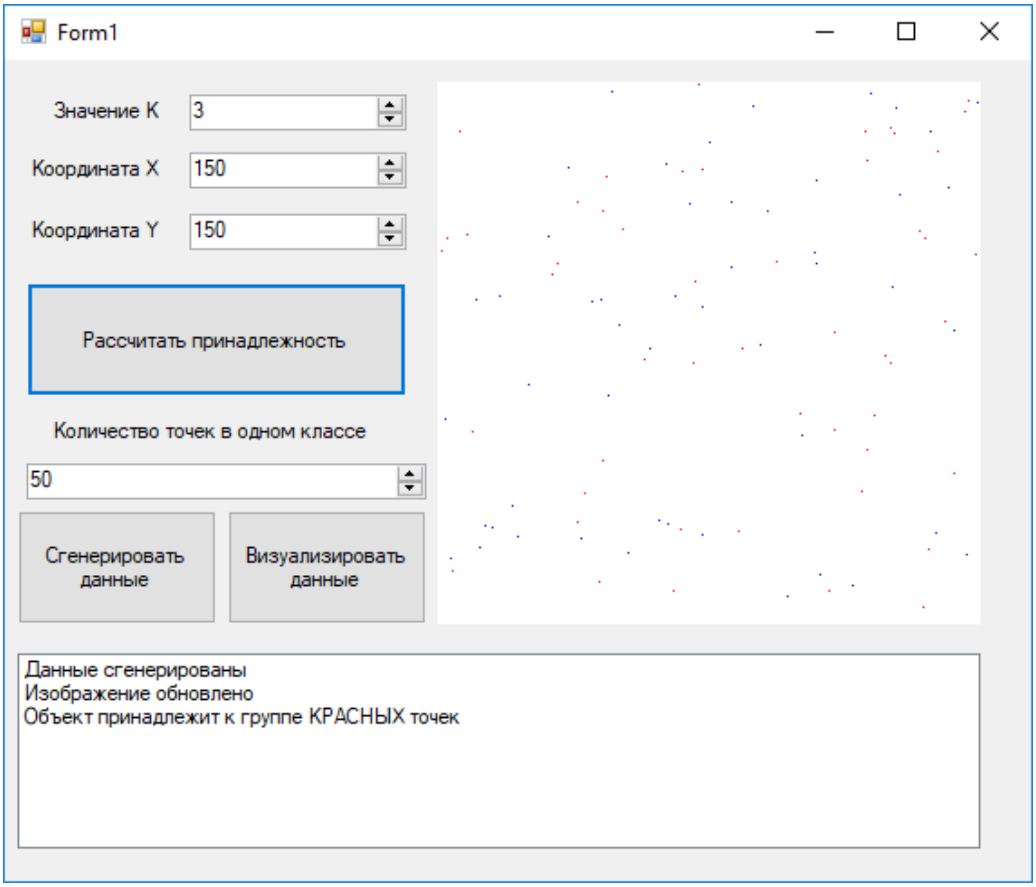


Рисунок 5 – Классификация точки методом kNN (k=3)

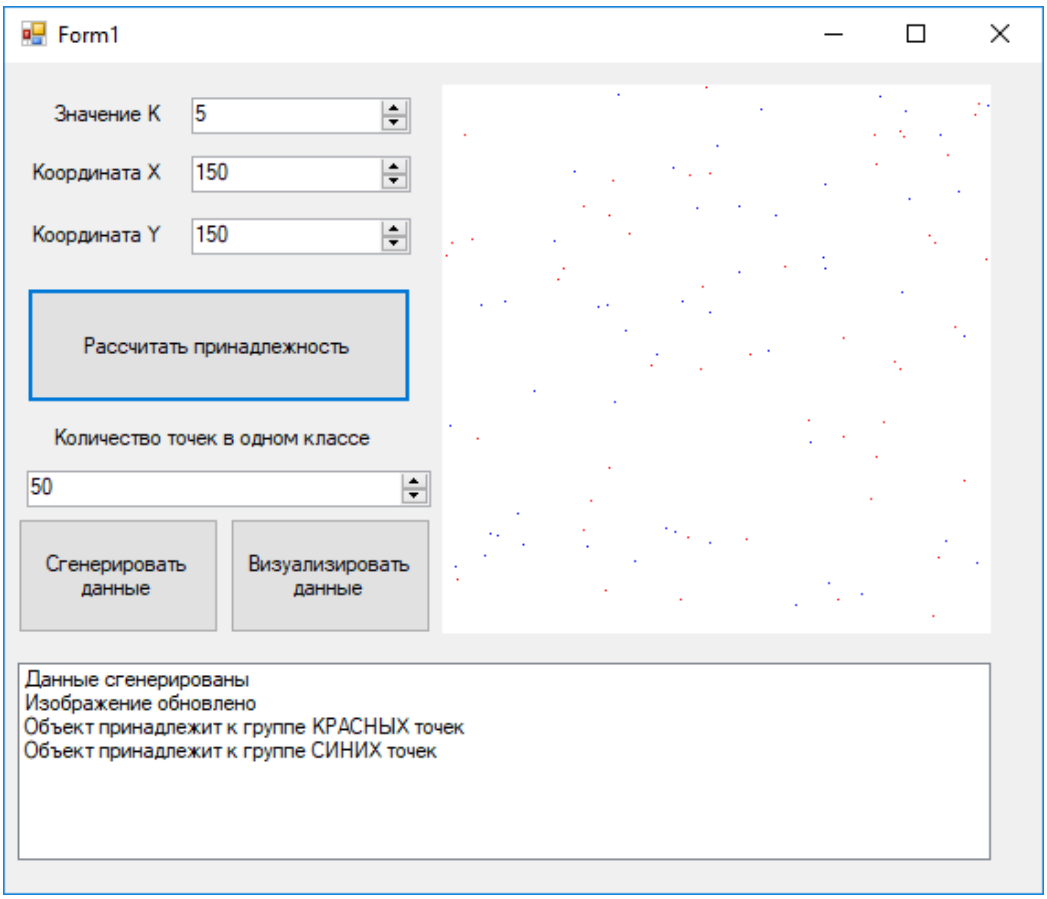


Рисунок 6 – Классификация точки методом kNN (k=5)

Примеры с другими точками и данными приведены на рис. 7-9.

