Программа учебной дисциплины «Языки программирования»

Утверждена
Академическим советом ОП
Протокол № от __.__.20__

Разработчик	Булгаков С.А., ассистент Департамента прикладной математики МИЭМ НИУ ВШЭ Чеповский А.А., к.ф.-м.н., доцент, доцент Департамента прикладной математики МИЭМ НИУ ВШЭ
Число кредитов	8
Контактная работа (час.)	136
Самостоятельная работа (час.)	168
Курс, Образовательная программа	1 курс, для специальности 10.05.01 «Компьютерная безопасность» подготовки специалиста
Формат изучения дисциплины	без использования онлайн курса

1. Цель, результаты освоения дисциплины и пререквизиты

Дисциплина «Языки программирования» призвана обучить студентов принципам функционирования информационных технологий, базовым концепциям технологий программирования и эффективных реализаций алгоритмов.

Цели освоения дисциплины:

- **Знакомство** слушателей дисциплины с основными парадигмами и теоретическими основами программирования, терминологией объектно-ориентированного программирования.
- **Формирование** базовых компетенций, связанных с разработкой программного обеспечения при решении профессиональных задач и представления о возможностях и особенностях объектно-ориентированных языков программирования при проектировании, разработке и отладке компьютерных программ;
- **Изучение** теоретических основ программирования, методов реализации алгоритмов различного типа, базовых принципов объектно-ориентированного программирования, основных подходов машинного обучения;

- **Выработка навыков** программирования и алгоритмизации с применением современных процедурных и объектно-ориентированных языков программирования;
- **Получение опыта** работы с механизмами статического полиморфизма, такими как шаблоны функций и классов; поиска эффективных реализаций различных алгоритмов, практических навыков разработки прикладных программ, в том числе для методов анализа данных.

Теоретический материал курса подкрепляется семинарскими практическими занятиями по программированию в целях реализации предлагаемых заданий в рамках изучаемой тематики.

В результате освоения дисциплины студент должен:

- **Знать** парадигмы и методологии программирования; общие принципы построения и использования современных языков программирования высокого уровня; особенности наиболее распространенных объектно-ориентированных языков программирования; базовые структуры данных; основные комбинаторные и теоретико-графовые алгоритмы; современные технологии программирования;
- **Уметь** формализовать поставленную задачу; работать с интегрированными средами разработки программного обеспечения; применять в профессиональной деятельности современные объектно-ориентированные языки программирования; профессионально решать задачи производственной и технологической деятельности с учетом современных достижений науки и техники; оценивать качество готового программного обеспечения;
- **Владеть** базовыми математическими знаниями и информационными технологиями, эффективно применять их для решения научно-технических и прикладных задач, связанных с развитием и использованием информационных технологий, математическими алгоритмами и методами автоматизации сбора и анализа данных при построении систем машинного обучения;
- **Иметь навыки (приобрести опыт)** постановки математических и информационных задач; разработки, документирования, тестирования и отладки программ; описания алгоритмов решения поставленной задачи и разработки программного кода на языке *C++;* тестирования программного обеспечения; разработки алгоритмов решения типовых профессиональных задач.

Уровни формирования компетенций:

РБ – ресурсная база, в основном теоретические и предметные основы (знания, умения);

СД – способы деятельности, составляющие практическое ядро данной компетенции;

МЦ – мотивационно-ценностная составляющая, отражает степень осознания ценности компетенции человеком и готовность ее использовать.

2. Содержание учебной дисциплины

В результате освоения дисциплины студент осваивает следующие компетенции:

Компетенция	Код по ФГОС/ НИУ	Уровень форми-рования компетенц ии	Дескрипторы – основные признаки освоения (показатели достижения результата)	Формы и методы обучения, способствующие формированию и развитию компетенции	Форма контроля уровня сформирован ности компетенции
Способен решать проблемы в профессиональной деятельности на основе анализа и синтеза	УК-3	СД	Подбор и анализ материалов по теме задания. Самостоятельно пишет программы по заданию преподавателя	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.
Способен оценивать потребность в ресурсах и планировать их использование при решении задач в профессиональной деятельности	УК-4	РБ	Владеет навыками разработки алгоритмических и программных решений с использованием современных технологий программирования	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.
Способен работать с информацией: находить, оценивать и использовать ин формацию из раз личных источников, необходимую для решения научных и профессиональных задач (в том числе, на основе системного подхода)	УК 5	РБ	Применяет современные стандартные среды разработки (IDE) при создании и отладке программных продуктов	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.
Способен вести исследователь скую деятельность, включая анализ проблем, постановку целей и задач, вы деление объекта и предмета исследования, выбор	УК-6	РБ	Понимает стадии и этапы разработки программного обеспечения. Демонстрирует знание современных языков программирования низкого и высокого уровня	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.