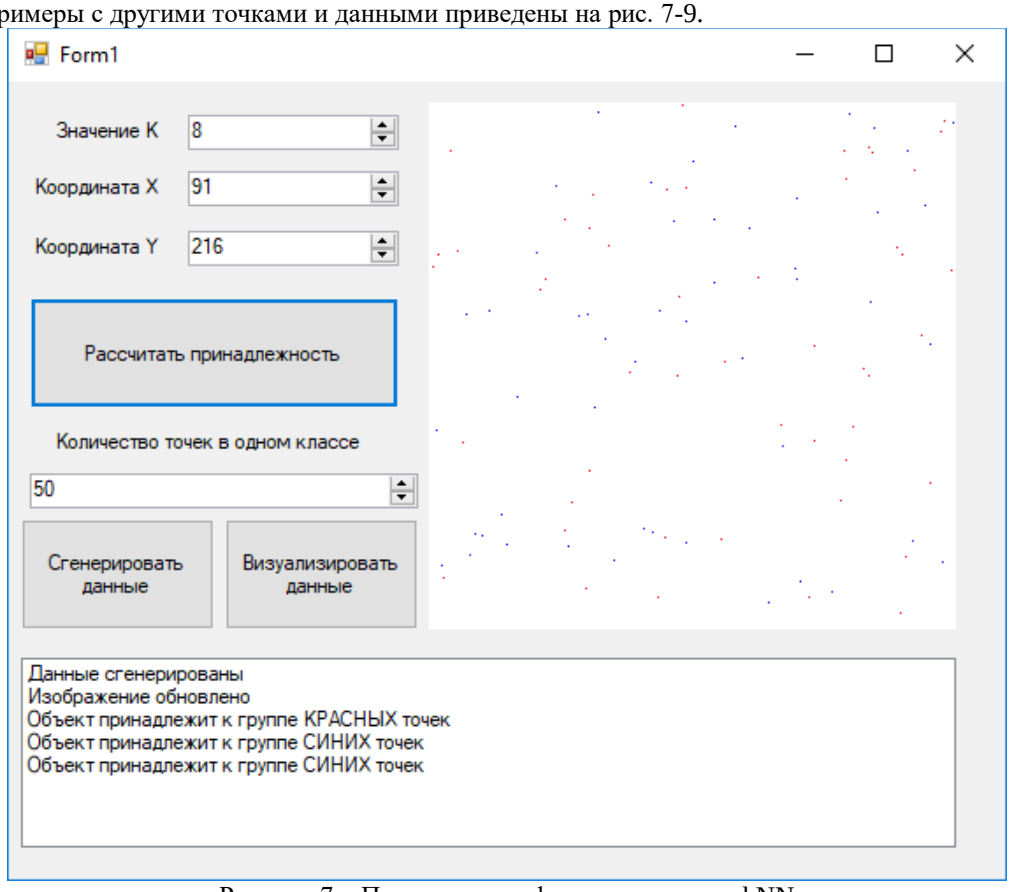


Рисунок 7 – Пример классификации методом kNN

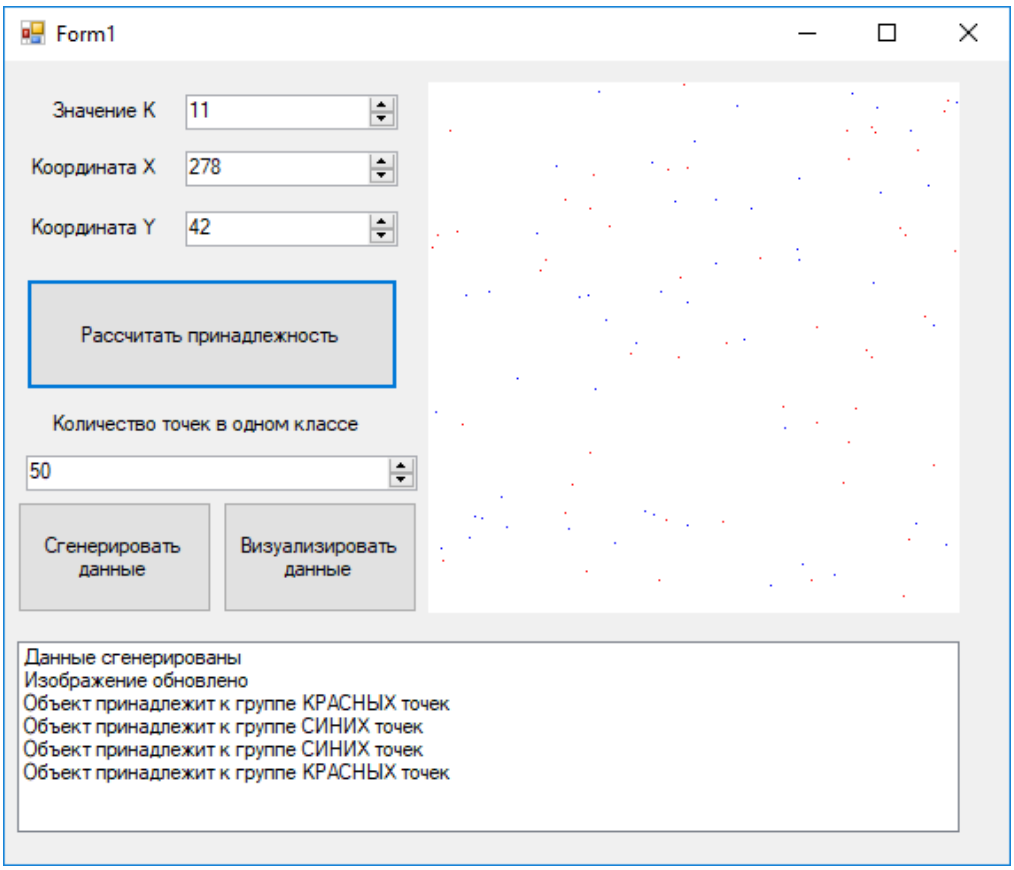


Рисунок 8 – Пример классификации методом kNN

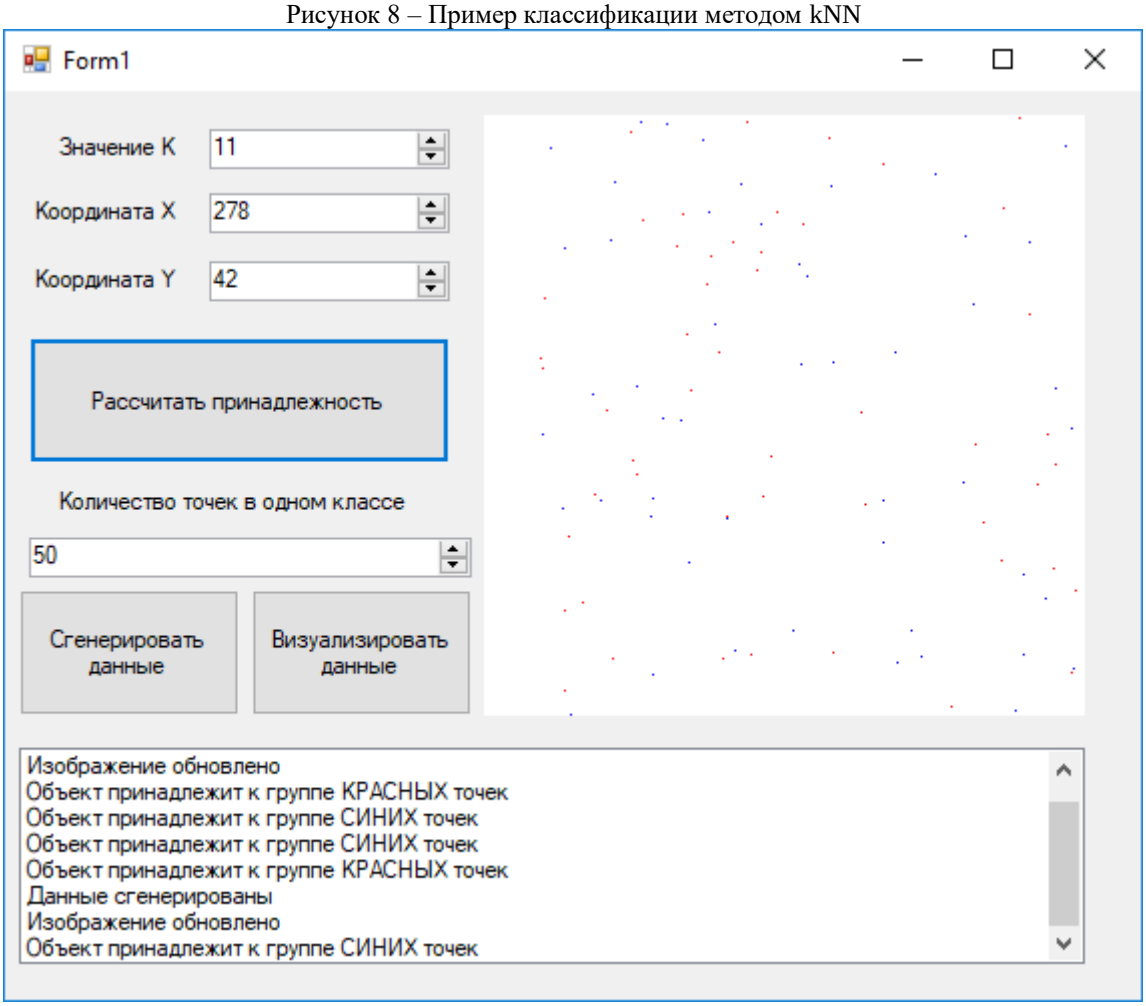


Рисунок 9 – Пример классификации методом kNN

Пример исходного текста на C# разработанного приложения

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Algoritm_kNN
{
public partial class Form1 : Form
{
public int numberOfPoints = 30;
public Form1()
{
InitializeComponent();
}

private void label3_Click(object sender, EventArgs e)
{

}

private void generateButton_Click(object sender, EventArgs e)
{
Random rand = new Random();
int N1 = 300;
int N2 = 300; //размер плоскости
numberOfPoints = Convert.ToInt32(numericUpDown4.Value); //количество генерируемых значений
for (int m = 0; m < 2; m++) //выполняется дважды для каждого класса значений
{
int[] key1 = new int[numberOfPoints];
int[] key2 = new int[numberOfPoints];
bool check = true;
for (int i = 0; i < numberOfPoints; i++) //генерация значений
{
check = true;
int temp1 = rand.Next(0, N1);
int temp2 = rand.Next(0, N2);
for (int j = 0; j < key1.Length; j++) if (Math.Abs(temp1 - key1[j]) <= 1)
for (int k = 0; k < key2.Length; k++) if (Math.Abs(temp1 - key2[k]) <= 1) { check = false; break; }

if (check)
{
key1[i] = temp1;
key2[i] = temp2;
}
else i--;
}
if (m == 0)
{
using (StreamWriter sw = new StreamWriter(@"D:\fff\keysBlue.txt", false, System.Text.Encoding.Default))
{
for (int i = 0; i < numberOfPoints; i++) //сохранение значений класса СИНИЙ
{
string temp = key1[i] + "," + key2[i];
sw.WriteLine(temp);
}
}
}
```

```
else
{
using (StreamWriter sw = new StreamWriter(@"D:\fff\keysRed.txt", false, System.Text.Encoding.Default))
{
for (int i = 0; i < numberOfPoints; i++) //сохранение значений класса КРАСНЫЙ
{
string temp = key1[i] + "," + key2[i];
sw.WriteLine(temp);
}
}
}
}
StatusListBox.Items.Add("Данные сгенерированы\n");
}

private void numericUpDown4_ValueChanged(object sender, EventArgs e)
{

}

private void redrawButton_Click(object sender, EventArgs e)
{
string path;
Bitmap image1 = new Bitmap(@"D:\fff\1.png", true); //загрузка изображения
for (int m = 0; m < 2; m++) //заполнение пустого изображения точками из сгенерированных файлов
сначала синего цвета, затем красного
{
if (m == 0) path = @"D:\fff\keysBlue.txt";
else path = @"D:\fff\keysRed.txt";
using (StreamReader sr = new StreamReader(path, System.Text.Encoding.Default))
{
for (int i = 0; i < numberOfPoints; i++)
{
string temp = sr.ReadLine();
string[] coordinates = temp.Split(',');
int x = Convert.ToInt32(coordinates[0]);
int y = Convert.ToInt32(coordinates[1]);
Color bs = image1.GetPixel(x, y);
if (image1.GetPixel(x, y) == Color.FromArgb(255, 255, 255, 255))
{
if (m == 0) image1.SetPixel(x, y, Color.Blue);
else image1.SetPixel(x, y, Color.Red);
}

}
}
}
image1.Save(@"D:\fff\new.png");
using (var file = new FileStream(@"D:\fff\new.png", FileMode.Open, FileAccess.Read,
FileShare.Inheritable))
{
pictureBox1.Image = Image.FromStream(file); //обновление картинки

}
StatusListBox.Items.Add("Изображение обновлено\n");

}

private void pictureBox1_Click(object sender, EventArgs e)
{

}
```

```
        private void pictureBox1_Click_1(object sender, EventArgs e)
        {

        }

        private void calculateButton_Click(object sender, EventArgs e)
        {
        double[] distanceArrayBlue = new double[numberOfPoints];
        double[] distanceArrayRed = new double[numberOfPoints]; //инициализация массивов для хранения
        рассчитанных расстояний
        using (StreamReader sr = new StreamReader(@"D:\fff\keysBlue.txt", System.Text.Encoding.Default))
        {
        for (int i = 0; i < numberOfPoints; i++) //расчет расстояния от точки до всех объектов принадлежащих
        классу СИНИЙ
        {
        string temp = sr.ReadLine();
        string[] coordinates = temp.Split(',');
        int x = Convert.ToInt32(coordinates[0]);
        int y = Convert.ToInt32(coordinates[1]);
        distanceArrayBlue[i] = Math.Sqrt(Math.Pow(Convert.ToInt32(numericUpDownX.Value) - x, 2) +
        Math.Pow(Convert.ToInt32(numericUpDownY.Value) - y, 2));

        }
        double tmp;
        for (int i=0; i< numberOfPoints;i++)
        {
        for (int j= i+1; j < numberOfPoints; j++) //сортировка массива в порядке возрастания значений
        { (изм.)
        •
if (distanceArrayBlue[i] > distanceArrayBlue[j])
{
tmp = distanceArrayBlue[i];
distanceArrayBlue[i] = distanceArrayBlue[j];
distanceArrayBlue[j] = tmp;
}
}
}
}
using (StreamReader sr = new StreamReader(@"D:\fff\keysRed.txt", System.Text.Encoding.Default))
{
for (int i = 0; i < numberOfPoints; i++) //расчет расстояния от точки до всех объектов принадлежащих классу
КРАСНЫЙ
{
string temp = sr.ReadLine();
string[] coordinates = temp.Split(',');
int x = Convert.ToInt32(coordinates[0]);
int y = Convert.ToInt32(coordinates[1]);
distanceArrayRed[i] = Math.Sqrt(Math.Pow(Convert.ToInt32(numericUpDownX.Value) - x, 2) +
Math.Pow(Convert.ToInt32(numericUpDownY.Value) - y, 2));
}
double tmp;
for (int i = 0; i < numberOfPoints; i++)
{
for (int j = i + 1; j < numberOfPoints; j++) //сортировка массива в порядке возрастания значений
{
if (distanceArrayRed[i] > distanceArrayRed[j])
{
tmp = distanceArrayRed[i];
distanceArrayRed[i] = distanceArrayRed[j];
distanceArrayRed[j] = tmp;
}
}
}
```