Компетенция	Код по ФГОС/ НИУ	Уровень форми-рования компетенц ии	Дескрипторы – основные признаки освоения (показатели достижения результата)	Формы и методы обучения, способствующие формированию и развитию компетенции	Форма контроля уровня сформирован ности компетенции
способа и методов исследования, а также оценку его качества					
Способен проектировать и разрабатывать компоненты программного обеспечения на основе современных парадигм, технологий и языков программирования	ПК-4	СД	Разрабатывает техническое задание на разработку программных компонент автоматизированных систем.	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.
Способен применять знания жизненного цикла современных проектов по созданию и эксплуатации программных систем и инструментальные средства управления проектами в области ИТ.	ПК-8	СД	Демонстрирует знание современных языков программирования	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.
Способен использовать и развивать методы математического моделирования и применять аналитические и научные пакеты прикладных программ	ПК-11	МЦ	Применяет современные стандартные среды разработки (IDE) при создании и отладке программных продуктов	Лекции, семинарские занятия, самостоятельное выполнение домашних заданий.	Домашние и аудиторные задания, лабораторные работы, экзамен.

Содержание разделов дисциплины:

№	Наименование раздела дисциплины	Содержание раздела	Аудиторная работа	Самостоятельная работа	Литература к разделу
1.	Введение в языки программирования	Общие сведения о языках программирования. Инструментальные средства разработки ПО. Основные понятия архитектуры ЭВМ.	12	4	[5, 13]
2.	Знакомство с ООП.	Структурные типы данных. Абстракция, инкапсуляция, наследование. Полиморфизм.	8	8	[1, 5, 7]
3.	Введение в программирование на C++.	Общая структура программы. Пространства имен. Ввод/вывод в C++. Операции инкремента и декремента. Особенности операторов инкремента и декремента при работе с указателями. Функции. Ссылки. Выделение/освобождение памяти в C++. Константы и макроопределения. Перегрузка функций.	16	12	[1-3, 5]
4.	Построение пользовательских типов данных.	Понятие объекта и класса. Создание пользовательских классов. Принцип композиции при конструировании новых классов. Инициализация данных объекта класса (Конструктор по умолчанию, конструктор с параметрами). Назначение деструктора. Способы передачи объекта в функцию, конструктор копирования. Перегрузка операторов. Статические переменные и методы класса. Константные методы.	20	28	[1-3, 5]
5.	Знакомство с библиотекой STL.	Классы стандартной библиотеки для работы со строками. Классы стандартной библиотеки для работы с контейнерами. Операции с объектами стандартной библиотеки. Преобразование типов.	20	24	[1-3, 5, 8]
6.	Обработка исключений.	Понятие исключения. Механизм генерации и обработки исключений.	4	12	[1-3, 5]
7.	Наследование, виртуальные функции, полиморфизм.	Спецификаторы доступа public, protected, private. Понятие и механизм наследования классов. Преимущества и недостатки наследования по сравнению с композицией. Перегруженные методы. Виртуальные функции, понятие полиморфизма. Чистые виртуальные функции и абстрактные классы. Доступ к объекту через указатель, преобразование типов. Непрямые базовые классы. Виртуальный деструктор. Множественное наследование.	16	24	[1-3, 5]
8.	Шаблоны C++.	Понятие обобщенного программирования. Определение шаблона класса. Создание объектов шаблона класса. Наследование шаблонных классов. Шаблоны функций.	16	20	[1-3, 5]
9.	Использование C++ в алгоритмах на графах.	Основные алгоритмы на графах и структуры данных, используемых в них, поиск в глубину, поиск в ширину, алгоритм Дейкстры, минимальное остовное дерево. Более сложные алгоритмы на графах.	18	24	[13]

3. Оценивание

Элемент контроля	Наименование элемента контроля	1 год				2 год		Параметры
		1	2	3	4	1	2	
Промежуточный	Экзамен			*				Создание программы в IDE и защита ее текста, 60 минут
Итоговый	Экзамен						*	Создание программы в IDE и защита ее текста, 60 минут

Существуют следующие элементы контроля:

- Выполнение домашних заданий;
- Выполнение лабораторных заданий;
- Текущий контроль на лекциях и семинарских практических занятиях;
- Экзамен в конце модулей: 3 первого года (промежуточный);
- Экзамен в конце модулей: 2 второго года (итоговый).

Порядок формирования оценок по дисциплине

Преподаватель оценивает работу студентов на практических занятиях: оценивается активность студента в дискуссиях, скорость и правильность решения задач. Оценки за работу на занятиях преподаватель выставляет в рабочую ведомость.

Преподаватель оценивает самостоятельную работу студентов: оценивается правильность и количество реализованных заданий. Оценки за самостоятельную работу студента преподаватель выставляет в рабочую ведомость.

Оценка $O_{текущего\ контроля\_1\text{-}3}$ за элементы текущего контроля 1-3 модуля учитывает результаты студента по всем элементам текущего контроля за соответствующие модули.

Результирующая оценка за текущий контроль в форме экзамена по итогам 3 модуля рассчитывается следующим образом:

$$O_{промеж} = 0{,}5 \cdot O_{текущего\ контроля\ 1\text{-}3} + 0{,}5 \cdot O_{экзамен}$$

Оценка $O_{текущего\ контроля\_4\text{-}6}$ за элементы текущего контроля 4-6 модуля учитывает результаты студента по всем элементам текущего контроля за соответствующие модули.