```
}
int K = Convert.ToInt32(numericUpDownK.Value), m = 0, n = 0;
if (distanceArrayBlue[0] == 0 || distanceArrayRed[0] == 0) StatusListBox.Items.Add("Координаты точки совпали с шаблоном\n"); //проверка на совпадение с уже отмеченными объектами
else
{
while (K != 0) //перебор k ближайших соседей точки сравнением расстояний объектов разных классов до неё
{
if (K == 1 && distanceArrayBlue[m] == distanceArrayRed[n]) break;
else if (distanceArrayBlue[m] > distanceArrayRed[n]) n++;
else if (distanceArrayBlue[m] <= distanceArrayRed[n]) m++;
K--;
}
if (m > n) StatusListBox.Items.Add("Объект принадлежит к группе СИНИХ точек\n");
else if (m < n) StatusListBox.Items.Add("Объект принадлежит к группе КРАСНЫХ точек\n");
else
{
double summBlue = 0, summRed = 0;
for (int i = 0; i < m; i++) summBlue += distanceArrayBlue[i];
for (int i = 0; i < m; i++) summRed += distanceArrayRed[i]; //при равенстве количества ближайших объектов расчет наименьшей суммы всех расстояний объектов каждого класса до точки
if (summRed > summBlue) StatusListBox.Items.Add("Объект принадлежит к группе СИНИХ точек.\n");
else if (summRed < summBlue) StatusListBox.Items.Add("Объект принадлежит к группе КРАСНЫХ точек.\n");
else StatusListBox.Items.Add("Не удалось определить принадлежность данной точки. Измените значение K.\n");
}
}
}
}
}
```

Лабораторная работа 4

Считая что на вход пропускной системы подается массив различных точек(для системы распознавания) реализовать алгоритм k-means для кластеризации.

Алгоритм K-means относится к алгоритмам обучения без учителя. На вход алгоритма подаётся набор точек произвольной размерности. Задача алгоритма – разделить их на заранее известное число кластеров.

Шаги алгоритма:

1) выбирается число кластеров K;
2) выбирается K случайных точек, которые будут служить начальными центрами кластеров;
3) для каждой точки определяется ближайшая центроида, при этом точки, «притянутые» к одной точке из 2-го шага, образуют кластер;
4) вычисляются центроиды – центры тяжести кластеров. Каждый центроид – это вектор, элементы которого представляют собой средние значения соответствующих признаков, вычисленные по всем записям кластера;
5) центр кластера смещается в его центроид, после чего центроид становится новым центром кластера;
6) если на 5 шаге произошло смещение, то алгоритм повторяется с третьего шага. Если же нет, то алгоритм завершил свою работу.

Преимуществом данного алгоритма являются его простота реализации и скорость работы. Недостатком – неопределённость выбора начальных центров кластеров (на шаге 2) и то, что число кластеров должно быть известно заранее.

Описание работы

Реализация алгоритма K-means осуществлена на языке программирования Java с использованием библиотеки графического интерфейса JavaFX.

Исходный код программы состоит из нескольких частей:

1) Main.java – содержит точку входа (приложение А);
2) Controller.java – содержит логику графического интерфейса (приложение Б);
3) kmeans.fxml – содержит разметку графического интерфейса (приложение В);
4) clustering/KmeansAlgorithm.java – содержит код реализации алгоритма K-means (приложение Г);

5) clustering/Point.java – содержит класс многомерной точки, используется в реализации алгоритма K-means (приложение Д);

6) exampleData.txt – пример данных для обучения (приложение Е).

При запуске скомпилированного и собранного проекта откроется окно, изображённое на рисунке 2.1.

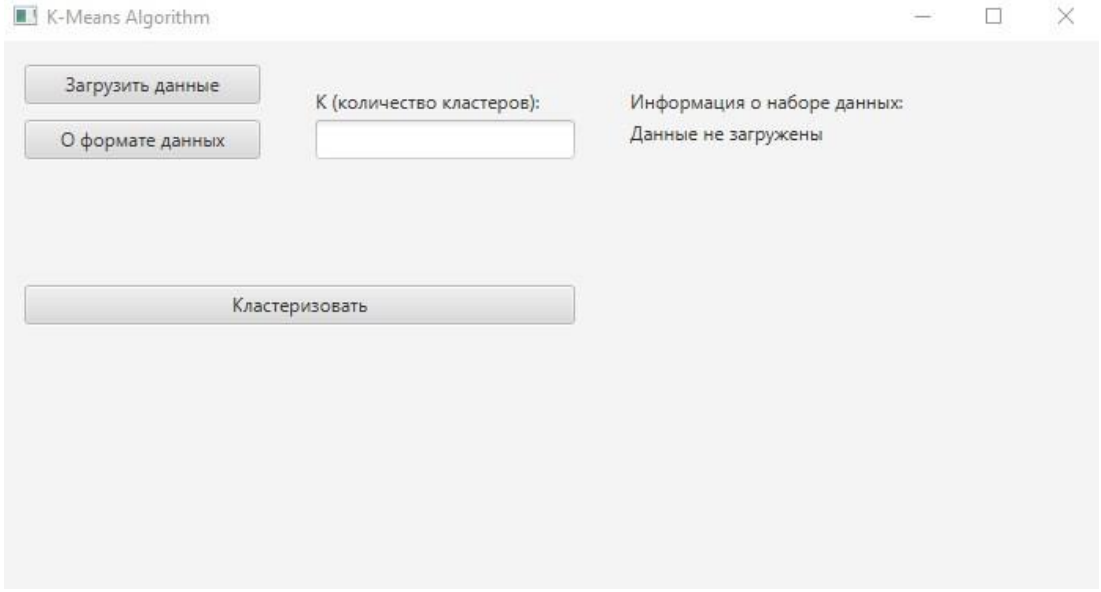


Рисунок 2.1 – Главное окно программы

Далее необходимо ознакомиться с форматом входных данных. Информация об этом доступна по нажатию кнопки «О формате данных». При этом откроется окно, изображённое на рисунке 2.2.

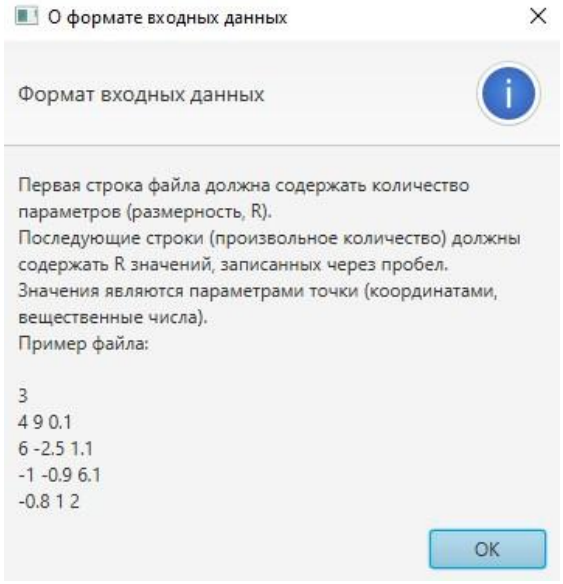


Рисунок 2.2 – Формат входных данных

После ознакомления с форматом входных данных необходимо создать файл, содержащий входные данные (данные для обучения). Пример такого файла приведён в приложении Е.
Далее необходимо загрузить данные. Это производится путём нажатия на кнопку «Загрузить данные». Откроется окно, где требуется выбрать файл входных данных. При невозможности открыть и прочитать файл, пользователь увидит сообщение об ошибке, пример которого изображён на рисунке 2.3.

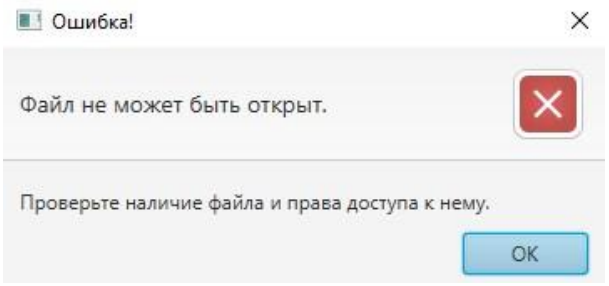


Рисунок 2.3 – Ошибка открытия файла

Если же файл был успешно открыт и прочитан, однако формат данных в файле не соответствует требуемому, пользователь в правой части программы увидит сообщение, представленную на рисунке 2.4.

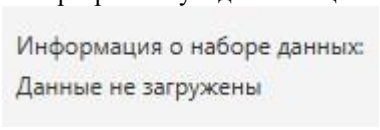


Рисунок 2.4 – Несоответствие формата данных

При успешной загрузке данных в правой части программы будет отображена информация о наборе данных. Её пример приведён на рисунке 2.5.

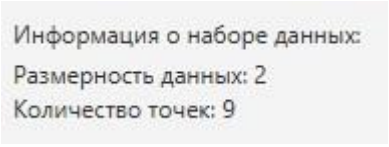


Рисунок 2.5 – Информация о наборе данных

Далее необходимо указать значение K – количество кластеров. Это производится в соответствующем поле вверху окна. При указании некорректного значения K пользователь увидит сообщение об ошибке, пример которого изображён на рисунке 2.6.

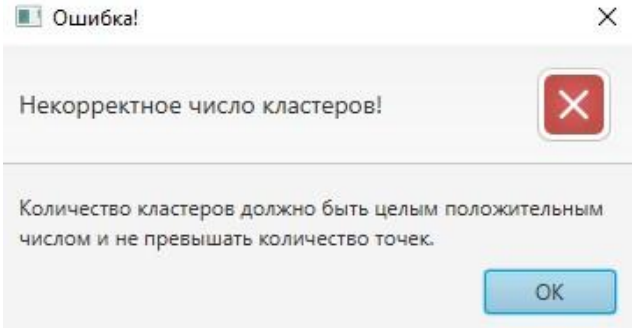


Рисунок 2.6 – Некорректное значение K

После ввода всех необходимых параметров при нажатии кнопки «Кластеризовать» будет запущен метод predictClusters из KmeansAlgorithm.java.
После окончания работы алгоритма будет создан файл output.txt в папке с программой, в котором содержится информация о центрах кластеров и список точек с указанием кластера. Пример файла изображён на рисунке 2.7.

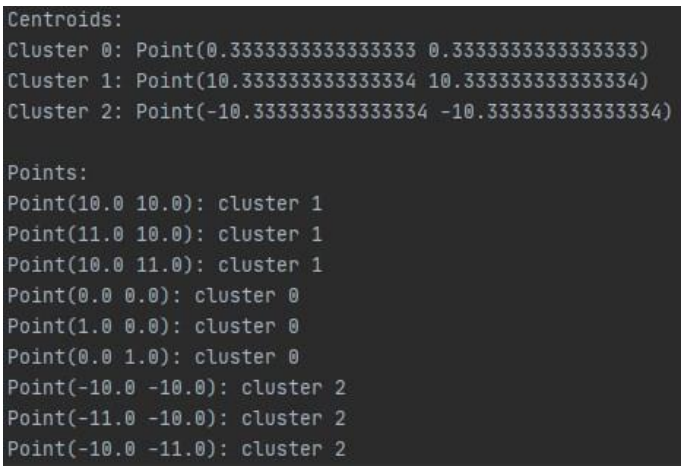
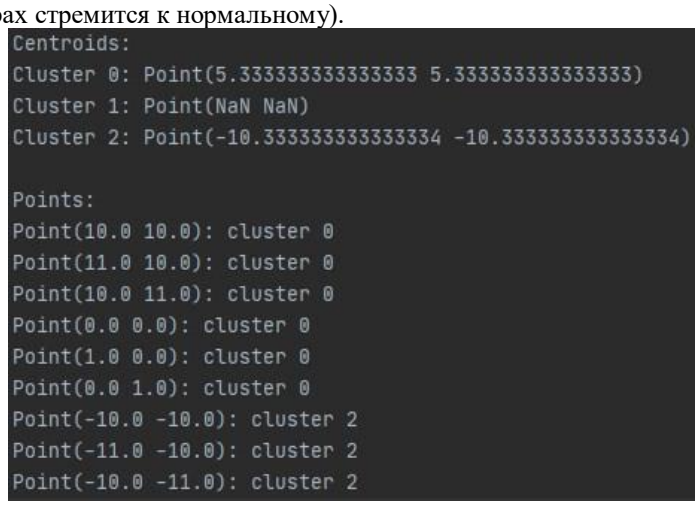


Рисунок 2.7 – Результат работы программы

В некоторых случаях центр кластера может иметь координаты NaN. Это возникает в тех случаях, когда на втором шаге алгоритма точки сгенерировались слишком близко друг к другу, из-за чего одна точка «забирает на себя» точки из выборки, а другая сгенерированная точка перестаёт участвовать в алгоритме из-за того, что к ней не привязана ни одна точка из выборки. Такой случай изображён на рисунке 2.8. Возможность возникновения таких случаев является недостатком алгоритма, поэтому иногда для кластеризации применяют другие алгоритмы, похожие на K-means, например, g-means (gaussianmeans, распределение в кластерах стремится к нормальному).



Пример разработанного приложения на java.

Main.java

```
package com.sbelousov.kmeans;

import javafx.application.Application; import ja-
vafx.fxml.FXMLLoader; import javafx.scene.Parent; import ja-
vafx.scene.Scene; import javafx.stage.Stage;

public class Main extends Application {

   @Override
   public void start(Stage primaryStage) throws Exception{
      Parent root = FXML-
Loader.load(getClass().getResource("kmeans.fxml"));        primaryStage.setTitle("K-Means Algorithm");
primaryStage.setScene(new Scene(root, 700, 350));        primaryStage.show();
   }

   public static void main(String[] args) {        launch(args);
   }
}
```

Controller.java

```
package com.sbelousov.kmeans;

import com.sbelousov.kmeans.clustering.KmeansAlgorithm; import javafx.fxml.FXML; import
javafx.scene.control.Alert; import javafx.scene.control.Label; import
javafx.scene.control.TextField; import javafx.scene.layout.VBox; import
javafx.stage.FileChooser;

import java.io.File;

public class Controller {

   public KmeansAlgorithm kmeansAlgorithm = new KmeansAlgorithm();

   @FXML public VBox vbox;
   @FXML public Label dataInfoText;
   @FXML public TextField kField;

   private void showAlert(String title, String headerText,
String contentText, Alert.AlertType type) {         Alert alert = new Alert(type);
alert.setTitle(title);         alert.setHeaderText(headerText);
alert.setContentText(contentText);         alert.showAndWait();
   }
   public void onLoadDataButtonClick() {
      FileChooser fileChooser = new FileChooser();
      File            file        =           fileChooser.showOpenDialog(vbox.getScene().getWindow());         if (file ==
null) {            showAlert("Ошибка!",
                "Файл не может быть открыт.",
                "Проверьте наличие файла и права доступа к нему.",
                Alert.AlertType.ERROR);            return;
      }

      try {
         kmeansAlgorithm.parseFile(file);         }
```

```
        catch (NumberFormatException e) {            showAlert("Ошибка!",
                "Некорректный формат файла.",
                "Проверьте соблюдение формата входных данных.",
                Alert.AlertType.ERROR);            return;
        }

        dataInfoText.setText("Размерность данных: " + kmeansAlgorithm.getDimensionNumber() + "\n" +
                "Количество точек: " + kmeansAlgorithm.getSize()
+ "\n");
    }
    public void onAboutDataFormatButtonClick() {        showAlert("О формате входных данных",
            "Формат входных данных",
            "Первая строка файла должна содержать количество параметров (размерность, R).\n" +
                    "Последующие строки (произвольное количество) должны содержать R значений, " +
                    "записанных через пробел.\n" +
                    "Значения являются параметрами точки
(координатами, вещественные числа).\n" +
                    "Пример файла:\n\n" +
                    "3\n" +
                    "4 9 0.1\n" +
                    "6 -2.5 1.1\n" +
                    "-1 -0.9 6.1\n" +
                    "-0.8 1 2",
            Alert.AlertType.INFORMATION);
    }
    public void onComputeButtonClick() {
        // Set number of clusters        try {
            kmeansAlgorithm.setClusters(kField.getText());
        }
        catch (NumberFormatException e) {            showAlert("Ошибка!",
                "Некорректное число кластеров!",
                "Количество кластеров должно быть целым положительным числом " +
                        "и не превышать количество точек.",
                Alert.AlertType.ERROR);            return;
        }

        // Clustering
        kmeansAlgorithm.predictClusters();

        // Output
```

```
        return;
    }
}
```

kmeans.fxml

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.control.TextField?>
<?import javafx.scene.layout.AnchorPane?>
<?import javafx.scene.layout.VBox?>

<VBox prefHeight="700.0" prefWidth="350.0" xmlns="http://javafx.com/javafx/11.0.1"
xmlns:fx="http://javafx.com/fxml/1"       fx:controller="com.sbelousov.kmeans.Controller" fx:id="vbox">
   <AnchorPane VBox.vgrow="ALWAYS">
      <Button layoutX="15.0" layoutY="15.0" mnemonicParsing="false" prefHeight="25.0" prefWidth="150.0"
            text="Загрузить данные" onAction="#onLoadDataButtonClick"/>
      <Button layoutX="15.0" layoutY="50.0" mnemonicParsing="false" prefHeight="25.0" prefWidth="150.0"
text="О формате данных" onAction="#onAboutData-
FormatButtonClick"/>

      <Button layoutX="15.0" layoutY="155.0" mnemonicParsing="false" prefHeight="25.0" prefWidth="350.0"
text="Кластеризовать" onAction="#onComputeButtonClick"/>

      <Label        layoutX="200.0"              layoutY="30.0"   text="K (количество кластеров):"/>
      <TextField fx:id="kField" layoutX="200.0" layoutY="50.0" prefHeight="25.0" prefWidth="165.0"/>

      <Label layoutX="400.0" layoutY="30.0" text="Информация о наборе данных:"/>
      <Label fx:id="dataInfoText" layoutX="400.0" layoutY="50.0" text="Данные не загружены"/>
   </AnchorPane>
</VBox>
```