Результирующая оценка за итоговый контроль в форме экзамена по итогам 2 модуля 2 года (6 модуля) выставляется по его результатам – $O_{результ}$.

$$O_{результ} = 0{,}25 \cdot O_{экзамен} + 0{,}25 \cdot O_{текущего\ контроля\ 4\text{-}6} + 0{,}5 \cdot O_{промеж}$$

Способ округления: арифметический.

На пересдаче студенту не предоставляется возможность получить дополнительный балл для компенсации оценки за текущий контроль.

Блокирующие элементы не предусмотрены.

4. Примеры оценочных средств

Основным оценочным средством на экзамене и для текущих элементов контроля является создание программы в IDE и защита ее текста.

Примерный перечень вопросов к зачету (экзамену) по всему курсу или к каждому промежуточному и итоговому контролю для самопроверки студентов.

1. Классификация языков программирования. Парадигмы программирования.
2. Объектно-ориентированный подход. Основные понятия.
3. Принципы наследования и композиции при конструировании новых классов. Преимущества и недостатки наследования по сравнению с композицией.
4. Инициализация данных объекта класса (Конструктор по умолчанию, конструктор с параметрами). Назначение деструктора. Способы передачи объекта в функцию, конструктор копирования.
5. Перегрузка операторов.
6. Статические переменные и методы класса. Константные методы. Перегруженные методы.
7. Виртуальные функции, понятие полиморфизма. Чистые виртуальные функции и абстрактные классы. Виртуальный деструктор.
8. Множественное наследование.
9. Понятие обобщенного программирования. Шаблоны классов. Шаблоны функций.
10. Понятие исключения. Механизм генерации и обработки исключений.
11. Основные алгоритмы на графах и структуры данных, используемых в них, поиск в глубину, поиск в ширину, алгоритм Дейкстры, минимальное остовное дерево.
12. Более сложные алгоритмы на графах.

Блокирующие элементы не предусмотрены.

5. Ресурсы

5.1. Рекомендуемая основная литература

1. Г. Шильдт. C++: руководство для начинающих, 2-е издание. : Пер. с англ. – М. : Издательский дом «Вильямс», 2005. – 672 с.
2. Б. Страуструп. Программирование и практика с использованием C++, 2-е издание. : Пер. с англ. – М. : ООО «И.Д. Вильямс», 2016. – 1328 с.
3. Б. Страуструп. Язык программирования C++. Специальное издание. Пер. с англ. – М.:Издательство Бином, 2011 г. – 1136 с.
4. Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. Приемы объектно-ориентированного проектирования. Паттерны проектирования – СПб: «Питер», 2007. – 366 с.
5. Лебедев П.А. Языки программирования (рабочая версия).

5.2. Рекомендуемая дополнительная литература

6. Qt Documentation. [Электронный ресурс]. URL: http://doc.qt.io/
7. Русское Qt-сообщество. Программирование Qt. [Электронный ресурс]. URL: http://qt-doc.ru/
8. Л.В. Городняя. Парадигмы программирования: Курс лекций. – Новосибирск: НГУ, 2007.
9. М. Шлее, Qt 4.5. Профессиональное программирование на C++ – БХВ-Петербург, 2009, 896
10. Херн Д., Бейкер М., Компьютерная графика и стандарт OpenGL, Вильямс, 1168 с
11. С. Майерс. Эффективное использование STL. — СПб: Питер, 2002. — 224 с.
12. Boost Documentation. [Электронный ресурс]. http://www.boost.org/doc/
13. Кнут Д.Э. Искусство программирования. Томы 1-3. Основные алгоритмы. М. Вильямс, 2012, третье издание.

5.3. Программное обеспечение

Для успешного освоения дисциплины, студент использует следующие программные средства:

- Компилятор языка C++ стандарта не ниже C++11;
- Интегрированная среда разработки (рекомендуется Qt Creator);
- Дополнительные инструменты разработки (отладчики, системы сборки,…) в зависимости от используемых инструментов;
- Различные сторонние библиотеки, зависящие от заданий, выполняемых студентом.

5.4. Профессиональные базы данных, информационные справочные системы, интернет-ресурсы (электронные образовательные ресурсы)

14. Г. Шильдт, полный справочник по Си++ , 4-е изд.: Пер. с англ. – М.: Вильямс, 2006. — 800 с.

5.5. Материально-техническое обеспечение дисциплины

Для проведения лекций необходима аудитория, оснащенная компьютером и проекционным оборудованием.

Для проведения практических занятий необходимо наличие компьютерного класса, оборудованного ПК из расчёта одно рабочее место на обучаемого. На данных ПК требуется наличие ПО, перечисленного в п. 10.5. Для упрощения работы с материалами желательно наличие доступа к внешним носителям. Для упрощения доступа к справочным материалам желательно наличие доступа к сети Интернет.

6. Особенности организации обучения для лиц с ограниченными возможностями здоровья и инвалидов

Не предусмотрены.