KmeansAlgorithm.java

```
package com.sbelousov.kmeans.clustering;

import java.io.File; import java.io.IOException;
import java.nio.file.Files; import ja-
va.nio.file.Path; import java.nio.file.Paths; im-
port java.util.*;
import java.util.stream.Collectors;

public class KmeansAlgorithm {     private int clusters = 2;     pri-
vate int dimensionNumber = 0;
   private final List<Point> points = new ArrayList<>();     private List<Point> centroids = new ArrayList<>();

   public void setClusters(int clusters) {
      if (clusters <= 0 || clusters > this.getSize()) {            throw new NumberFormatException("Incorrect number
K");        }
      this.clusters = clusters;
   }
   public void setClusters(String clusters) {        this.setClusters(Integer.parseInt(clusters));
   }
   public int getSize() {        return points.size();
   }
   public int getDimensionNumber() {        return dimensionNumber;
   }
   public void parseFile(File file) {
      Path path = file.toPath();
      List<String> lines = new ArrayList<>();        try {
         lines = Files.readAllLines(path);
      }
      catch (IOException e) {
         e.printStackTrace();
      }
```

```
    dimensionNumber = Integer.parseInt(lines.get(0));       for (int i = 1; i < lines.size(); i++) {
      List<Double> coordinates = new ArrayList<>();          String[] splitted =
lines.get(i).split("\\s+");          for (int j = 0; j < dimensionNumber; j++) {              coordi-
nates.add(Double.parseDouble(splitted[j]));          }
      points.add(new Point(coordinates));
    }
  }
  public void writeToFile() {
    Path path = Paths.get("output.txt");
    StringBuilder stringBuilder = new StringBuilder();       try {
      stringBuilder.append("Centroids:\n");          for (int i = 0; i < centroids.size(); i++) {              string-
Builder.append("Cluster ").append(i).append(": ").append(centroids.get(i)).append("\n");
      }
                        stringBuilder.append("\nPoints:\n");          for (Point p : points) {              stringBuild-
                                                                                er.append(p).append(":    cluster
").append(p.getCluster()).append("\n");
      }
      Files.writeString(path, stringBuilder);
    }
    catch (IOException ex) {          ex.printStackTrace();
    }
  }
  private double getMinimalCoordinate() {
    List<Double> coordinates = new ArrayList<>();       for (Point p : points) {
      coordinates.addAll(p.getCoordinates());
    }
    return Collections.min(coordinates);
  }
  private double getMaximalCoordinate() {
    List<Double> coordinates = new ArrayList<>();       for (Point p : points) {
      coordinates.addAll(p.getCoordinates());
    }
    return Collections.max(coordinates);
  }
  public void predictClusters() {
    // Step 0: Determine min and max ranges for choosing random points in next step
    double rangeMin = getMinimalCoordinate();
```

```
    double rangeMax = getMaximalCoordinate();

    // Step 1: Select K random points as cluster centers called centroids
    centroids = new ArrayList<>();        for (int i = 0; i < clusters; i++) {
      List<Double> coordinates = new ArrayList<>();           for (int j = 0; j < dimensionNumber; j++) {
Random random = new Random();
          double coordinate = rangeMin + (rangeMax rangeMin) * random.nextDouble();
          coordinates.add(coordinate);
      }
      Point centroid = new Point(coordinates);           centroids.add(centroid);
    }

    boolean flag;       do {
      flag = false;

      System.out.println("Centroids:");           for (Point c : centroids) {
        System.out.println(c);
      }
      System.out.println();

      // Step 1.9: Save current centroids coordinates
      List<List<Double>> oldCoordinates = new ArrayList<>();
      for (int i = 0; i < centroids.size(); i++) {              oldCoordinates.add(new ArrayList<>());
for (int j = 0; j < dimensionNumber; j++) {                 oldCoordi-
nates.get(i).add(centroids.get(i).getCoordinates().get(j));
          }
      }

      // Step 2: Calculate distances from each point to each centroid
      Map<Point, List<Double>> distancesFromPoints = new HashMap<>();
      for (Point centroid : centroids) {              for (Point p : points) {
          if (!distancesFromPoints.containsKey(p)) {                    distancesFromPoints.put(p, new Ar-
rayList<>());
          }
          distancesFromPoints.get(p).add(p.getEuclideanDistance(centroid));
        }
      }
```

```
            // Step 2.5: Assign cluster number
            for (Map.Entry<Point, List<Double>> entry : distancesFromPoints.entrySet()) {
                Point point = entry.getKey();
                double minDistance = Collections.min(entry.getValue());
                int clusterNumber = entry.getValue().indexOf(minDistance);
                point.setCluster(clusterNumber);
            }

            // Step 3: Move centroids
            for (int i = 0; i < centroids.size(); i++) {                int finalI = i;
                List<Point> pointList = points.stream().filter(p
-> p.getCluster() == finalI)
                        .collect(Collectors.toList());
                List<Double> newCoordinates = new ArrayList<>();                for (int j = 0; j < dimensionNumber;
j++) {                        double sum = 0;
                    for (Point p : pointList) {
                        sum += p.getCoordinates().get(j);
                    }
                    newCoordinates.add(sum / pointList.size());
                }
                centroids.get(i).setCoordinates(newCoordinates);            }

            // Step 3.1: If centroids weren't moving that it's the end
            for (int i = 0; i < centroids.size(); i++) {
                if                                                                                (!centroids.get(i).getCoordi-
nates().equals(oldCoordinates.get(i))) {                    flag = true;
break;
                }
            }

        } while (flag);

        writeToFile();
    }
}
```

Point.java

```
package com.sbelousov.kmeans.clustering;

import java.util.ArrayList; import java.util.List;

public class Point {
   private List<Double> coordinates;     private int cluster;

   public List<Double> getCoordinates() {         return coordinates;
   }
   public void setCoordinates(List<Double> coordinates) {         this.coordinates = coordinates;
   }
   public int getCluster() {         return cluster;
   }
   public void setCluster(int cluster) {         this.cluster = cluster;
   }
   public Point(List<Double> coordinates) {         this.coordinates = coordinates;
   }
   public double getEuclideanDistance(Point point) {
      List<Double> items = new ArrayList<>();

      if (this.getCoordinates().size() != point.getCoordinates().size()) {
         throw new IllegalArgumentException("Dimensions of two points are unequal");
      }
      int dimensionsNumber = this.getCoordinates().size();

      for (int i = 0; i < dimensionsNumber; i++) {             double oneCoordinate = this.getCoordinates().get(i);
double anotherCoordinate = point.getCoordinates().get(i);             items.add(Math.pow(oneCoordinate - another-
Coordinate, 2));         }
```

```
        double sum = items.stream().mapToDouble(Double::doubleValue).sum();
        return Math.sqrt(sum);
    }
    public String toString() {
        StringBuilder builder = new StringBuilder();         builder.append("Point(");
        for (Double coordinate : coordinates) {              builder.append(coordinate).append(" ");
        }
        builder.deleteCharAt(builder.length() - 1);          builder.append(")");          return build-
er.toString();
    }
}
```

exampleData.txt

```
2
10 10
11 10
10 11
0 0
1 0
0 1
-10 -10
-11 -10
-10 -11
```

Темы / задания расчетно-графической работы

Считая что пропускная система распознает лица проходящих через нее людей Написать программу, которая будет находить лицо на изображении. Использовать библиотеку Dlib.

1 Обнаружение лиц на изображении

В данной работы будет использоваться dlib и OpenCV для обнаружения лицевых ориентиров на изображении, которые используются для локализации ключевых областей на лице человека:

- Глаза;
- Брови;
- Нос;
- Рот;
- Линия подбородка.

Выявление лицевых ориентиров состоит из двух этапов:

- Локализация лица на изображении;
- Обнаружение значимых областей на лицах в областях интереса.

Существует множество методов обнаружения лиц на изображении.

Нас интересует алгоритм, который определяет контуры лица, т.е. значения точек с координатами (x;y).

Определив на изображении область с лицом человека, можно переходить к шагу 2 – локализация отдельных частей лица.

В методе, который используется в библиотеке dlib применяется следующее:

- Тренировочные задания с маркированными лицевыми ориентирами на изображениях. Эти изображения промаркированы вручную точками (определяющимися координатами (x;y)), окружающими каждую ключевую область лица;
- Вероятности расстояний между парами пикселей на входных изображениях.

Данный обученный детектор из библиотеки dlib использует положение 68 точек, заданных координатами (x;y), соответствующих положению ключевых частей на лице человека.

Индексы точек приводятся на рисунке 1.

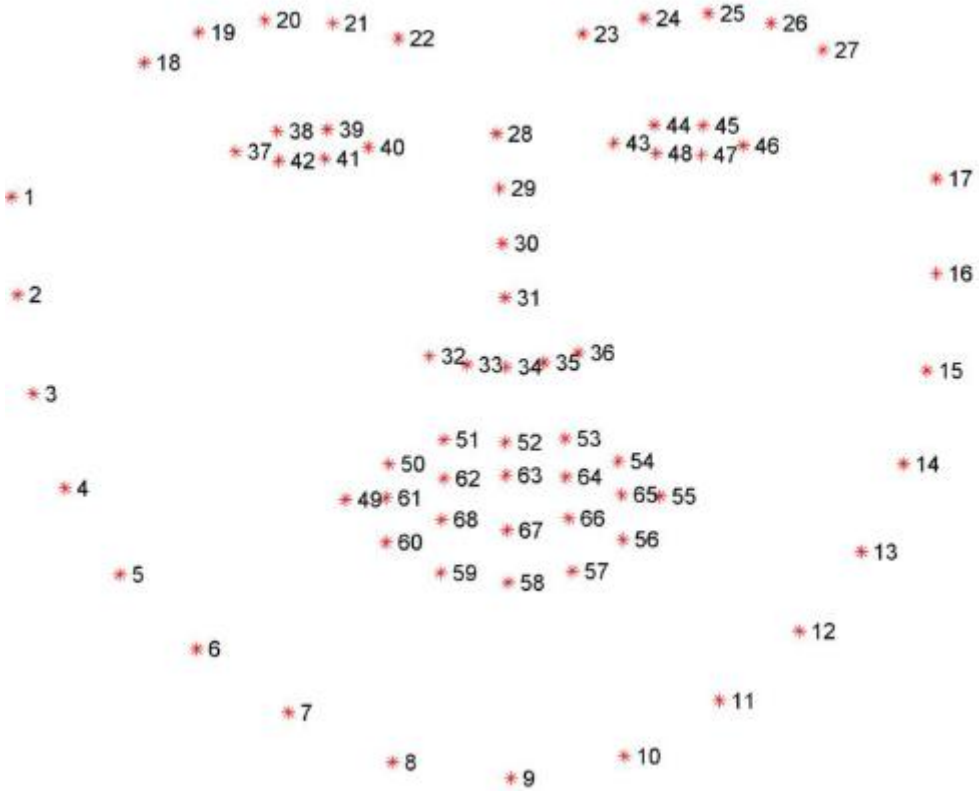


Рисунок 1 – Графическое представление 68 координат лицевых ориентиров

Поняв примерный принцип работы алгоритма, обнаруживающего ориентиры на лице человека, используя готовый обученный заранее детектор ключевых частей лица из библиотеки dlib, можем обнаруживать человеческие лица на изображениях при помощи скрипта на языке Python.

2 Результат работы

Выполним скрипт, который обнаруживает лица на изображениях и маркирует их (рисунок 2).

Контур лица выделяется зеленой рамкой, а части лица по отдельности помечаются красными точками.

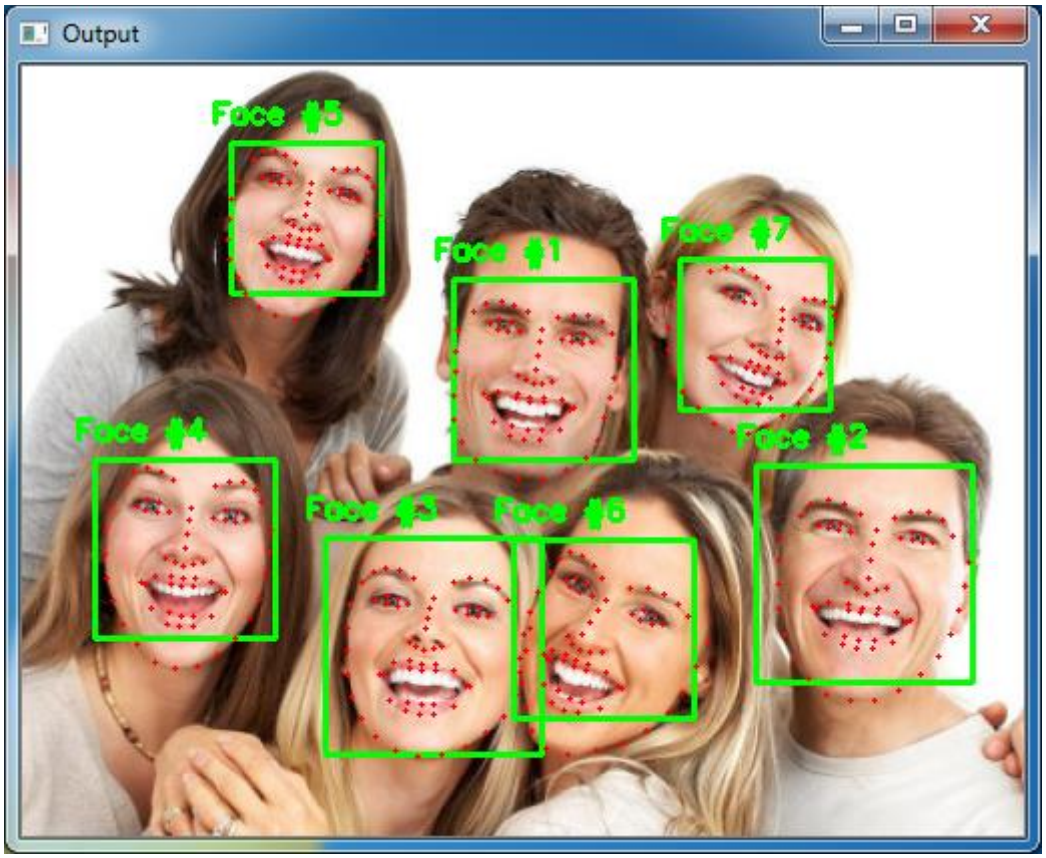


Рисунок 2 – Результат работы скрипта,
обнаруживающего лица на изображении

Как можно увидеть, что губы отмечены у людей, которые открыли рот. Также обнаружено лицо №6, повернутое вбок.

На рисунках 3-8 приводится результат работы скрипта, маркирующего различные части на лице по отдельности.

Для этого в скрипте явно указаны координаты точек, принадлежащих конкретным частям на лицах (листинг 1).

Листинг 1 – Координаты лицевых ориентиров в библиотеке face_utils

```
FACIAL_LANDMARKS_IDXS = OrderedDict([
("mouth", (48, 68)),
("right_eyebrow", (17, 22)),
("left_eyebrow", (22, 27)),
("right_eye", (36, 42)),
("left_eye", (42, 48)),
("nose", (27, 35)),
("jaw", (0, 17))
])
```

Рисунок 3 – Точки, описывающие рот

Рисунок 4 – Точки, описывающие губы

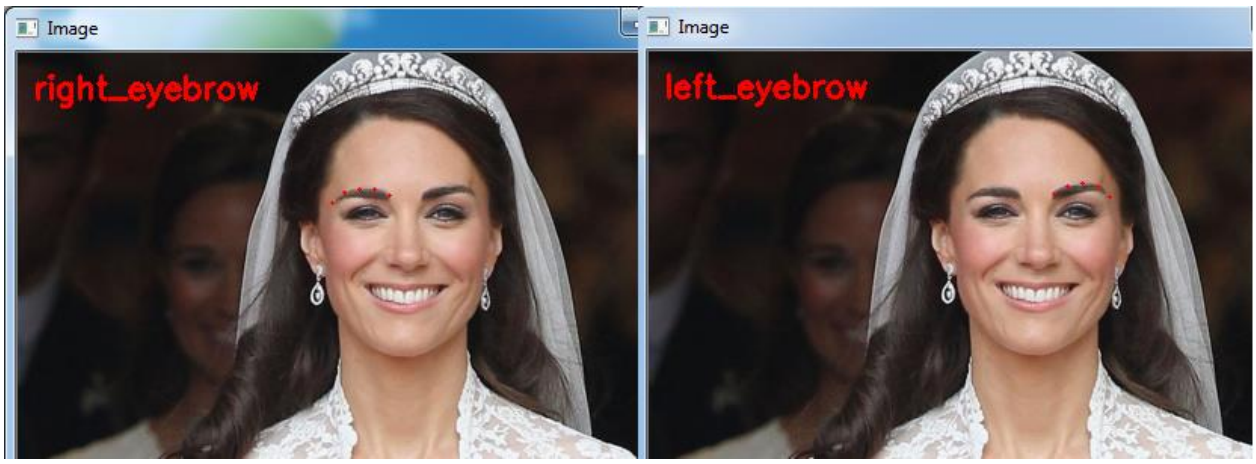


Рисунок 5 – Точки, описывающие брови

Рисунок 6 – Точки, описывающие глаза

Рисунок 7 – Точки, описывающие нос

Рисунок 8 – Точки, описывающие челюсть

На рисунке 9 показан результат работы скрипта, распознающего лицо на видео потоке.

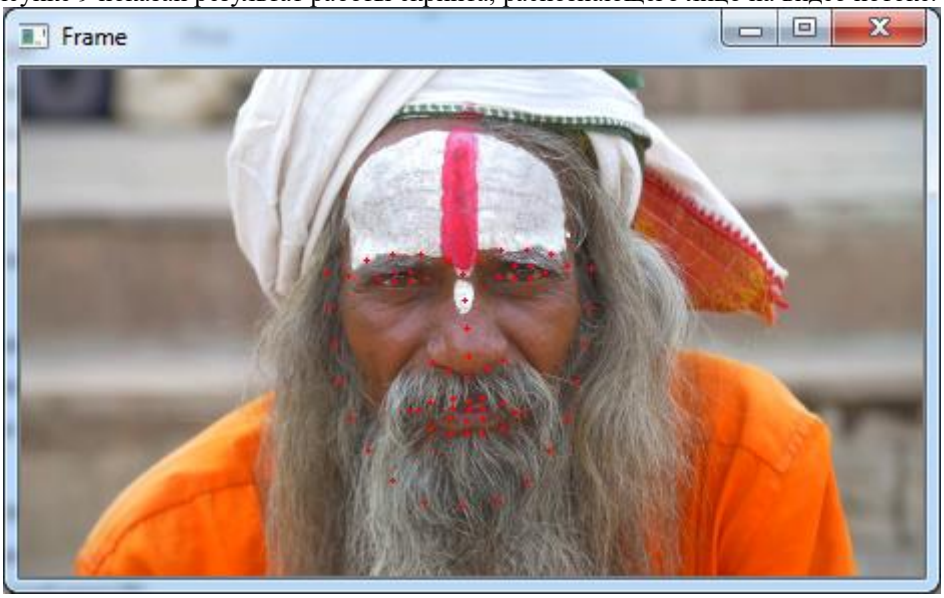


Рисунок 9 – Результат распознавания лица на видео

Исходный код алгоритма распознавания лиц

Исходный код facial-landmarks на языке Python

```
from imutils import face_utils
import numpy as np
import argparse
import imutils
import dlib
import cv2
ap = argparse.ArgumentParser()
ap.add_argument("-p", "--shape-predictor", required=True,
	help="path to facial landmark predictor")
ap.add_argument("-i", "--image", required=True,
	help="path to input image")
args = vars(ap.parse_args())
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])
image = cv2.imread(args["image"])
image = imutils.resize(image, width=500)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
rects = detector(gray, 1)
for (i, rect) in enumerate(rects):
	shape = predictor(gray, rect)
	shape = face_utils.shape_to_np(shape)
	 (x, y, w, h) = face_utils.rect_to_bb(rect)
	cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
	cv2.putText(image, "Face #{}".format(i + 1), (x - 10, y - 10),
		cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
	for (x, y) in shape:
		cv2.circle(image, (x, y), 1, (0, 0, 255), -1)
cv2.imshow("Output", image)
cv2.waitKey(0)
```

Исходный код detect-face-parts на языке Python

```
from imutils import face_utils
import numpy as np
import argparse
import imutils
import dlib
import cv2
ap = argparse.ArgumentParser()
ap.add_argument("-p", "--shape-predictor", required=True,
	help="path to facial landmark predictor")
ap.add_argument("-i", "--image", required=True,
	help="path to input image")
args = vars(ap.parse_args())
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])
image = cv2.imread(args["image"])
image = imutils.resize(image, width=500)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
rects = detector(gray, 1)
for (i, rect) in enumerate(rects):
	shape = predictor(gray, rect)
	shape = face_utils.shape_to_np(shape)
	for (name, (i, j)) in face_utils.FACIAL_LANDMARKS_IDXS.items():
		clone = image.copy()
		cv2.putText(clone, name, (10, 30), cv2.FONT_HERSHEY_SIMPLEX,
			0.7, (0, 0, 255), 2)
		for (x, y) in shape[i:j]:
			cv2.circle(clone, (x, y), 1, (0, 0, 255), -1)
		(x, y, w, h) = cv2.boundingRect(np.array([shape[i:j]]))
		roi = image[y:y + h, x:x + w]
		roi = imutils.resize(roi, width=250, inter=cv2.INTER_CUBIC)
		cv2.imshow("ROI", roi)
		cv2.imshow("Image", clone)
		cv2.waitKey(0)
	output = face_utils.visualize_facial_landmarks(image, shape)
	cv2.imshow("Image", output)
	cv2.waitKey(0)
```

Исходный код real-time-facial-landmarks на языке Python

```
from imutils.video import VideoStream
from imutils import face_utils
import datetime
import argparse
import imutils
import time
import dlib
import cv2
ap = argparse.ArgumentParser()
ap.add_argument("-p", "--shape-predictor", required=True,
	help="path to facial landmark predictor")
ap.add_argument("-v", "--video", required=True,
	help="path to input video file")
args = vars(ap.parse_args())
detector = dlib.get_frontal_face_detector()
predictor = dlib.shape_predictor(args["shape_predictor"])
vs = cv2.VideoCapture(args["video"])
while True:
	(grabbed, frame) = vs.read()
	if not grabbed:
		break
	frame = imutils.resize(frame, width=450)
	gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
	rects = detector(gray, 0)
	for rect in rects:
		shape = predictor(gray, rect)
		shape = face_utils.shape_to_np(shape)
		for (x, y) in shape:
			cv2.circle(frame, (x, y), 1, (0, 0, 255), -1)
	cv2.imshow("Frame", frame)
	key = cv2.waitKey(1) & 0xFF
	if key == ord("q"):
		break
vs.release()
cv2.destroyAllWindows()
```

Лист регистрации изменений к РПД

	Номер протокола заседания кафедры, дата утверждения изменения	Количество страниц изменения	Подпись разработчика РПД
1	12.10.2021 протокол №2. Воспитательная работа обучающихся. Основание: *Федеральный закон от 31.07.2020 N 304-ФЗ "О внесении изменений в Федеральный закон "Об образовании в Российской Федерации" по вопросам воспитания обучающихся"*	1	
2	12.10.2021 протокол №2. Практическая подготовка обучающихся. Основание: *Приказ Министерства науки и высшего образования Российской Федерации, Министерства просвещения Российской Федерации от 05.08.2020 г. № 885/390 "О практической подготовке обучающихся"*	1	